The Role of the Magnocellular System in Implicit Cognition

Douglas John Mansfield

MNSDOU002

Dissertation submitted for the requirements of the Degree of

DOCTOR OF PHILOSOPHY

In the department of Psychology

Faculty of Humanities

University of Cape Town

November 2014

Declaration

This work has not been previously submitted in the whole, or in part, for the award of any degree. It is my own work. Each significant contribution to, and quotation in this dissertation from the work, or the works of other people has been attributed, and has been cited and referenced.

Signature_____ Date_____

Acknowledgments

I wish to sincerely thank my supervisor, Prof. Colin Tredoux for his patience, guidance and support throughout the process of this research. I want to express my gratitude to my mother, Mrs. V. Judd, for her support, kindness and assistance, and my friends David and Letitia Lea and Gustav Schloms. I also would like to thank Embarcadero Technologies, Inc. and their African distributer: EOH, South Africa for providing academic versions of Delphi and C++ software development systems and for providing training and active support for software developers in South Africa. Importantly, I would like to express my thanks to friends and colleagues at the University of KwaZulu-Natal, especially Ms. Viv O'Neill, Dr. Bev Killian, Dr. Mike Quayle, Prof. Graham Lindegger, Prof. Kevin Durrheim. Professors Sid Segalowitz and Jane Dywan from Brock University in St Catharines, Ontario have been especially important in the way that they have inspired and supported me. Thank you to Alicia Nortjie for being an extremely reliable, resourceful and capable research assistant at the University of Cape Town. I especially want to thank to thank the participants at UKZN and UCT. Needless to say, the research would not have been possible without their participation. Thank you in particular to Nick Collins for helping to proof-read various drafts and wade through some difficult and technical material. A special thanks to Microsoft, the staff on their call centres and their technical staff for assisting with complications in installing *Microsoft Office* on different platforms.

Abstract

Implicit cognition paradigms, such as the IAT (Implicit Association Test) are not well understood and are frequently thought to involve 'unconscious' attitudes. Although there has been a theoretical shift away from psychoanalytic ideas about consciousness towards more cognitively orientated views, a mystique lingers. The magnocellular (M) system is thought to be involved in rapid but coarse information processing which is a basis for certain kinds of automaticity in information processing, such as reading or visual object recognition. The question this research addressed concerns visual perceptual processes which are not easily controllable and which occur without conscious effort. The role of the M system was investigated in word recognition, object recognition and race feature recognition in a series of experiments. There is evidence that the M system facilitates word recognition in terms of detection accuracy. An experiment involving object recognition replicated the research of Kveraga, *et al*. (2007b) and similar results were obtained in that when objects were presented under a condition which favoured the M system, recognition accuracy was significantly better and reaction time was significantly shorter than when objects were presented in a condition which favoured the parvocellular (P) system. A series of IAT experiments replicated findings from 1) species, 2) race experiments and similar results were obtained to those reported in the literature. The race IAT experiment was then adapted to use images that were biased towards either the M or P visual systems. As the M system appears to facilitate object recognition, probably by a top-down processing mechanism which may be associated with perceptual automaticity, it was predicted that IAT scores would not be affected in this condition. It was predicted that when images were presented in a condition which inhibited M system functioning, IAT scores would be more neutral (suggesting less response bias). There was a trend which supported this prediction, but the summary score analysis did not show a statistically significant difference. When the data was decomposed and the reaction time data was analysed, there was evidence of race-related perceptual dynamics. The M system appears to facilitate processing of race-related facial features in the ethnic groups which showed more implicit prejudice, but not to the same extent, or in the same manner in the low prejudice ('Black') group. A possible interpretation is that racial features have different degrees of salience in these groups. It seems likely that perceptual automaticity has an important role in implicit cognition and the M system appears to be an important part of this mechanism.

Table of Contents

Table of Figures

List of Tables

Preface

Software

In this research project, the computer-based experimental procedures were written for the Windows platform and customised for local conditions. One reason that the more ubiquitously used experimental packages, such as *E-Prime* or *Matlab* were not used, is that they were not available because of their relatively high cost and my limited budget. Another reason is that many of the experiments had to be run in different locations (the University of KwaZulu-Natal and the University of Cape Town which are 1800 km apart. At the time these experiments were run, it was not known whether *E-Prime* or *Matlab* could be adapted to use remote databases and support multiple, concurrent groups of users in different regions. Data integration was an important challenge in this experiment. It was for these reasons that software was written to run on the Windows platform which was available in all of the experimental locations and could be easily customised to suit conditions in the different laboratories. Inevitably, this meant that a lot of time and effort had to be devoted to writing and testing software before it could be deployed and used in the various experiments.

The experiments were run on Windows pc work stations with LCD monitors. A refresh rate of 60 Hz was selected for all the experiments and this means that the screen is updated every 16.67 ms. Although the calibration procedure in Experiments 1, 2, 4 involved minimum exposure times of approximately 9 ms, stimuli displayed for less than 16.67 ms would not necessarily have been consistently visible. In testing this procedure however, it was found that at very brief exposure times stimuli were frequently perceptible. Nevertheless, a constant of 50 ms was added to the calibration values so that in the test phases, exposures were in a range that would not be affected by screen refresh rate. Apart from the fact that many different monitor types were used, it was not possible to test the accuracy of the exposure times at the actual screen because specialized equipment is necessary (see Simmons, 2014). However, the individual calibration procedure was intended to correct for variation in monitor characteristics. When exact exposure times are given, these are based on timing with the CPU clock, not actual exposure on the screen.

I have made the computer code available to any interested readers in a *Dropbox* location which will be shared on request to Prof. Colin Tredoux (colin.tredoux@uct.ac.za). Other resources, such as demonstration applications and files will be available in this location.

IAT Scoring Procedure

In order to avoid confusion for the reader, I would like to point out that in the current IAT experiments, block 3 always refers to what is typically the 'easy' (or compatible) sort (for example, flowers and pleasant words). Block 5 always refers to what is typically the 'difficult' (or incompatible) sort (for example, insects and unpleasant words). In every case that these blocks (3, 5) are mentioned, these (compatible, incompatible) sorts are implied. Although the ordering of the blocks was varied randomly, the data was uploaded according to this block naming convention and the nature of the task (compatible/incompatible) is always the same. Using this naming convention, calculation of the d score involves subtracting the block 5 mean from the block 3 mean, divided by the combined standard deviation. This means that the score is usually negative because the incombatible block mean is subtracted from the compatible block mean, and this is equivalent in meaning to a positive score on other IAT experiments if the sign is reversed. Therefore, the scores are essentially similar if they are simply negated (-0.4 in the current experiments is equivalent to 0.4 in the standard notation). It is therefore an idiosyncracy of this experiment and the score interpretation is identical.

Screen luminance values

A full demonstration of screen lumination concepts and 'RGB' convention for defining these concepts appears in Appendix 6. The following is a brief explanation.

A line drawing has values which range from 0 – 255 ($FF in hex, depending on the notation). The brightest colour that a computer screen can display is pure white (RGB 255, 255, 255). That is, each pixel is illuminated to its maximum value of 255. This is a decimal value that corresponds to $FF in hex, or 11111111 in binary. An LCD or CRT screen contains very small dots or pixels which vary in illumination from 0 to 255 ($FF). There are Red, Green and Blue coloured pixels which vary by the same amount (i.e. 0 – 255). The method of colour mixing for a computer screen is the same as that used in a television screen – namely, additive colour mixing. It is not possible to measure the actual contrast at each screen, since each varies according to the design, manufacturer, the setting in the computing lab., and ambient light levels. It was not possible to control these exactly, which is why the colours and contrast could only be determined natively as RGB colour settings.

Synopsis

The view taken in this dissertation is that the tradition known as *Functionalism* is a foundation for various, empirically orientated disciplines, such as Social Neuroscience. Social Neuroscience provides a framework for understanding behaviour, consciousness and other psychological phenomena and I argue that cognition serves purposes that can have to do with the physical and social environment. I do introduce this idea early in the Introduction and the discussion is intended to establish some of the broad philosophical perspectives before I explore the more general psychological perspectives that inform the general frame of reference for this research. I assert that overwhelmingly, neuroscientists have abandoned Cartesian ontology in favour of identity theories. The philosophical perspective of Neuroscience provides at least for degree of compatibility between the methods of the natural and social sciences (see Jost, Nam, Amodio & Van Bavel, 2014).

From a comparative perspective, there are similarities between the brains of humans and non-human animals and much of our knowledge of vision, for example, has been derived from studies of monkey brains. Primate studies have also proved useful in studying interactions between social variables, brain states and neuroendocrinology and have proved to be relevant to humans (see Gunnar & Cheatham, 2003; Uno, Tarara, Else, Suleman, & Sapolsky, 1989). The requirements of sociability are presumed to have had a major influence on the structure and function of the mammalian brain (Gawronski & Cesario, 2013). Social cognition provides a framework for understanding consciousness and its significance. I explore aspects of social cognition from a neuroscientific and developmental perspective in order to lay a foundation for understanding implicit cognition and its role in social behaviours.

Implicit cognition theory involves the idea that individuals do not always, or necessarily have insight or access into the social perceptions, judgements and attitudes on which their behaviours are based. Implicit cognition involves psychological attributes, processes and response tendencies that persons may not be able to access or report. Although there is a debate about the extent to which attitudes are conscious, there seems to be some agreement that perceptual processes may predispose individuals to stereotype-congruent responses and behaviours unless higher cognitive processes are activated. It is proposed that the magnocellular (M) system is a component of the visual system which promotes efficiency in various perceptual and cognitive systems and that this contributes to a degree of automaticity

in routine cognitive processing and spontaneous responding. It is this tendency to respond in familiar ways that promotes socially learned attitudes and behaviours and is thought to involve to some degree, the M system.

One of the ways in which this automaticity may be manifest, is in affective responses to visual (and other perceptual information). Emotionality has important and valuable influences on social behaviour and without emotional awareness, social functioning would be impaired. The visual system provides important inputs to the limbic system and is thus a major source of moment by moment emotional input that assists with cognition, decisions and behaviour. The visual system performs constant, rapid and complex processing which keeps the individual abreast of events and changing states of the world. It is efficiently and sensitively tuned to reading faces and facial emotions. Faces give information about an individual's identity, mood state, intentions, rank and many other social variables (see Ratner, Dotsh, Wigboldus, van Knippenberg & Amodio, 2014) which facilitate rapid decisions about relationships, alliances and ultimately promote survival and reproductive fitness.

In the neuroscience literature on implicit cognition, the amygdala has become a central focus for understanding implicit cognition since it is understood to be highly responsive to emotional facial expressions various affective nuances. The implicit cognition literature has emphasised the role of the amygdala in prejudice, and its involvement in responses to 'other' races. Part of its significance is in the manner in which it responds and produces arousal which is thought to be 'automatic', 'preconscious' and unintended. The role of affect in implicit cognition is considerable and the various limbic structures such as the amygdala, insula and striatum are regarded as singularly important to understanding the 'neural basis of prejudice' (Amodio, 2014).

The amygdala is implicated in producing race-related (threat) arousal, and according to Amodio (2014) it receives direct efferents from sensory systems which enables it to respond very rapidly to immediate threats in advance of more elaborative processing of stimuli' (p. 671). The role of the amygdala is discussed in Section 5.3. The general idea is that there are brain systems which detect threat-related stimuli very rapidly and tend to implement survival-related responses which tend to lead to immediate action. Inhibition and control are important cognitive functions in social behaviour. Cognitive control may be exerted, for example, when there is conflict between stereotype congruent arousal and high level cognitive goals to 'act

without bias' and potentially diminishes the way that visual information would otherwise tend to bias this processing.

Emotional processing is done in real-time, is ongoing and relatively automatic, so one of the major processing challenges is the recognition, utilisation and sometimes inhibition of arousal, and associated, or habituated response patterns. To exert control, one has to be able to process perceptual information with a degree of awareness and override pre-potent responses with more measured and strategic decisions. Dual Systems theory provides a useful frame of reference for implicit cognition in that it posits two systems of *impulse* and *self-control* which relate logically to the systems of *perceptual automaticity* and *conscious reflection* and *inhibition* which I have mentioned. Control theory is theoretically important for implicit cognition.

One of the important innovations which has emerged in recent years in social psychology research is the Implicit Association Test (IAT). It was devised after decades of research into social attitudes proved to have major limitations in the measurement of attitudes and the prediction of actual behaviour. One of the major limitations of attitude research is the problem that people tend to respond to questionnaires in socially desirable ways. Instead of direct measurement of participants' responses to various propositions, the IAT measures the strength of association between social objects and evaluative categories. This is done by means of response timing, and inferences are drawn about the relative speed of associating various exemplars with evaluative terms. Longer response times are understood to represent incongruence, and vice versa. The basic measure is thus indirect and to this extent is regarded as implicit.

The IAT is a computer administered task which requires participants to respond rapidly to pictures and words with various key-presses. It involves visual perceptual processes, cognitive processing in the form of decisions, and manual responses. It is a task which places participants under pressure to respond quickly and accurately. In some ways, this equates to the pressures one faces in everyday life in that one has to make rapid decisions, often with minimal opportunities for reflection. This made the IAT seem like a suitable research instrument to test hypotheses about the visual system and its contribution to fast decision making – in particular of the kind that the IAT requires. Since the IAT requires rapid visual perception and responses, it thus seemed to be suitable for testing hypotheses regarding the

visual system. The magnocellular (M) system is thought to be a relatively fast system which is in a position to potentiate quick responses, both because of its rapid conduction properties and the cortical circuits it appears to activate. It is also thought to optimise rapid visual perceptual processes and decisions by means of a top-down mechanism. Research shows that it is involved in facilitating object recognition by means of this top-down cognitive process. It was hypothesised that that the M system might also have an important role in recognising facial metrics and race-related features. If the M system is involved in rapid perceptual processing of facial features, such as emotional markers, race (possibly social rank, and other socially salient variables), it was hypothesised that the M system could contribute to response automaticity that is associated with implicit prejudice. In other words, the M system was thought to be associated with the perception and activation of higher level cognitive information which corresponds to social attitudes.

A critical focus for this research was to replicate the research of Kveraga, Boshyan & Bar (2007b) which supported their hypotheses that the magnocellular system provides a low spatial frequency signal which facilitates object recognition. The research process which lead up to a replication of their experiment began with two experiments that examined the role of the M system in reading. One of the major problems was in finding a colour solution that would dissociate the parvocellular (P) and M visual systems. This was both a practical and a theoretical problem. The problem lay in finding colour combinations that were isoluminant, so that the contribution of the M system could be partialled out. The first word recognition experiment did not produce convincing evidence that the initial M-biasing colour solution was effective. However, the second experiment using a different solution was more promising.

The problem of how to dissociate these visual systems was investigated methodologically and a different isoluminance solution to that of Kveraga, *et al.* (2007b) seemed attractive. This prompted an extensive search of the literature on visual perception to try and understand the architecture and neurophysiology of the visual system, so that a plausible theoretical rationale for the methodology could be established. Chapter 1.4 provides a detailed review and discussion of the literature and the implications for the (heterochromatic flicker photometry) methodology are drawn out. The theoretical reflection was important to this research, since the methodology departed from what has been traditionally used and what has practically become conventional wisdom.

Another approach to investigating the M system was followed. The motion illusion phenomenon suggested that the M system might be involved in illusory motion. There has been a debate about whether it is possible to generate isoluminant motion stimuli (Lu, Lesmes & Sperling, 1999). Yeshurun (2004) has suggested that the M system is responsible for the processing of motion, so the inhibition of M system functioning by means of isoluminant stimuli should impair such illusions. It does seem that motion illusions such as the Rotating Snakes Illusion by Kitaoka & Ashida (2003) can be altered when the colours are interfered with. This is possibly because the luminance gradients are altered. Another related perceptual phenomenon, known as the 'Barber's Pole Illusion' was also investigated.

The third experiment was a precursor to the replication experiment and it was built with the theorising and methodological experimentation in mind. This experiment was moderately successful. The fourth experiment appeared to support the contention that the traditional isoluminance solution was not easy to implement and it appeared that the new isoluminance solution is theoretically and practically viable.

The Replication experiment successfully reproduced the research of Kveraga, *et al.*, (2007b), albeit with some methodological variation (which I introduced and discussed above). The results support those of Kveraga *et al.* and their inference that the M system does appear to facilitate object recognition via some top-down mechanism. This finding cleared the way for the final series of experiments which tested the idea that the M system is involved in implicit cognition.

Three IAT studies were done: firstly the 'species' IAT, which tested the program and procedure that were devised. The results fit the general pattern of findings seen previously. The second IAT collected baseline data on 'Race'. Again, the results fitted those of other IAT experiments. There were challenges that had to be solved in the grouping of participants. The major question concerned the grouping of 'White' and other minority groups vs. the 'Black South African' participant group. It turned out that the 'White' and minority groups were very similar in terms of their score patterns and both groups showed significantly higher IAT (d) scores than the 'Black' group, when they were compared individually, and when they were combined. The IAT trial blocks were decomposed and the reaction times were analysed. The

resulting analysis showed interesting patterns of differences between the various ethnic groups and gave a useful overall picture of responses to various stimuli and stimulus combinations based on reaction time.

The final IAT experiment was the same as the previous 'Race' experiment, but it used M-biased and P-Biased stimuli in order to investigate the contribution of the M system in implicit cognition. The same grouping strategy was used again and the pattern was virtually identical to that of the previous experiment. The null hypothesis was accepted when the summary data was analysed, but it appears that there were different degrees of facilitation between the various ethnic groups. A case may be made that in the low bias (generally 'Black') group, the effects of the stimulus biasing were cancelled out. In any event, there appeared to be interesting race-related perceptual dynamics which emerged from the reaction time analysis. In this analysis, there seemed to be clear effects of the stimulus biasing condition and differential facilitation effects in the different critical blocks. There appeared to be gender, ethnic group and task-related dynamics. Reaction time was generally faster for M-biased pictures than P-biased pictures by about 115 ms. The overall result suggested that 'Black' participants were less reactive to race-based facial features than the other groups. This effect was limited to the critical trial blocks where pictures were presented in either the M- or P-biased condition. No such effect was seen in the neutral blocks for pictures in the control condition. It appears that when images were presented in the P-biased condition, all participants reaction times appeared to be less reactive to the task conditions. The conclusion was drawn that perceptual discrimination was poorer when images were presented in the P-biased condition, but greater when images were presented in the M-biased condition.

The conclusion was drawn that there is evidence of magnocellular facilitation in the processing of race-related facial features. While no clear conclusion could be drawn with regard to magnocellular involvement in implicit cognition *per se* (playing a role in response bias or automaticity), there does seem to be evidence that the M system is involved in processing race-related facial features.

Chapter 1. Review of Literature

1.1 General Introduction

This chapter deals firstly, with some general issues regarding the relevance of the magnocellular system and then locates implicit cognition withing the theoretical context of social cognition. Relating consciousness and phenomenal awareness to the brain is a significant challenge and some of the general philosophical perspectives and issues are discussed briefly. This is followed by a general introduction to the magnocellular and parvocellular systems and some of the research on implicit cognitions, such as the attentional blink. It became important to review the neurophysiology of visual perception in some depth, so as to develop a rationale for the methodological challenges that were encountered. A general framework for implicit cognition was sketched, and as part of this sketch, the Implicit Association Test (IAT) is briefly described. It is described and critiqued in more depth in a later section – Section 4.4 and some critical commentary is offered.

The title of this dissertation suggests that the magnocellular system is linked with implicit cognition. The rationale of examining the role of the magnocellular (M) system in implicit cognition arises from literature which suggests that it plays a possible role in mediating a distinctive mode of cognition and response. The fact that it appears to be associated with a number of psychiatric (schizophrenia – see Kohler, Tuner, Brensinger, Siegel, Kanes, Gur & Gur, 2003; Laycock, Crewthera & Crewthera, 2007) and perceptual/cognitive deficits such as dyslexia (Livingstone, Rosen, Drislane, & Galaburda, 1991) has made it seem, for some time, to be a promising research area. Some fairly recent research on schizophrenia and dyslexia, focuses on the role of the M system in driving attention (Laycock, *et al.*, 2007*).*

The idea that is explored in this research has to do with visual and cognitive processing. It concerns the contribution of different visual systems to perception, cognition and response. The focus of the investigation is on the magnocellular system which seems to be associated with fast visual categorisation (Fabre-Thorpe, 2011) and thought to mediate rapid, relatively automatic or ‘reflexive’ (Satpute & Lieberman, 2006) responses critically important to survival. This fast-response system is contrasted with perceptual and cognitive processing thought to be associated with the parvocellular (P) visual system. The P system carries much more detailed information, including colour and is slower. Since the amount of information available from this channel is greater and the processing is slower (see Fabre-Thorpe, 2011),

it may mediate more 'reflective' (Satpute & Lieberman, 2006) cognitive appraisals and more measured responses.

Sensory information ultimately provides a basis for action. This is evident even in neonates who utilise sensory information in sensorimotor schemas (Piaget, 1947, 1953, 1967). There is an important distinction between reflexive responses like smooth pursuit eye movements, based on features extracted from a rapid, coarse analysis of visual information and those which are based on much more fine-grained information. Visual motion analysis informs the control of motor behaviour (Spering & Gegenfurtner, 2008) and processing of motion information does not require conscious effort. However, brain systems that are associated with the perception of visual information may modulate motor behaviour. Inhibitory control may be exerted when, for example, there are multiple visual targets and responses to irrelevant ones have to be suppressed (*ibid.*). Thus, it seems clear that there are different kinds of information processed in the visual system and different levels at which this is analysed and integrated.

The general line of argument in this review is that behaviour needs to be guided by information from the environment. This places a burden on an organism to organise and transform sensory data into information that can provide a basis for action (Holyoak, in Wilson & Keil, 2001). However, the world yields a vast amount of information and this creates a tendency towards overload. It is therefore important to be able to selectively filter the constant barrage of information so that sufficient cognitive resources can be allocated to complex problem solving, while the environment is adequately monitored. Complex problem solving makes potentially heavy demands on cognitive resources as one reflects on and integrates multiple, sometimes nested variables and attempts to foresee the results of different responses and strategies. It is crucial however, that while this kind of intense cognitive activity occurs, vigilance to events in the environment is maintained so that one can shift attention abruptly when necessary.

While many tasks require synthesis of different kinds of information, leisurely reflection and complex planning, it is necessary that a considerable amount of fast sensory processing (auditory, olfactory, tactile, or visual) and near-immediate behavioural output occurs. For example, one may be deep in conversation with a colleague while driving back from a

meeting. At times, a deep focus on the conversation would diminish one's awareness of the road, but this does not mean that one cannot steer straight, change gear, monitor other drivers, listen to the engine speed and brake suddenly when the traffic slows or a pedestrian steps off the sidewalk in front of the car. Some of these behaviours are done without much cognitive load and as they are over-learned, with a high degree of automaticity. One may reach a destination with little memory of an uneventful journey while being mindful of the conversation that took place. This relative dissociation of conscious thoughts from more automatic, over-learned behaviours is partly a product of how attention is allocated (Starn, 2007).

Some behaviours, such as reflexes, occur without any central cognitive processing. The withdrawal reflex is triggered by signals from pain receptors in the peripheral nervous system which conduct signals via nerve fibres which enter the dorsal spinal roots and the muscular contraction which results in the withdrawal of the limb is mediated by fibres which exit the ventral spinal roots. It is a simple, but elegant mechanism which does not involve cerebral processing and which occurs before there is any awareness of pain.

However, other kinds of actions do need rapid sensory and cognitive processing. This is especially true when they occur in the social realm. Social interactions occur in real time and are extremely complex. There are social signals like facial expressions and body language which frame the more overt content of communication and these need to be processed rapidly and efficiently. For example, there is value in being able to perceive a fleeting hostile glance since it carries important information which may be critically important for interpreting and framing the actual verbal content of an interaction. Ashwin, Wheelwright & Baron-Cohen (2005) suggest that '...facial threat is a particularly potent social signal.' (p. 2). It tends to recruit attention rapidly (Vuilleumier & Schwartz, 2001, in Ashwin, *et al.,* 2005) and non-verbal danger cues tend to produce rapid affective arousal and preparation for a fight or flight response. The ability to extract and utilise this information to prime behaviour can be critically important.

1.1.1 Social Intelligence, empathy and consciousness

It has been suggested that social animals have an innate ability to imitate the behaviours and mirror the emotions of others. Cacioppo & Decety (2011) suggest that mirroring is an aspect

of empathy, and this theme was explored by Trevarthen & Reddy (2006) in a reflection on research findings on consciousness in infants. They write that ‘An infant shows consciousness by moving in interested, selectively attentive and well-coordinated volitional ways, showing emotions about what happens.’ (p. 4). A starting point for a discussion about consciousness may therefore be in the idea of sociability and the precursors of consciousness may be derived from, or expressed as imitation. Although consciousness is inherently mysterious, its value seems to become clearer when one understand its development.

Meltzoff & Moore’s (1977) work implies that imitation is an innate capacity and is vital for adaptive social functioning. The necessary condition for imitation may be found in strong, pre-verbal and possibly innate links between visual and motor systems so that what an infant sees can be mapped onto their own motor system using proprioceptive, but not visual feedback. The context for imitation is not neutral, however and neonates thrive in a relationship where mutually positive affect is expressed (see Trevarthen & Reddy, 2006 for a discussion on early infant / care-giver interactions).

Early primary visual-motor (sensorimotor) integration is supplemented or largely displaced in the course of cognitive development by more symbolic modes of cognition (for example, in Piaget’s pre-operational, concrete operational, and formal operational stages - see Inhelder & Piaget, 1953).

There are no simple explanations or reductions available to explain how phenomenal consciousness is linked to the underlying neurobiological substrate of the brain, but it is useful to consider what the overall purpose of consciousness might be. In a general sense, the overall influence of the cerebrum is inhibitory (Lezak, Howieson & Loring, 2004) and inhibition allows for the possibility of thinking about what to do and how to do it, before doing it. The fact that after severe brain injury, primitive reflexes which disappear with cortical development, may re-appear (Lezak, *et al.*, 2004) underscores the general idea that the cerebrum tends to have an inhibitory influence on motor systems. Conscious awareness may also provide the opportunity to think about and suppress emotions rather than reveal them. The ‘fear grin’ in chimpanzees (Kirkpatrick, 2007) which they attempt to conceal, may be an instance of this. Self-presentation is important in social situations and this may vary between different social settings.

1.1.2 Emotion and cognition

The influence of emotion on reasoning and behaviour is an important aspect which was highlighted by Damasio (1994). The amygdala, and limbic system have been identified as important aspects of this discussion (Anderson, Bechara, Damasio, Tranel & Damasio, 1999). According to these authors, emotion biases reasoning in useful ways:

> Emotionally related knowledge is presumed to bias the reasoning process covertly, namely, by enhancing attention and working memory related to options for action and future consequences of choices, as well as to bias the process overtly, by qualifying options for action or outcomes of actions in emotional terms. (p. 1035).

The limbic system may be activated at an early stage of perception by emotionally significant cues and there appear to be sub-cortical pathways which are activated before any conscious percept occurs (see Davidson & Irwin, 1999 for a brief review). Pre-conscious activation and attentional capture may be derived from relatively coarse sensory information processed sub-cortically. 'Survival circuits' are thought to be activated by biologically relevant stimuli (LeDoux, 2012), probably involving the lateral amygdala (*ibid.*) to potentiate responses with adaptive outcomes. Low level appraisals of this kind may be associated with behaviours that are relatively stereotyped, performed with a high degree of automaticity and are inherently difficult to inhibit. It is the disjunction between 'automatic' and 'controlled' processing and the visual pathways that may be associated with these processing styles that concern this research.

1.1.3 Perception, social cognition and consciousness

It has been suggested that there are survival circuits based on early pattern recognition, which when activated, help to organize behaviours in specific ways (LeDoux, 2012). LeDoux writes that

> In sum, survival circuits are sensory-motor integrative devices that serve specific adaptive purposes. They are tuned to detect information relevant to particular kinds of environmental challenges and opportunities, and they use this information to control behavioral responses and internal physiological adjustments that help bring closure to the situation. (p. 655).

Strategic and measured responses are extremely important as the dynamic and rapidly shifting interpersonal social landscapes in which a social animal, primate, or human exists, requires frequent adjustments to longer term plans or strategies. But social alignments, coalitions and estimates of social rank shift frequently and abruptly, especially when a social hierarchy or political situation is unstable and there is a constant need to monitor and track

others' mood states, predict intentions and prepare for action (Burish, Yuan & Wang, 2004). Miscalculation of social rank or intention on the part of another social player could be disastrous and possible clues about social threats may be monitored reflexively, such as gaze monitoring (Haxby, Hoffman & Gobbini, 2000). Survival priorities include attachment to a parent or coalition partner, feeding, reproduction, avoidance of predators, maintaining status in the group and attending to the needs and safety of offspring (p. 579). As Worden (1996) puts it,

> Each one of these goals involves complex coordinated patterns of behaviour, and can be studied as a behavioral system (Hinde 1982). At any one time, an animal is involved in typically one, or at most two or three behavioural systems. In higher animals, a behavioral system involves not just stereotyped reflexes, but also goal-directed behaviour. To achieve the goals of any behavioral system, complex locomotor problems, problems of navigation, and social problems may need to be solved. For instance, in order to feed, a primate might have to navigate to a food source, negotiate social obstacles in the form of dominant peers, and then climb a tree to pick fruit.' (p. 579).

These are external, environmental priorities that significantly inform behaviour and a great deal of information is derived from vision. Part of the value of vision is that it allows for appraisal of a situation at a comparatively comfortable distance. Some visual appraisals are done in a leisurely manner, whilst others require fast, less conscious and more reflexive responding – like dodging an attack from a predator, ducking to avoid a large, fast moving object and making fast directional changes when pursuing prey. Kveraga, Ghuman & Bar (2007) argue that while 'A primary function of the brain is to predict its environment...' (p. 145), they also acknowledge that some visual analysis and response has to be accomplished very rapidly, such as in the case of a baseball player, facing a pitched ball (*ibid*.). Their argument about how the brain solves the considerable problem of being able to react rapidly to fast events, but develop a fine-grained picture of the world is that there are visual systems which work in parallel:

> The complete understanding of how the brain solves this problem so efficiently remains in many respects elusive. We propose that this efficiency of vision arises in part from the integration of two interacting and cooperating mechanisms - a fast, coarse subsystem that initiates top-down predictions based on partially processed visual input, and a slower, finer subsystem that is guided by the rapidly activated predictions and refines them based on slower-arriving detailed information. (p. 145).

Indeed, some visual information processing is very fast. Born & Pack (2002) point out that motion encoding responses in the MT (Middle Temporal area) begin at about 80 ms after the onset of target motion. There are even earlier signals associated with target eccentricity in terms of the relative angular shift in the field of vision and these are seen as early as ≤ 40 ms.

It is fairly clear, according to Spering & Gegenfurtner (2008), that motion information enters the primary visual cortex '...via the magnocellular layers of the lateral geniculate nucleus...' (p. 77). Neurons in MT have large receptive fields, possibly consistent with the known characteristics of the M system. There seems to be a high degree of integration between sensory (visual) input and behavioural response (see discussion by Lisberger, Morris & Tychsen, 2001 of visual motor and sensory motor integration, also a discussion by Laycock, Crewthera & Crewthera, 2007). This integration may be at a reflexive, or pre-cognitive stage and though it may be associated with a conscious intentions to act, the presumably causal relationship between the intention to act and the actual response is not as simple as common sense might suggest. Libet's (1983) work complicated the notion that conscious intentions cause actions, by showing that 'readiness potentials' preceded 'voluntary' decisions to respond ('W judgements') by several hundred milliseconds (in Haggard, 2005). In other words, although it would seem logical that signals associated with a voluntary intention should precede 'readiness potentials' generated in the pre-motor cortex, this proved not to be the case. This finding complicates the idea of simple, temporally linear volitional control (see Haggard, 2005).

1.2 The brain, behavior and consciousness

The idea that a psychological problem may be investigated by exploring functional aspects of a neurological system is attractive, but in reality it is difficult to map the relationship between brain structure and function. Lezak, Howieson, & Loring (2004) caution that 'The relationship between brain and behavior is exceedingly intricate and frequently puzzling' (p. 40). Even the description and identification of a brain *system* posited to be involved in aspects of visual processing is tentative because of the sheer complexity, and interconnectedness of brain systems. To take an example from visual perception: there is evidence that an object's features (like colour, shape and motion) are represented in different areas of the visual cortex, and this means that perception is not a simple, linear or deterministic process, but that it involves the interaction of spatially separate areas of the visual cortex.

Compounding the complexity is the fact that higher order processes such as attention, and probably other brain, and somatic states like motivation, arousal and mood regulate what information is represented in consciousness in accordance with transient priorities. Crick & Koch, (2003) suggest that at the level of the neural network, there is competition between

‘transient coalitions of neurons’ to attain sufficient levels of activation to stand out against the background noise of other competing coalitions, and cross a threshold of some kind and enter consciousness. This sounds like a political strategy of competing for public attention by recruiting others to a cause, in order to generate enough interest to enjoy the limelight for a few moments, and this is exactly the analogy that Crick and Koch use. The attentional spotlight, in their theory, may have a role in modulating activation. Attentional selection may be based on saliency (how important is the information) and this could be considered a kind of bottom-up attentional system – recruiting attention because of some relevant feature. Volition (deciding to attend to, or ignore something) is essentially a kind of type of top-down control exerted by the executive system (see Crick & Koch, 2003, p. 122, and Mutalik, 2003 for further explanation and Hopfinger, Buonocore, & Mangun, 2000 regarding voluntary attentional control). It seems clear that the activation of a primary sensory area is a necessary, but not sufficient condition for a conscious percept to be produced (see Dehaene, Changeux, Naccache, Sackur & Sergent, 2006), although saliency (seeing a venomous snake in the path for example) would and indeed should, hijack a behavioural plan to walk there. It is these, and other complexities which make it difficult to predict behavioural responses to visual stimuli.

1.2.1 Consciousness and ontology

It has been argued that responses may occur without conscious awareness and even when large parts of the brain are destroyed, primitive reflexes may persist. Neural events at these different levels are seen as relatively independent – one may consciously inhibit a reflex, but one cannot usually exert complete control. For example, one may stifle a cough, or inhibit an involuntary withdrawal reflex after the fact, to limit the social impact, or avoid embarrassment. Conversely, reflexive responses do not require conscious effort, or even any cognitive processing, since they are mediated by structures in the spinal cord, brainstem, or midbrain and consciousness usually follows after the fact.

The Cartesian notion of ‘mind’ has been offered as an explanation for why a thought, feeling, or idea is private and accessible only to the ‘self’, while a reflex is mechanistic - or even that it functions ‘hydraulically’. The qualitative distinction between 'private' cognition (*res cogitans*) and ‘objective’ neuromuscular linkages or reflexes, which are characterised as ‘mechanical’ (*res extensa*) is unsatisfactory when one considers that the two kinds of

phenomena are mediated by a common neurological substrate. The private, non-material Cartesian 'self' which Descartes proposes in the *Cogito* seems to a version of 'hard problem' of consciousness (Chalmers, 1995) and as such, would be difficult to tackle scientifically.

While the subjective phenomenology of the 'self' seems indisputably real, yet inaccessible to direct observation or objective measurement, this does not mean that the mystery is that of ontology, or metaphysics. Strawson (1959) argued that it is possible to speak about and ascribe states of consciousness to persons, without resorting to 'Cartesianism'. Strawson suggested that the apparent dualism is a feature of language and does not imply dualism in terms of ontology. He coined the term 'P-predicate' as a way of describing objective, physical attributes of a person or entity. Strawson suggested that one may ascribe such predicates on the strength of observation: 'I mean such things as "going for a walk", "coiling a rope", "playing ball", "writing a letter"...' (p. 110). Conversely, M-predicates refer to experiences or sensations (perhaps *qualia* would be somewhat equivalent) which are not observable, but are descriptors from the first-person perspective, such as 'I am in pain'. An inference can be made from the second/third person perspective that 's/he is in pain' and is an instance of what Strawson termed a 'P-predicate'. Its validity rests on the degree to which it is based on observable pain behaviour. Strawson's position seems to allow for inference about possible states of mind from experimental data, without involving metaphysics or difficult arguments about ontology.

1.2.2 Comments on the neurobiology of consciousness

I have introduced the problem of dualism with the suggestion that it is a problem that arises from linguistic ambiguity, rather than a problem of ontology. Another problem arises when explanations of consciousness involve some version of the homunculus theory. The 'hard problem' of consciousness (Chalmers, 1995) is not easily resolved. In short, either one has to accept that there is an 'explanatory gap' (Levine, 1983) and leave it at that, or pursue the question by trying to determine what the homunculus consists of. There have been suggestions about what parts of the brain are necessary for consciousness, such as the intralaminar nuclei of the thalamus (Purpura & Schiff, 1997) and there is much debate about what areas (thalamocortical loops/pyramidal neurons) or dynamic states are associated with conscious percepts (see Llinás, Ribary, Contreras & Pedroarena, 1998 for a review of the role of thalamocortical systems in consciousness).

Interestingly, Crick & Koch (2003) reject the social-cognitive version of the 'self' in favour of just such a notion, and they explicitly suggest the existence of two modes of thought: one of them associated with a 'conscious system', and the other, a 'Zombie mode'. This is partly a deliberate play on the naïvety of the homunculus idea, but they offer a simplified model, or framework for understanding consciousness. They begin with the suggestion that 'A good way to begin to consider the overall behaviour of the cerebral cortex is to imagine that the front of the brain is "looking at" the sensory systems, most of which are at the back of the brain' (p. 120). While this explanation seems to be of the kind generally rejected as unsatisfactory, the various 'entities' and modes can be plausibly mapped onto neural systems as suggested earlier and the key elements of their 'Framework' do not covertly involve metaphysics, or a naïve homunculus notion.

Since the 'hard' problem of consciousness is so difficult to address, researchers tend to be more interested in the theoretical viability of the idea that 'neural correlates of consciousness' (NCC) exist. It may be that consciousness is a property of many neurons working synchronously, perhaps with the 40 Hz gamma oscillation being the signature of conscious states (Engel, Fries, Konig, Brecht, Singer, 1999; Dehaene, *et al.,* 2006). Many researchers, such as Baars (1988) believe that consciousness is distributed and is not the property of a small class of neurons, or cells.

The integration of different aspects of phenomenal experience which James (1890) seemed to be referring to, is known as the 'binding' (Tononi & Edelman, 1998; Zeki, 2003) of various aspects of sensory experience (e.g. movement, and colour of a visual object) into a unified percept. The idea of temporal binding was first suggested by Singer (1993). Temporal binding refers to the synchronous, correlated activity of spatially segregated populations of neurons which seem to be associated with feature processing.

Although the framework suggested by Crick & Koch has been criticised as speculative, even 'woolly' (Mutalik, 2003), it contributes some useful ideas:

1. The idea that the anterior part of the brain deals with more abstract and integrated information as it 'looks at' the back of the brain, which is unconscious and performs more

stereotyped information processing. Crick and Koch characterise the back part of the brain as working in a ‘zombie mode’, in that it is not ‘conscious’ of itself, but simply processes sensory information and makes it available to the front part. An illustration of a 'zombie', or unconscious integration of visual information is the dorsal visual stream which contributes to movement, but does not lead to a conscious percept (See Ungerleider & Mishkin in Ingle, Goodale & Mansfield, (Eds.), 2002).

2. The ‘zombie’, or primary, secondary, and tertiary sensory areas deal with high volumes of information and some of their outputs are like ‘cortical reflexes’, which deal with eye fixations, object tracking and auditory reflexes such as orienting. Their information processing mode involves structured, and stereotyped information processing and although the transient sensory information in these circuits may become available to consciousness, it is not always, or necessarily represented since it is gated by top-down, attentional control. In short, this information processing occurs without effort, and one is not conscious of the process of perception, but only of the product. The frontal areas of the brain deal with more abstract and general information, and allow for a different mode of thought which is more integrative and reflective, allowing for planning, evaluation, self-regulation and awareness.

1.2.3 The nature and function of consciousness

Since the topic of implicit cognition is an important focus, I would like to outline some of the theoretical trends and research which inform directly, or more generally, the current status of this concept. It has been suggested that the value of sensory information is to guide behaviour, whether or not this involves conscious cognitive control. This is a pragmatic definition which links sensory input to action. I will not attempt to engage with the controversial aspects of consciousness (e.g. the problem posed by Nagel, 1974 about whether worms or gnats possess consciousness). The conclusion by Starn (2007) that ‘In summary, nature testifies that animals have consciousness...’ may be theoretically viable, but this an open question. Sociability may involve consciousness and that this may provide a fundamental link between human and non-human consciousness.

Crick & Koch (1995) suggest that the ‘...function of visual awareness is to produce the best current interpretation of the visual scene, in the light of past experience either of ourselves, or

of our ancestors (embodied in our genes), and to make it available, for a sufficient time, to the parts of the brain that contemplate, plan, and execute voluntary motor outputs (of one sort or another)' (1995, p. 121). Essentially, the function of the brain in any species is to utilise sensory information in such a way that it usefully informs behavioural 'output'. For example, most species rely on visual information for foraging, hunting, navigation and escaping predators (see Corbetta, Patel, Shulman, 2008). The visual system provides information which has a direct and often immediate influence on movement, especially via subcortical circuits which mediate, for example, eye movements. This signal processing is unlikely to be associated with awareness (Crick & Koch, 1995). In humans, primates and other social animals it is likely that visual information like faces, facial expressions and body language are sufficiently important to modulate attention (in a bottom-up manner) and gaze following, tracking others' mood states, location, rank and relationship to the self may be automatic (see Gómez, in Whiten, 1991 for a discussion of gaze monitoring in primates). Focused consideration that informs and guides complex behavioural programs could probably be considered 'conscious' (what is often called 'mind reading' - see Byrne & Whiten's (1988) discussion, in Whiten, 1991 of tactical deception in primates). Tactical deception seems to provide some evidence for consciousness in non-humans, in that it requires them to think about, and predict what another individual's intentions are and take this into account in their (deceptive) actions.

It is difficult to draw firm conclusions about whether non-human animals possess 'consciousness' especially since there is little consensus on the definition of consciousness. The concept of 'theory of mind' is frequently used, perhaps as a proxy for 'consciousness', and perhaps 'thinking about what the self, or others are thinking', is necessarily a conscious activity. Agnew, Bhakoo & Puri (2007) suggest that 'Theory of mind refers to two concepts: the knowledge that other animals have mental states which may differ from our own; and the ability to infer what these internal states may be.' (p. 288). Social cognition in non-human animals seems to require perspective-taking, and making inferences about the intentions of others (Whiten & Perner in Whiten, 1991), in order to devise strategies that take others' intentions into account, and which aim to circumvent them. There is a considerable and growing literature based on observations of tactical deception, and counter-deception in chimpanzees (Byrne & Whiten, in Whiten, 1991), and other non-human animals. The relevance of social cognition and social variables informed Sapolsky's work (e.g. Uno, Tarara, Else, Suleman & Sapolsky, 1989) on the neurobiology of stress.

The idea that consciousness may serve a fundamentally social function is intriguing. Humphrey (1976), wrestled with various definitions of human intelligence, but was struck with the fact that they had little relevance to non-human animals. Humphrey argued that the problem with these definitions was the lack of a link between intelligence and biological fitness (p. 303). The answer to the puzzle of why some animals are so intelligent could not be accounted for by their technical ability (tool making and usage), which is not reliably associated with encephalisation. The answer, Humphrey argued, was to be found in the social environment in which they lived, and the need for individuals to be 'calculating beings', guessing the intentions of other individuals, and shrewdly calculating how to manipulate situations for their personal benefit. This ability has a direct bearing on social rank, and ultimately biological fitness.

Conventional scientific wisdom has tended to the position that non-human animals do not share any demonstrable attributes of human consciousness, and that the burden of proof is on those who disagree. Moral and ethical reasoning has not been overly concerned with questions about sentience, or consciousness in primates. Ironically, the anatomical similarity between human and monkey brains has been convenient for researchers and there is a considerable literature on single cell recordings and ablation, using live monkeys as subjects. Much of our knowledge about the visual system is based on animal research.

The suggestion of 'mirror neurons' in monkeys by Rizzolatti, Fadiga, Gallese & Fogassi (1996), seemed to provide support for what many ethologists, and comparative psychologists had theorised about – namely theory of mind (Premack & Woodruff, 1978 in Whiten, 1991). The focus on sociability from comparative psychology provides an important context for the finding of Rizzolatti *et al.,* and illustrated the relevance of social neuroscience which seeks to '...investigate the biological mechanisms that underlie these social structures, processes, and behavior and the influences between social and neural structures and processes.' (Cacioppo & Decety, 2011, p. 162). Rizzolatti, *et al.* recorded activity from the ventral premotor cortex of monkeys, and observed that the same cells fired when the monkey did the specified action, as when the monkey observed another monkey doing that action.

Rizzolatti remarks:

> ...It might sound bizarre that in order to recognize an action, one should activate the motor system. As a matter of fact, this is not so strange. A mere visual perception, without

> involvement of the motor system would only provide a description of the visible aspects of the movements of the agent. It would not give, however, information on the intrinsic components of the observed action... (2005, p. 419).

Although imitation may seem to be a mere curiosity or extravagance of 'nature', research in the last few decades suggests otherwise. Imitation seems to be an important, early capacity which human neonates possess. Its implications are philosophically and theoretically interesting.

When Meltzoff & Moore (1977) produced evidence of neonatal imitation, their finding sparked a major debate. Their evidence of neonatal imitation is interesting because it conflicts with the Piagetian notion that newborn babies are solipsistic, '...experiencing their own internal sensations and seeing the movements of others without linking the two...' (Piaget 1962, in Wilson & Keil. 2001, p. 389). Meltzoff & Moore's explanation involves the idea of 'active intermodal mapping', by which the infants map visual information onto their own motor systems and output (p. 961). They suggest therefore, that 'A common representational code may unite the perception and production of basic human acts.' (p. 954). Obviously, this 'representational code' is not linguistic, although it might be related to some innate neural architecture that is related to both language and behaviour.

While the findings of Meltzoff & Moore (1977) have been replicated, there is still debate between theorists about methodological issues and possible interpretations of the psychological processes which underlie imitation. Their idea of intermodal mapping fits with the idea of mirror neurons, and the possibility that there are 'early perception-production links' which could be innate, and are perhaps an important link for social animals that leads to social comprehension and empathy.

The general point of this discussion is not to argue that humans and non-humans share fundamental similarities, but to point out the relevance of social cognition to consciousness. In many ways, humans and non-humans face some of the same challenges and the need for consciousness seems to be illustrated in the need to make comparisons between self and other and formulate strategies which take this into account. In a human situation, one does have to consciously consider the impact of one’s own behaviors, dispositions, alliances and perceived attitudes on others. Although many behaviours are not intended to deceive, the fact that they frequently vary with social context means that there is at least an element of seeking

approval, or wanting to conform to norms and expectations. Some social behaviours may be conscious, such as when attitudes about sensitive social issues are expressed. Social desirability may motivate an individual to suppress behaviours in one context but not in another. Suppressing the expression of social prejudice and creating an impression of egalitarianism probably takes conscious effort if one privately holds views that are prejudiced. Speaking about social issues amongst like-minded peers does not require such a degree of conscious effort.

The fact that individuals need to attend to self-presentation in any given social context does not mean that they are basically dishonest or deceptive. It illustrates the fact that in almost any situation there is competition and conflict which has to be managed smoothly. This probably amounts to social sensitivity or competence (thinking about how others will feel and behave if one does or says something). It takes a conscious effort to assess the attitudes of others and adjust one's own behaviours to conform to a perceived social norm. In the company of vegetarians, for example, a non-vegetarian might not offer cooking tips about meat dishes, for fear of causing offence, or being socially embarrassed.

It seems that social cognition is an important area in which consciousness is expressed, although many behaviours in a social context are executed without conscious awareness. A *faux pas* usually occurs precisely because one is not aware of a social norm or what others will think about some behavior. It would be difficult to function if everything one did required conscious consideration, so presumably this is why many behaviours occur spontaneously and without conscious thought.

1.2.4 Language, action, cognition, and emotion

While there seem to be plausible links between perception and action, the manner in which language and action are related at the neurological or functional level is a point of debate. From the embodied cognition perspective, there have been claims of an intimate link between word meanings and sensory-motor systems (e.g. Hauk, Johnsrude & Pulvermuller (2004); Bedny, Caramazza, Grossman, Pascual-Leone & Saxe (2008); Bedny & Caramazza (2011); Bedny, Caramazza, Pascual-Leone, Saxe (2012). Although they acknowledge the evidence that '...action-verb comprehension and motion perception interact...' Bedny *et al.* do not accept that there is sufficient support to conclude that these links are neurological, innate, or

structural. However, it is possible that visual consciousness has a qualitatively different significance for language speaking humans and that it might be subordinate to other layers of abstract, language-based representations, and conceptual structures. Bedny *et al.* (2012) found that left middle temporal gyrus activation was similar in sighted and blind participants, suggesting that the neural representation of the action component of verbs is not uniquely linked with visual percepts.

Lieberman, Hariri, Jarcho, Eisenberger, & Bookheimer (2005) investigated the effects of perceptual vs. verbal encoding on amygdala activation in a task where participants viewed both African American and Caucasian American faces. Amygdala activation for the perceptual processing of African American faces was greater than that for Caucasian American targets for all participants (both African American and Caucasian American). Interestingly, their findings suggest that verbal encoding (using text instead of faces) of African-American faces attenuated amygdala activation in comparison with perceptual encoding. This suggests that verbal processing which occurs in another cognitive system, inhibits affective arousal. While the common pathway for both types of information in this instance is the visual system (seeing text, vs. seeing faces), it is unlikely that the amygdala has any major role in decoding verbal information (presented as text). However, it does seem clear that it has an important role in subcortical circuits which are involved in early processing of facial metrics (referred to, and processed as low spatial frequencies) and affect (a finding confirmed by authors, such as Holmes, Winston, & Eimer, 2005; Vuilleumier, Armony, Driver & Dolan, 2003; Vuilleumier, 2005; Jetha, Zheng, Schmidt, & Segalowitz, 2012), although in the research of Lieberman *et al.*, the task conditions did make different demands in terms of attention and processing (in the perceptual encoding task, participants had to choose the face at the bottom of the screen which matched the race of the centre target face, and in the verbal encoding trials, they had to choose the corresponding race label which matched the centre target face).

The idea that the amygdala is a key structure in fear conditioning has been well documented (Phelps & LeDoux, 2005) and evidence seems to be available that there are subcortical projections to the amygdala which lead to its activation before awareness occurs. According to Phelps & LeDoux, even a subliminally presented CS, '...one presented so quickly that subjects are unaware of its presentation leads to coactivation between the amygdala and both

the superior colliculus and pulvinar which was not apparent for a CS presented supraliminally.' (Morris et al., 1998a, in Phelps & LeDoux, 2005).

The research of Lieberman *et al.* seems to demonstrate the effects of different modes of encoding and the possibility that verbal encoding may inhibit affective arousal, possibly because it activates processes thought to be involved in 'controlled processing' which are linked with linguistic and semantic categorisation. It may be a function of the extra cognitive processing which disrupts affective arousal, or perhaps it activates a 'non-affective controlled processing' mode (*ibid.,* p. 3). However more recently, their research into the mechanisms of 'affect labeling' suggests that attenuation of activity in the amygdala is associated with activity in the right ventrolateral prefrontal and medial prefrontal cortex (Lieberman, Eisenberger, Crockett, Tom, Pfeifer & Way, 2007). They suggest that the prefrontal activity is inhibitory and its '...activity during affect labeling may be involved in disrupting the amygdala's response to emotionally evocative images.' (p. 2). Essentially, these two brain regions which show inversely correlated activity with the amygdala, may be instrumental in dampening affective arousal. Statistically, the medial prefrontal area appeared to mediate the relationship between the two structures. The interplay between affective, perceptual and cognitive processes is potentially an interesting aspect of this research, and it illustrates both the nuances of, and possible disjunctions between brain systems.

1.2.5 Implicit perception and cognition

I have mentioned instances of behaviour which occur without consciousness. Some of these are trivial, e.g. reflexive arcs mediated by spinal ganglia and others are more complex, like visual orienting and tracking reflexes. While these do occur without conscious effort or volition, so do many other processes, such as blood pressure regulation and respiration. In a sense they are complex, but they are not the kind of implicit information processing I am referring to.

One of the most dramatic demonstrations of a dissociation between consciousness and visual/motor systems is provided by Weiskrantz (1986) who documented the phenomenon of blindsight, whereby a patient who could not consciously report visual experience, was able to make accurate decisions regarding the position and movement of objects. This phenomenon again received popular attention in 2010 when De Gelder published another case in Scientific

American (De Gelder, 2010). These findings have highlighted the fact that some perceptual and cognitive processes inform behaviour without accompanying awareness.

Greenwald (1992), suggested that the new interest that cognitive researchers gained in implicit cognition, as this area was salvaged from the disinterest of behaviourists, and the exotic ideas of psychoanalytic researchers, grew into the New Look 2, in about 1974, as a credible scientific project. Cognitive researchers began to seriously examine unconscious influences, or processes in various forms, such as Implicit Memory (e.g. Graf & Schacter, 1985; Schacter & Graf, 1986a), Automaticity (Shiffrin & Schneider, 1977), semantic priming (Greenwald, Schuh, Klinger, 1995), Implicit Social Cognition (Greenwald & Banaji, 1995).

Underlying these concepts is the notion that there may be antecedents, influences, and motivations for behaviour that are not accessible. For example, Schacter (1987) suggested that 'Implicit memory is revealed when previous experience facilitates performance on a task that does not require conscious or intentional recollection of those experiences…' (p. 501). In about the late 80s and early 90s there was an increased interest in the idea of both perception and cognition which occurs without awareness, but may influence both cognition and behaviour. At this time, Greenwald & Banaji (1995) began to theorise about how attitudes, social behaviour, self-esteem, and stereotypes operate in an unconscious manner and they formulated a working definition of implicit social cognition.

They also argued that there was a degree of overlap between the set of theoretical distinctions which existed at the time, i.e. the notions of 'aware-unaware', 'unconscious-conscious', 'intuitive-analytic', 'direct-indirect', 'procedural-declarative', and 'automatic-controlled' (p. 4). These Washington University researchers were influenced by Greenwald's earlier semantic priming research, which helped to lay some of the ground conceptually for the Implicit Association Test paradigm which Greenwald, McGee & Schwartz published in 1998, and demonstrated at the first international conference of the Association for the Scientific Study of Consciousness in 1997. The IAT research paradigm was revealed at a point when confidence was developing from the new degree of convergence in the subliminal research and the clarity which had begun to emerge from the semantic priming experiments.

1.3 Visual perception theory and research

In this section I will give an overview of the visual system and describe two important circuits which originate in the retina and form the basis of two major visual systems. This review aims to summarise and establish generally accepted knowledge on aspects of the visual system and introduce some related research. I will reflect more critically and in more depth on some of the key ideas and concepts which have a bearing on this research project in section 1.4.

Historically, two main classes of cells in the primate retina are thought to relay information on colour (midget ganglion cells) and luminance (parasol ganglion cells) (Kaplan & Shapley, 1986). These retinal ganglion cells are thought to form the basis of two main visual systems: the parvocellular (small-cell) and magnocellular (large-cell) systems respectively (Livingstone, 1988).

Midget ganglion cells in the retina receive input originating from cone cells concentrated in the fovea, and carry information about colour. The receptive field of a midget ganglion cell is small as its afferent is derived ultimately from a single cone cell. There are 3 basic cone cell photoreceptor types which respond to short, medium, and longer wavelengths (corresponding approximately with blue, green and red respectively) – see Figure 1.

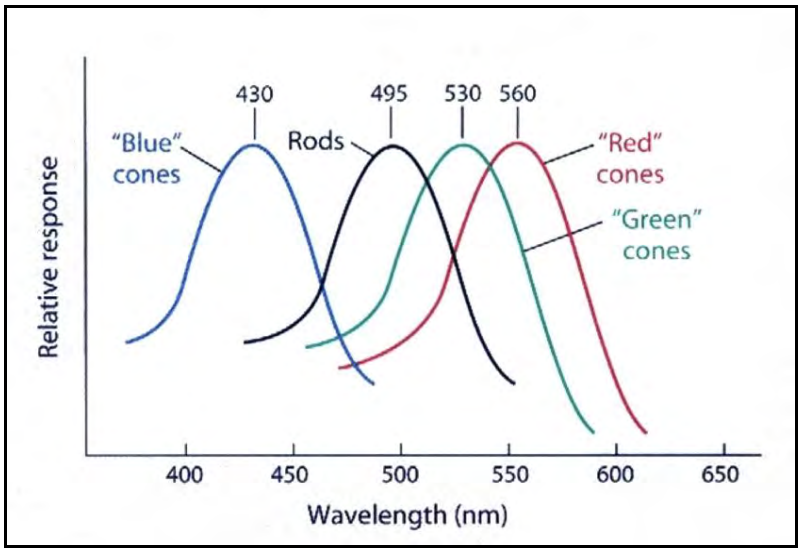


Figure 1. Photoreceptor frequency sensitivity (Gazzaniga, 2005, p. 152).

Figures 2 and 3 depict the receptor and cell types, connections and interconnections between cells in the retina.

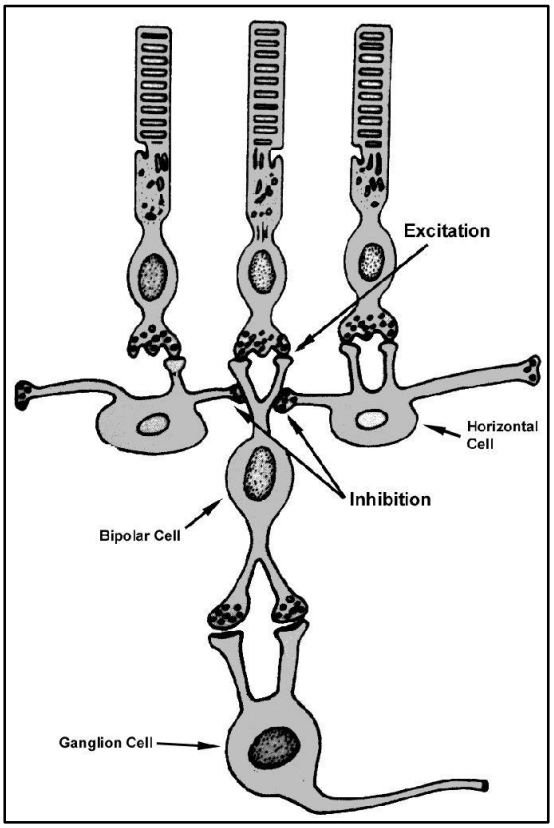


Figure 2. Cell connectivity (from Caceci, 2012).

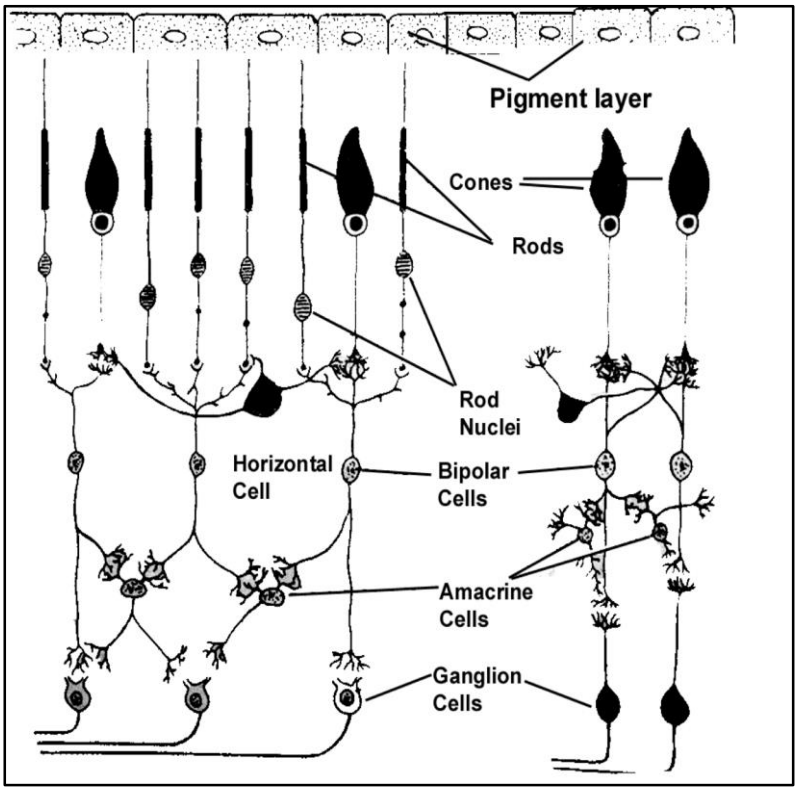


Figure 3. Cellular organisation of the retina (from Caceci, 2012).

In Figure 2 the arrangement of photoreceptor outputs (graded, excitatory), and the inhibitory horizontal cell connections is shown. The photoreceptor outputs, and the integrator neurons, including the horizontal cells are electrotonic (i.e. there is a direct flow of electrical current along the membrane). Bipolar cells may depolarise, or hyperpolarise, depending on their inputs, and their outputs to the ganglion cells determine these cells' rate of firing (action potentials, not electrotonic conduction). Figure 3 depicts the variety of interconnections between different photoreceptor types and their destination ganglion cells. This representation shows lateral connections between different photoreceptor (rod/cone) outputs.

Colour perception depends on the relative colour sensitivity of the cone cells and the overall opponency of red/green and yellow/blue signals received from various bipolar cell types. These (red/green, yellow/blue) colour dimensions are thought to be partly a function of the connections between various bipolar cell types and are present at the level of the midget ganglion cells[1]. To put it simplistically[2] midget ganglion cells receive inputs from bipolar cells which receive various inhibitory presynaptic inputs from horizontal cells and ultimately input from different cone cell types. The output of a bipolar cell is thus a summation of presynaptic cone signals and the inhibitory horizontal cell signal. The product of this arrangement is that midget ganglion cells may be excited by one wavelength (e.g. red), and inhibited by another (e.g. green). The recent finding by Crook, Manookin, Packer & Dacey (2011) suggests that red-green opponency results from a linear, non-selective and excitatory presynaptic horizontal cell signal. This implies that the traditional explanation that red-green antagonism is derived from cone selective circuits is not valid, since cone outputs appear to be summed linearly and are neither influenced by inhibitory (GABAergic), nor glycinergic circuits (see Crook, *et al.*, 2011).

---

[1] Recently, Crook, Manookin, Packer & Dacey (2011) found that red/green opponency is mediated presynaptically by horizontal cells and is not a product of centre/surround inhibition between different cone types.

[2] Retinal signal processing is much more complex than described. There is a considerable amount of preprocessing within the retina itself and numerous types of horizontal cells and bipolar cells. See Chan, Martin, Clunas & Grünert, 2001 for a discussion of the implications of bipolar cell diversity in the primate retina.

Midget ganglion cells receive projections from bipolar, amacrine and horizontal cell axons in the inner plexiform layer of the retina, and the axons of these ganglion cells join others to form a large bundle, known as the optic nerve.

The retinal signal source for parasol ganglion cells is diverse and includes rod cells, diffuse bipolar cells, amacrine cells and probably 'small' bistratified ganglion cells (SBCs), (Dacey & Lee, 1994 - in Martin, 1998). Parasol ganglion cells have large receptive fields as they receive input from bipolar cells which collect information from an average of 10 photoreceptor cells (Boycott and Wässle 1991, in Calkins & Sterling, 2007). Parasol ganglion cells receive direct input, mainly (80%) from amacrine cells (Jacoby, Stafford, Kouyama, & Marshak, 1996), but not *directly* from rod cells (Grünert & Martin, 1991 in *ibid*). Amacrine cells receive input from various bipolar cell types, including rod and diffuse bipolar cells (Chan, Martin, Clunas & Grünert, 2001).

The connectivity of non-midget achromatic cells complicates the idea that the M and P systems function discretely. The question of whether the M system receives exclusively achromatic input also arises. Calkins & Sterling studied the circuitry of achromatic ganglion cells in a macaque retina and described two types, namely parasol and garland cells. When they quantified the inputs of parasol cells, they discovered that about 120 bipolar cells and about 85 amacrine cells made contact with it, carrying signals from approximately 50 photoreceptors. The garland cell received a much higher number of synapses from amacrine cells. The bipolar signal sources for parasol and garland cells were mainly diffuse bipolar (DB) types 3 and 2 respectively. Both these bipolar types were also found to collect information 'indiscriminately' from all cone types (p. 2651), including S-cones. They speculate that both cell types may contribute to the known properties of the magnocellular (M) pathway '...the parasol/brisk-transient... and another with better spatial resolution but slower kinetics (the garland/local edge).' (*ibid.*). Importantly, they also speculate that because the parasol pathway is known to project to MT, this might account for the S-cone signal projections in MT (see Calkins & Sterling, 2007, p. 2651 for this discussion).

The achromatic ganglion cells described by Calkins & Sterling are potentially important to understanding the function of the M system which seems to have been oversimplified as a fast visual system, with good temporal resolution, but poor spatial resolution (e.g. Hubel &

Livingstone, 1990; Livingstone, Rosen, Drislane & Galaburda, 1991; Chase, Ashourzadeh, Kelly, Monfette and Kinsey (2003).

Another interesting consideration arises from the work of Crook, Peterson, Packer, Robinson, Troy & Dacey (2008). It is widely assumed that parasol cells project to the superior colliculus (SC), and while this does appear to be the case, the assumption that this is the *only*, or *most important* projection, may be misleading. If the M system carries achromatic information and if it is the only projection to the SC, it follows that the SC must be colour-blind. These authors confirmed existing evidence of a magnocellular projection to the SC in a macaque brain using retrograde photodynamic staining.

White, Boehnke, Marino, Itti & Munoz (2009) note that '...the neural representation of color is widely believed to be absent in brain areas that control saccadic eye movements' (p. 12159). There is general agreement that colour vision gives primates the ability to discern ripe fruit from foliage. It seems to follow, hypothetically, that colour could play '...an important role in segmenting objects from backgrounds...' which could '...in turn facilitate visual search...' (D'Zmura *et al*., 1997, in White, *et al.*, 2009). White *et al.* suggest that since humans are able to detect and orient rapidly towards stimuli that are chromatically isoluminant with the background, it is likely that some information is available to the saccadic system, beyond the (allegedly null) luminance information propagated via the M system (*ibid.*). Their research appears to show that neurons in the SC of rhesus monkeys do respond to isoluminant stimuli, albeit with low colour specificity. Their isoluminant stimuli were coloured in keeping with the major colour opponencies described earlier (Red/Green).

For a species that is highly reliant on foraging, there would be an advantage if colour, especially red/green discrimination, could recruit attention via subcortical circuits that influence saccades at an early stage. An interesting comment and illustration was made by Jerison (1973) (see Figure 4), that the problem of discriminating figure from background in a forest or landscape requires extensive visual processing and Jerison speculates that this may have contributed to encephalisation in primitive birds. This seems to fit with the argument of White, *et al.*, 2009.

While White, *et al.* acknowledge the critical possibility that their isoluminant stimulus calibration may not have been perfect, and argue 'We would nonetheless expect the effect of residual luminance to be relatively small...' (p. 12162), this may be a flaw which undermines this study to some extent, since it is practically impossible to verify that the monkeys experienced the colours as isoluminant. Their claim that these results produce 'strong evidence' that the primate SC receives information from a pathway other than the M system, may be overstated. However, they saw vigorous responses to the chromatic stimuli, which were even stronger than responses to the maximum luminance stimuli. They also found that visual latencies were longer by about 30-35 ms in the colour condition, than in the contrast condition.

Figure 4. 'The forest is a visual nightmare' (Jerison, 1973).

Another interesting aspect to this discussion emerges from recent research by Otero-Millan, Macknik, & Martinez-Conde, 2012). These authors examined the well-known 'Rotating Snakes' illusion (Figure 5), and found support for their hypothesis that the motion illusion is triggered by microsaccades. The Rotating Snake illusion is derived from luminance gradients which are thought to produce a motion signal. Since eye movements are considered to be controlled by mid-brain nuclei, probably in conjunction with cerebellar processing (Lezak, *et al*, p. 44) the finding that microsaccades are correlated with the onset of the subjective motion illusion appears to fit with traditional thinking that eye fixations and saccades are derived from the M system (direct projections, or as part of a 'bottom-up' attention system – see Omtzigt, Hendriks & Kolk, 2002; Kveraga, Ghuman & Bar, 2007; Hirai, Saunders & Troje, 2011). Whether or not M system deficits are directly associated with poor ocular motor control is not clear (see Stein, 2001), but there is clearly a tight relationship between attention and eye movements (Omtzigt, *et al*, 2002) which may be bottom up (recruiting attention), or top down (intentionally directing gaze).

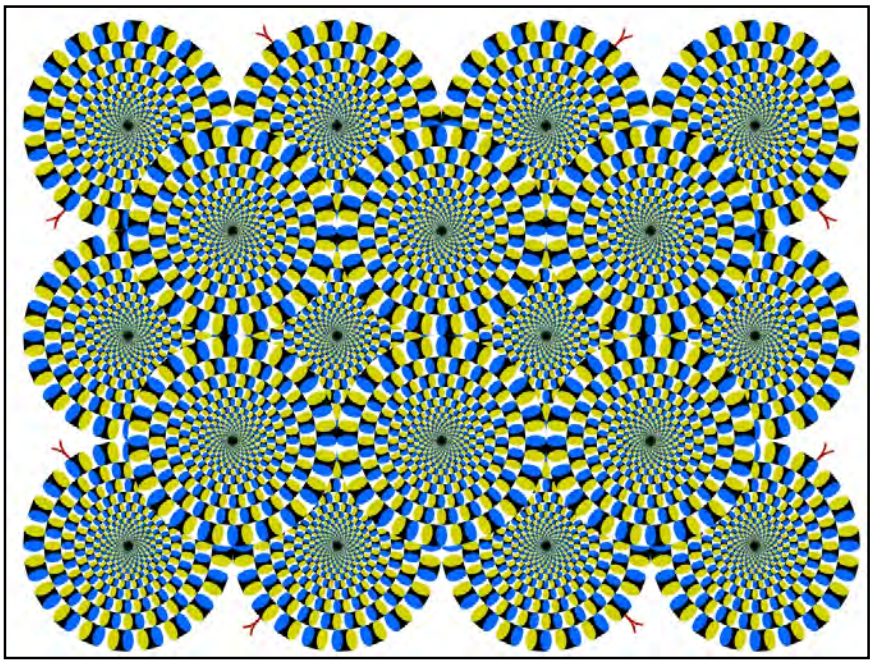


Figure 5. Rotating Snake Illusion (Kitaoka & Ashida, 2003).

The major debate is whether the saccadic system is colour blind, as traditionally thought (Breitmeyer (1984); Lovegrove, Garzia & Nicholson (1990), in Livingstone, Rosen, Drislane & Galaburda, 1991), or whether there are projections to the superior colliculus which are sensitive to colour as suggested by White, *et al.*, 2009, which help to drive attention, and

saccades in a bottom up manner. Cole, Heywood, Kentridge, Fairholm & Cowey (2003) report that motion signals based on chromatic information succeeded in capturing attention in patients with cerebral achromatopsia. However when the chromatic contours were masked, they failed to recruit attention. They argue that colour opponent systems were responsible for attentional capture, even in the absence of colour experience. White, *et al.*, contend that their finding of colour sensitivity in the superior colliculus has '...important implications for the role of cone-specific signals in visual orienting...' (p. 12165).

1.3.1 General features of the magnocellular and parvocellular systems

Segregation of the output of the two retinal ganglion cell types appears to be maintained throughout the visual pathway, through the lateral geniculate nucleus (LGN) and to the primary visual areas. The large parasol ganglion cells output to the magnocellular (M) pathway and the midget ganglion cells output to the parvocellular (P) pathway. The M and P visual sub-systems, provide input to the dorsal and ventral streams respectively (Ungerleider & Mishkin, 1982; Goodale & Milner, 1992) but *not* exclusively. To some extent, the M system responds to movement and form, and is responsible for locating objects in, and executing actions in global space whereas the (P) visual subsystem is sensitive to form and colour. There appear to be important interactions between these systems, one of which is that the parvocellular system receives inputs from the M system to prevent image overlap and blurring (Kolb & Whishaw, 1996).

In short, the M and P systems seem to be different in terms of their colour selectivity, temporal characteristics, contrast sensitivity and spatial resolution (Hubel & Livingstone, 1990; Livingstone, Rosen, Drislane & Galaburda, 1991). The M system does not appear to be colour sensitive (in the sense of coding colour information *per se*), has high contrast sensitivity and low spatial resolution, whereas the P system is colour sensitive, has low contrast sensitivity and relatively high spatial resolution (*ibid.*). Figure 6 shows the connectivity of the photoreceptors from which the P and M system signals are derived, and gives a simple illustration of the relative magnitude of the receptive fields.

The neurobiological meaning of the terms magnocellular and parvocellular need to be clarified. There is some ambiguity in the literature regarding the use of these terms. When these terms are used in this dissertation, they refer primarily to the retino-geniculate pathways

and although there are undoubtedly post-geniculate pathways which receive input from these pathways, they are not synonymous with them, and are not intended to be regarded as being synonomous with them, or even in any strict sense, continuations of them. Segregation of these fibres at the LGN is evident, but beyond the LGN, there are clearly many interactions between these pathways. Breitmeyer (2014) points out that although the 'dorsal' M-pathway contributes directly to conscious vision of motion and indirectly with the P-pathway to support various 'feed-forward' re-entrant loops, thought to be important for conscious vision, the P-pathway independently processes visual features which are necessary for conscious visual processing (see Breitmeyer, 2014). Bar, Kassam, Ghuman, Boshyan, Schmid, Dale, & Hämäläinen (2006) suggest that the pre-frontal cortex (most likely the OFC) receives imput from an early visual area – probably from the 'dorsal magnocellular pathway'. This evokes activity in OFC 50 ms earlier than in the temporal cortex. While they refer to the 'dorsal' M pathway, it is not clear that they are asserting that a post-geniculate M pathway exists in clear distinction to a parvocellular based post-geniculate pathway. Moreover, it does not seem important to their basic thesis that such a pathway could, or does indeed exist. Kveraga, Ghuman & Bar similarly discuss the cortical processing of M- vs. P-related signals, but although they suggest that the coarse (presumably M-derived signal) as being projected from early visual or subcortical regions, they do not state whether the M-signal is derived from a primary visual area, or some subcortical route.

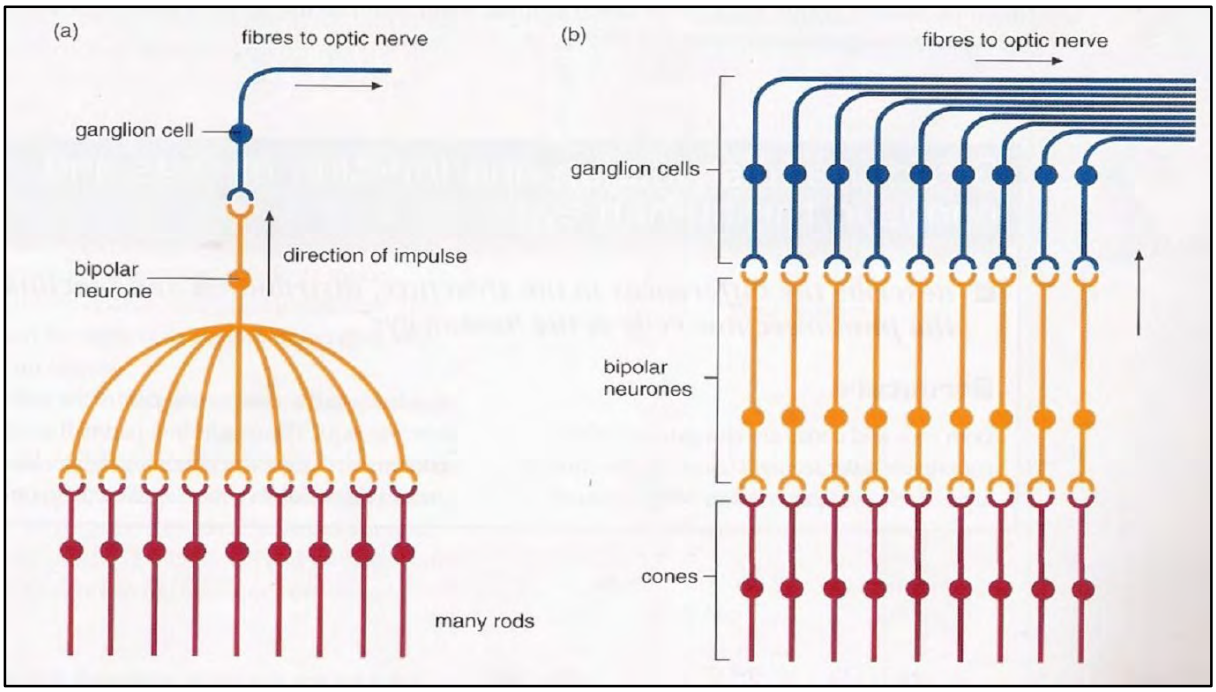


Figure 6. Rod vs. Cone receptor connections and receptive fields (Ravall, 2012).

It does not seem likely that this would be a projection directly from the M layers of the LGN, however. The fact that these authors refer to the 'dorsal magnocellular pathway' suggests that they support the idea that there is an M pathway that is distinct from the P pathway, and that it exists independently of the P pathway as a continuation of some distinct early area of the visual cortex and one of its important projections is to the PFC.

Another proponent of the idea that there is a dorsal stream in the cortex which is effectively a continuation of the M system which is clearly segregated at the LGN, is Bullier (2001). Bullier asserts that there are projections from a cortical area involving a network of regions in the parietal cortex (MT, MST, FEF) which feed information back to areas V1 and V2, and these are presumed to be part of what this author terms 'The Fast Brain' which belongs to the 'dorsal stream'. Bullier does not explicitly state that the feedback connections extend beyond V1 or V2, but suggests that activation of the dorsal stream is driven by the M channel. The essential point which Bullier makes is that the post-geniculate M channel drives a 'first pass' analysis of global level features of a visual percept and a cortical circuit is strongly dependent on inputs from this M channel information. Conduction speeds in this circuit are, according to Bullier's argument, fast enough to justify calling this network the 'Fast Brain' as horizontal connections conduct too slowly to cope with the speed that this processing requires.

1.3.2 The magnocellular system and reading theory

M dysfunction has been suggested as a possible cause of visual-perceptual reading disabilities. The research of Livingstone & Hubel (1987b); Hubel & Livingstone (1990); Livingstone, Rosen, Drislane, & Galaburda (1991); Galaburda & Livingstone (1993) found a variety of evidence which suggested M system abnormalities in people diagnosed with developmental dyslexia. In a post-mortem study, these abnormalities were seen clearly in the magnocellular layers of the lateral geniculate nuclei – see Figure 7. Note also the differences in the laminar organisation.

Whether a 'dorsal magnocellular pathway' exists as such, or there is simply a cortical network which receives input from a pathway which is dominated by magnocellular input is a case in point, and the literature is sometimes ambiguous. For example, Fabre-Thorpe (2011) refers to such a 'dorsal magnocellular pathway' which affects object processing by means of 'scene gists', suggesting that it may modulate such processing at very early stages. This is the

mechanism which Bar, *et al.*, propose. Although the sheer conduction speed of magnocellular fibres between the LGN and primary visual areas is in the order of a few milliseconds faster than those of parvocellular fibres, conduction speed alone cannot explain the processing disruptions or delays that Galaburda & Livingstone's (1993) research found. These authors found 20-50 ms delays in the evoked potential signal and 100 – 200 ms delays in associated visual processing tasks. This signifies that although the retino-geniculate signals to the primary visual areas may be delayed slightly by magnocellular deficits, more significant problems are evident beyond this pathway and in a cortical network. It appears, therefore, that magnocellular conduction speed is not the only, or perhaps most critical factor in the higher processing of M-related signals. These authors speculated that deficits stemming from magnocellular dysfunction could be evident in 'higher cortical association areas' which they suggested, could reflect functional segregation. This suggests that M system related deficits and therefore M system related functions are the product of segregation which stem from the LGN to cortical systems.

Galaburda & Livingstone's research could only indentify potential problems with perceptual processing and could only conclude that it was the relatively slow speed of the M system, relative to the P system which might explain the resulting perceptual problems. They did not establish the existence of separate and segregated cortical systems which could be described as extensions of these perceptual pathways.

It does not seem clear from any of the literature consulted, that the dorsal stream depends exclusively on input from the M system. Although Kveraga, Ghuman & Bar assert that the '…M projections comprise most of the dorsal or "where", visual stream…' they do not state that the M system is the exclusive source of its input. They do make the suggestion that M and P projections remain largely segregated (in V1) they form the dorsal and ventral streams. They also assert the existence of connections between the two 'cortical visual systems' and various subcortical nuclei, such as between the posterior IT to the superior colliculus (SC) and between the anterior IT to the pulvinar and mediodorsal nucleus of the thalamus. This adds somewhat to the unclarity about whether the M and P pathways (or at least the M pathway) is as distinct as a cortical stream as it is as a retino-thalamic-primary visual pathway. In any case, these authors make it clear that there are numerous interactions between these pathways at both subcortical and cortical levels.

Other authors make this debate somewhat less clear by suggesting, for example, that the M-pathway provides the majority of the dorsal stream projection terminating in the parietal cortex (Laycock, Crewthera & Crewthera, 2007). However, there seems to be some agreement that a) there are multiple convergences of M, P and K (koniocellular) inputs in many cortical areas (MT/V5, V4). They explain these convergences are providing 'separable but temporally complementary contributions to the human Visual Evoked Potential' (and presumably functionally complementary contributions). They maintain that the characteristics of the M system suit it to making rapid feed-forward connections to pre-frontal areas which lead to substantially faster cortical processing than the mere 10 or so ms 'magnocellular advantage' would suggest. They also support the idea that this fast system probably enhances multiple cortical interactions at all levels of the visual system. The feedforward and feedback or recurrent processing provided by this fast magnocellularly based system may perform a critical modulation function for visual attention in V1. Importantly, their model implies that M signals arrive in V1, are projected through the dorsal stream to areas like V5 and possibly V4. Frontal projections might also be back-projected, along with the feedback connections from the dorsal stream, back into V1 where they may influence and modulate processing of slower P and M signals.

Other authors tend to support the idea that the M and P pathways are major primary sources for two major cortical visual system, or that these cortical systems may be effective continuations of these pathways. They specifically relate these pathways to the dorsal and ventral streams. Their conclusions appear to exclude the possible involvement of the dorsal stream in object recognition, and generally support the tradition model that the dorsal system mainly is involved in motion detection and the ventral stream in object recognition or discrimination.

The best available evidence seems to suggest that the M and P systems interact in important ways. According to Martínez, Revheim, Butler, Guilfoyle, Dias, & Javitt (2013) the two systems work together in complementary ways, and this is evident in studies of reading. The M system appears to play a role in word tracking, eye fixation and imput selection and the ventral system provides more detailed analysis of words.

In summary, there seem to be authors who suggest that cortical systems are virtual extensions of the M and P systems, and that these systems persist and are important beyond the retino-

geniculate nucleus. None of the literature consulted suggests that these systems are completely independent, and much of it suggests complementary and interacting roles. For the purpose of this dissertation, no strong views are taken regarding the relative independence of these systems, and the M and P systems are understood, by and large, to represent particular nerve fibre types and connections between the retina and LGN which project to various primary cortical and subcortical areas.

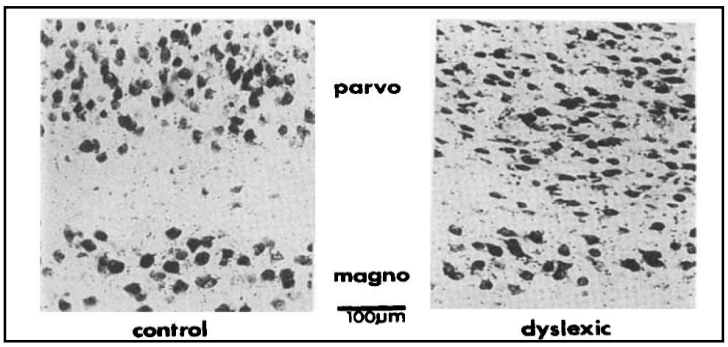


Figure 7. Comparison of P & M cells in the LGN (from Galaburda & Livingstone, 1993).

Many authors agree that the M system appears to play a role in timing or coordinating eye movements during reading (e.g. Stein, 2001) and this is one of the problems attributed to M-system dysfunction in reading disorders.

Since reading involves rapid processing of textual information, claims that participants are able to respond to subliminally presented text rest on an assumption that the visual system can process text fast enough for semantic activation to occur. This assumption seems closely related to the fact that rapid visual processing is necessary for fluent reading. Segalowitz & Zheng (2009) suggest that lexical access occurs as early as100 ms, and semantic access from 168 ms, although this does not imply that text shown at shorter durations cannot lead to activation, since visual system activation may persist beyond exposure time. The skill demonstrated in reading suggests that a significant amount of visual processing occurs automatically, without conscious effort and probably by means of massively parallel processing. It is useful to consider some aspects of reading theory, and in particular, theories of dyslexia, especially in relation to the putative function of the magnocellular system.

The research of Chase, Ashourzadeh, Kelly, Monfette and Kinsey (2003) was predicated on the notion of differential colour sensitivity in the M and P systems. These authors conducted a series of experiments which suggest that the M pathway (which is fast but carries low spatial

frequency information) is likely to be the dominant pathway used in reading. Chase, *et al.* (2003) findings showed that it may be selectively impaired by the use of red filters in reading, whereas the functioning of the slower P system may be facilitated by longer wavelength filters. The main implication of this study is the idea that reading fluency appears to depend on the fast M system conducting coarse information in order to facilitate text (and perhaps word contour) recognition so that reading is a relatively automatic and effortless task.

Although the role of the M system in dyslexia has been disputed (Skottun, 2005), and many researchers believe that developmental dyslexia is caused by phonemic awareness deficits and manifests in problems with sounding out words (Rayner *et al*. in Chase, 2003), there are indications, both theoretical and empirical which point to the potential significance of the M system in processing information quickly.

Fowler & Stein (2005) found evidence that yellow filters can improve M system functioning, by eliminating (blue) input from S-cone cells which is thought to provide inhibitory input to the M pathway. They claim that providing yellow filters to children with reading impairments, improves their reading ability by the enhancement of functions associated with the M system (motion sensitivity, convergence, and accommodation).

1.3.3 The M system and the attentional blink

A related set of ideas and research stems from what is termed the 'attentional blink' (AB) paradigm (Shapiro, Arnell & Raymond, 1997). The AB occurs when a series of masked targets are presented within 500 ms of each other and subjects are typically unable to report the second target (T2), although they are able to report the first (T1) correctly. When T1 and T2 are presented more than 500 ms apart, or when subjects are told to ignore T1, they are able to report T2. Shapiro *et al.* argued that although T2 cannot be reported under these conditions, it is processed by the visual system 'to a stage just short of awareness' (p. 291) since T2 elicits the usual N1 and P2 signals and also the N400 ERP associated with meaning (*ibid.*, p. 292). They attribute the inability of subjects to identify T2 to interference with a short term storage buffer and that the increased attentional effort required to bring a masked T1 stimulus to awareness means that T2 is vulnerable to decay. Even though T2 cannot be reported, interactions between T1 and T2 may be seen.

There is debate as to whether the interference occurs at a late stage (when the percept has already been processed to a high degree by the visual system), or an earlier stage in the visual system (Giesbrecht, Bischof & Kingstone, 2003). Giesbrecht, Bischof & Kingstone (2004) tested the AB effect under two conditions: photopic (when subjects are light adapted), and scotopic (dark adapted) viewing conditions. Their finding was that the AB was evident in the photopic, but not the scotopic condition. Thus, when subjects are light adapted, a typical AB effect is seen, but when subjects are dark adapted, their T2 detection accuracy is not affected at the same temporal lags.

It may be possible to interpret these findings by reference to the characteristics of the M and P pathways. It is known that the M system has significant inputs from rod cells (Jacoby & Marshak, 2000), and that these cells function under scotopic light conditions. The M system has high temporal resolution (i.e. it is able to segregate events that are temporally close to each other) and high contrast sensitivity, whereas the P system is slower, has lower temporal resolution, and lower contrast sensitivity. Nieuwenhuis, Jepma, La Fors & Olivers (2008) have explored this (M/P) possible explanation in a series of experiments on the AB where stimuli were presented against a red background, on an 'equiluminant' background, and following flicker or motion adaptation. Their reasoning was that if the M system's function could be selectively impaired, subjects' performance in the AB would be affected in the M condition (i.e. there would be an increased AB effect). This hypothesis was not supported, and they concluded that the AB 'was not systematically affected by manipulations that changed the relative involvement of the magno- and parvo-cellular pathways' (p. 44).

The reason for the failure of this experiment might lie in the assumption that the first two conditions (1. dark red type presented against a bright red background; 2. colour contrast condition: bright green type against a yellow background), did actually impair M system function (aside from whether the flicker condition was effective or not). Their results in fact indicate that subjects found the 'equiluminant' colour contrast stimuli **more** visible than the luminance contrast stimuli and this should not be the case, if equiluminance has been correctly determined and M system function is actually compromised. I will comment on this in more detail in the final chapter.

Another experiment, though not directly related to the AB, found evidence for M system involvement in a task in which subjects were required to identify flanked letters (e.g. 'xax') in 'magno-disadvantageous colour contrast' or 'parvo-disadvantageous weak luminance contrast' conditions (Omtzicht, Hendriks, Kolk, 2002). They concluded that naming performance in the luminance contrast ('parvo-disadvantageous') condition was significantly better. However, there was no difference in performance when letters were presented singly. This supports the idea of M system facilitation and possibly suggesting that it had a role beyond simple registration of the stimuli and there was some early processing which led to this performance advantage.

Another experiment potentially supports a role for the M system in mitigating the effects of the AB. Kristjánsson & Nakayama (2002) found that T1/T2 discrimination is better when T2 stimuli are presented quite far away from T1. They also found that discrimination was worse within a region around the attended site (presumably the fixation point), and attributed this to lateral inhibition. It is possible that M system activation might explain the better discrimination when T2 targets are presented away from T1 targets.

One of the major issues in perceptual and reading theory is the role of the M system and this was discussed further in relation to AB research, since one of the key theoretical issues in the AB has to do with low-level (visual system) vs. high level (attentional) processes. Several lines of research suggest that the M system has a role to play in the AB, and that it appears to mitigate the effects of the AB due to its fast, high temporal resolution characteristics. While the research of Nieuwenhuis, Jepma, La Fors & Olivers (2008) potentially refutes this notion, and possibly that of Chase *et al.* (2003), there may be flaws in the procedure by which they established color equiluminance. The assumption that any 'equiluminant' colour contrast condition disadvantages the M system may also not be valid.

1.3.4 The magnocellular system and implicit cognition

It is proposed that the M system is important for mediating fast, automatic responses in implicit cognition. Thus, the function of the M system may be to bias cognitive systems towards survival-orientated decisions and potentiate fast, and relatively automatic responses.

The M system is considered to be a phylogenetically primary system, geared towards detection of movement, and 'rapid orientation towards moving stimuli' (Barton, 2004). The P system performs fine-grain processing of information, including colour, and would be important for diurnal foraging where colour (for example in fruit) provides important clues about the availability and quality of food (Rowe, 2002).

The M system is thought to provide input to midbrain areas, such as the superior colliculus (SC), and a projection from this region to the pulvinar nucleus of the thalamus (Diamond & Hall, 1969, in Kveraga, *et al.*, 2007). There also appear to be early projections from the visual system (probably magnocellular) to the amygdala (Pasley, Mayes, & Schultz (2004) in Todorov, Fiske & Prentice (2011); Williams, in Duncan & Barrett (2007); Williams, Morris, McGlone, Abbott & Mattingley, 2004). Inputs to the amygdala are thought to carry low spatial frequency information (LSF) (Vuilleumier, Armony, Driver & Dolan, 2003), and support for the idea that these inputs emanate from the M system may be available. Vuilleumier *et al.*, 2003 found evidence that subcortical nuclei (amygdala, tectum, pulvinar) were responsive to LSF information from viewed faces, but 'blind' to high spatial frequency (HSF) information. This finding has been generally supported by other research (Holmes, Winston & Eimer, 2005; Mermillod, Vuilleumier, Guyader, Alleysson & Marendaz, 2005; Jetha, Zheng, Schmidt & Segalowitz, 2012; see de Gelder, Van den Stock, Meeren, Sinke, Kret & Tamietto, 2010 for a review of important literature in this area).

Interestingly, amygdala activation to threatening ('fear') facial expressions, does not depend on the subject's awareness of the presentation of the faces (Phelps & LeDoux, 2005), and even subliminal presentations increase activation in this subcortical circuit (SC, pulvinar, amygdala) (Morris, deBonis & Dolan 2002). This circuit may also enhance cortical sensitivity to emotional sounds and voices (Vuilleumier, 2005).

The idea of threat appraisal which occurs before any conscious representation is available has received the attention of researchers for some time. For example, LeDoux (1996) described a limbic circuit which responds to emotional features of a stimulus, before it can be analysed by the cortex. As LeDoux explains, 'The cortical systems that try to do the understanding are only involved in the emotional process after the fact…' (1996, p. 241). It is thus suggested that there are separate circuits in both limbic and cortical areas which process emotional and

cognitive aspects in parallel and that the limbic circuit produces threat-related arousal very rapidly, preparing the autonomic system for an emergency response. The adaptive benefits of such a system are not difficult to see.

It is generally acknowledged that the M system plays an important role in modulating visual attention (Steinman, Steinman & Lehmkuhle 1996). These researchers found that M-biased cues reliably overrode P-biased cues designed to evoke visual attention in a line-motion illusion. Another source of attentional bias or modulation, arises from signals generated by the amygdala in response to an appraisal of the emotional salience of a stimulus. Vuilleumier (2005) suggests this mechanism is like an emotional attention system, and that it might also be the trigger for a top-down attention system which exerts an information processing bias by means of emotional arousal. This might at times compete with other top down systems, but at other times, it may amplify 'other sources of top-down control on perception' (2005, p. 585). However, there are indications that affective primes may lead to stronger priming effects than semantic primes in an implicit task (Gawronski, Deutsch & Siedel, 2005) so the attentional bias exerted by the affective system may become dominant in situations where there is perceptual threat. As a source of bias, emotional salience may be a more important influence in implicit cognition than semantic or conceptual information.

1.3.5 Top-down perceptual priming

Bar (2003) has suggested a possible mechanism by which rapid cognitive decisions may be made. The idea involves top-down facilitation of visual perceptual processing and hypothesises several processing stages:

1. Projection of low spatial frequency information from early visual areas directly to the prefrontal cortex.

2. This low frequency (LF) information activates expectations or hypotheses, about possible interpretation of the input.

3. Information related to these hypotheses is back-projected to tertiary visual processing areas (inferior temporal cortex) '…where they activate corresponding object representations to be integrated with the bottom-up process'.

Top-down back-projections are hypothesized as constraining the number of possible identifications, in order to speed up processing. The early activation of an ‘‘initial guess’’ based only on coarse information is understood to facilitate recognition and a fast response.

Bar points out that object recognition may be accomplished within 150-200 ms from stimulus onset. Bottom-up visual processing is slow, and involves processing of HSF information, so a pure bottom-up process would probably not be fast enough for object identification when the stimulus signal is partial, or degraded. The top-down system suggested by Bar, would need to receive low spatial frequency information projected very quickly from early visual areas in order to provide an processing advantage.

Bar suggests 2 candidate systems for mediating top-down facilitation:

1. The ventrolateral PFC, and the orbital PFC. Bar suggests that these regions ‘comprise a network where object-related semantic knowledge is activated in the ventrolateral PFC by the rapid projection from early visual cortex, and then transferred to the orbital PFC where expectations are generated and projected top-down’ (Bar, 2003, p. 603), presumably to temporal cortical areas. The ventrolateral PFC is known to be involved in visual object recognition in monkeys but in human studies much of the data points to the role of the inferior frontal gyrus.

2. The second candidate, according to Bar, involves the orbital PFC. This system is highly linked to subcortical nuclei such as the amgydala. This input might constitute a kind of ‘relevance’ tagging, and Bar suggests that regions within the orbital PFC are involved with integrating inputs from the ventrolateral PFC, the amygdala and various other structures. This model is especially relevant for processing danger cues where a rapid analysis and response is required. Inference of danger would provide activation to other brain areas involved in organising relevant behaviour (such as the autonomic system and pre-motor areas).

Bar (2001, in Bar, 2003) suggests that top-down facilitation occurs in all recognition conditions, but reports evidence that it was enhanced when images were presented briefly and masked, compared with non-masked objects. Conversely, the top-down process may not

be activated when the bottom-up process is accomplished so quickly that the top-down process does not have time to develop. Another possibility is that the magnitude of the top-down processing is modulated by the ease with which the stimulus is recognized (thus fewer candidates are activated and the amount of processing required in the PFC areas is less with easier recognition problems). Context knowledge may also help to generate cues that facilitate object recognition.

This theory of top-down processing was tested by Kveraga, Boshyan & Bar (2007b) in research where subjects were asked to judge a series of serially presented line drawings by the criterion 'is the object bigger or smaller than a shoe box?' There were two conditions in the task: the stimuli were either line drawings presented as 'M-biased' (achromatic, with low luminance contrast) or 'P-biased' (red line drawings against a green background, i.e. isoluminant, but with colour contrast). For the latter, chromatically defined stimuli, a procedure was developed to determine the isoluminance point (between the brightness of the red line drawing against the green background) for each subject. This involved presenting the line drawings in rapidly alternating (red and green) colours in the range of 12 - 20 Hz. They found that the flicker frequency of 14 Hz gave the best results. After this isoluminance setting procedure for each subject, all P-biased stimuli were displayed at the determined point of isoluminance.

Kveraga *et al.* found a performance advantage of about 100 ms for the M condition, and a slight improvement in identification accuracy. The BOLD imaging signal showed results consistent with their hypotheses, i.e. that the M-biased presentations were associated with greater activation of the OFC, and the P-biased presentations were associated with more activation of temporal regions. They also found more activation in the amygdala for M-biased compared with P-biased stimuli.

This research provides support for the idea that the M and P systems can be functionally dissociated using a methodology which biases stimuli towards either the M or P system and the research supports the conjecture that there is a top-down system based on the M pathway, which lends a performance advantage in categorising visual objects.

One of the possible implications of this research, is that the M system, because of the manner in which it facilitates the speed and accuracy of responses, may be associated with a more automatic style of cognition. This is because identification is facilitated by back-projections that are based on a 'gist' of what the object might be. In other words, perceptual decisions are made on the basis of LSF information which prime certain compatible object representations. To use an example from Kveraga *et al.*, low spatial frequency information could activate several candidate object representations because of their similar LSF properties. This would provide a processing speed advantage in that the number of possible object representations is constrained when this top-down information is integrated with information from a bottom-up stream (i.e. from fine-grained analysis performed by the P system), which ultimately passes many stages of feature detection, and activates complex stuctures in the ventral temporal cortex. The ventral temporal cortex is hypothesised to be the juncture for top-down and bottom-up streams (Kveraga, *et al.,* 2007b; Kveraga, *et al.,* 2007; Bar, 2003).

These authors make explicit assumptions about the relative independence of these two visual sub-systems which need to be examined in some detail. The possibility that colour opponency information may be available to mid-brain structures which appear to drive attention in a bottom-up manner has been discussed (Crook *et al.*, White, *et al*.). These authors question the idea that the superior colliculus (SC) is 'colour blind' as traditionally assumed. This assumption rests on a conjecture that the SC receives signals mainly, or exclusively from the M system, and while this may be partly true, the experimental evidence is equivocal. In addition, it is not clear that parasol ganglion cells receive information exclusively from rod photoreceptors (see Bordt, Hoshi, Yamada, Perryman-Stout & Marshak, 2006). The ideas which informed the isoluminance determination of Kveraga, *et al*. (2007b) will be discussed in the review which follows and some of the practical and methodological problems will be discussed in section 2.3.

1.4 Review of the neurophysiology of visual perception

1.4.1 Introduction

The idea that there are neurophysiological, neuroanatomical and functional distinctions between the M and P systems has a history and I will attempt to present some of the important points of this history. I will review the basic theory of visual processing, with a

historical and developmental perspective and examine the implications for the theory and methodology underlying this research project

1.4.2 Historical perspective

Much of our knowledge of the visual system has been derived from non-human studies. For example, Hubel & Wiesel (1959) did some of the pioneering studies of the receptive fields of single 'neurones' in the cat's striate cortex, and they described excitatory and inhibitory regions around these cells (p. 576). Their (1962) study described simple and complex cells in the visual cortex. They also described simple cells which had 'on' and 'off' regions (p. 109) characterised by excitatory and inhibitory input. The complex cells had larger receptive fields and their complexity was in the fact that their response to light could not be predicted by the usual arrangements of excitatory and inhibitory surround regions that characterised the simple cells. Enroth-Cugell & Robson (1966, in Hochstein & Shapley, 1976) are credited with the classification of cat retinal ganglion cells as X- or Y -like '...on the basis of linearity or nonlinearity of spatial summation...' (Hochstein & Shapley, 1976, p. 237.). According to Hochstein & Shapley, Rodieck (1965, in *ibid*), assembled a theory which sought to explain the spatial and temporal response of ganglion cells linearly. The model explained the response of what was termed an X cell, but the Y cells' response was more complex. In the Y cell, there appeared to be a non-linear spatial summation of the receptive field (*ibid*., p. 238).

Gouras (1968, 1969 - in Dreher, Fukada & Rodieck, 1976) described 'tonic ganglion cells', with slow conduction and a linear response to input. Dreher *et al*. surmised that these ganglion cells receive input from X-cells in the retina. The 'phasic' ganglion cells, according to Dreher *et al*. receive input from retinal cells with Y-like properties, i.e. with faster conducting axons. These cells' receptive fields are larger, and the spatial signal summation is non-linear.

Shapley, Kaplan & Soodak (1981) studied spatial summation and contrast sensitivity of X and Y cells at the LGN of a macaque. Their work showed that ganglion axons which project to the LGN can also be classified as X, or Y on the basis of spatial summation (p. 543), and that while X cells can be found in the parvocellular layers and two of the magnocellular layers, Y cells can only be found in the magnocellular layers (*ibid*.). This was confirmed in a follow-up study when they found that the vast majority of cells were X-like. In the

parvocellular layers, 99% of the cells were X-like, and in the magnocellular layers, 75% were X-like. The other cells were Y-like and were mostly found in the magnocellular layers (Kaplan & Shapley, 1982).

The studies by Shapley *et al*. (1981), and Kaplan & Shapley (1982) show similarities between the cell types (X and Y) described in a cat by Hubel & Wiesel (1962) and those found in a macaque. The differences in their receptive fields appeared to capture this distinction. Kaplan & Shapley suggest that the Y cells found in the magnocellular layers of the macaque have distinct functional properties compared with X cells. Their investigations aimed to understand the structure and function of the cell types and the contributions of these pathways to visual perception. They speculate that separate analysis of colour and luminance begins in the retina, 'distal' to the ganglion cells' output (Kaplan & Shapley, 1986, p. 2755). The main difference between primate and cat tonic cells, according to Dreher, *et al*., is that monkey tonic cells are colour opponent, '...whereas phasic cells and cat X and Y cells are not'. (p. 434). The finding of differential contrast and receptive field properties and the clear segregation of cells at the LGN based on these characteristics suggests a significant functional distinction (Dreher, Fukada & Rodieck, 1976).

However, non-linear spatial summation may not be a definitive way of discriminating X and Y cells. Crook, *et al*., (2008) point out that although the 'Physiological link between the alpha-Y and magnocellular-parasol pathway has long been appreciated...' (p. 11277), most cells in the magnocellular layers of the LGN show linear spatial summation. Thus, the supposed Y-signature of non-linear spatial summation does not hold for the majority of M relay cells. They point out that non-linear summation is not unique to Y cell types and may be seen in many others. Their research from macaque recordings and histology suggests that the 'signature' of the Y cell is not non-linear spatial summation, but an 'F2' signal which occurs at double the temporal frequency of the stimulus. The F2 component shows a high contrast response and temporal sensitivity. Because only parasol cells show this frequency doubled response or harmonic, Crook, *et al*. imply that the magnocellular system which consists of parasol ganglion cells, is Y-like and suggest that parasol cells are the 'analog' of cat Y-cells in terms of their physiological, temporal and contrast properties.

The work of Crook *et al*. appears to resolve some of the confusion about the nature and properties of these basic cell classes and their relationship to the magnocellular system. They

show that the historic distinction between X and Y cells is still relevant, but that it is not based on spatial summation properties. Their research provides support for the notion that parasol cells are the '...correlate of the generic mammalian alpha cell...' (p. 11286), and this is reinforced by their finding of a collicular projection that consists of parasol cells with distinctively large cell diameters.

1.4.3 Perspectives on parallel pathways

The idea that there are at least two important information streams is generally accepted. As Goodale & Milner (1992) reflected on Ungerleider & Mishkin's (1982) finding that there are different and dissociated visual pathways for action and perception, they comment that this idea entered the literature in 1969 with a proposal by Schneider that there is '...an anatomical separation between the visual coding of the location of a stimulus and the identification of that stimulus...' (p. 20). Although the proposal has not been widely accepted in this form, Goodale & Milner acknowledge that the '...distinction between object identification and spatial localisation - between 'what' and 'where' has persisted...' (p. 20). There is clear evidence of segregation of the magnocellular and parvocellular pathways at the LGN, and also that these pathways provide the major input contributions to MT and V4 respectively, from which the posterior parietal and inferior temporal areas receive much of their input (*ibid*.). However, the segregation between these pathways is not rigid (see Goodale & Milner, 1992; Maunsell, Nealey & DePriest, 1990; Braddick, O'Brien, Wattam-Bell, Atkinson & Turner, 2000 in Laycock, Crewthera & Crewthera, 2007; Nealey & Maunsell, 1994). On the other hand, responses in MT are significantly reduced when the magnocellular division of the LGN is blocked in macaques (Maunsell, Nealey & DePriest, 1990). In fact, Maunsell *et al*. determined that area MT depends largely, but not exclusively, '...on the subcortical M channel for its excitatory input' (p. 2069). The fact that there are P channel contributions to this area does not diminish the significance of M channel inputs. Indeed, Greschner, Shlens, Bakolitsa, Field, Gauthier, Jepson & Sher (2010) report that there is correlated firing between major ganglion cell types in the retina, so it seems unlikely that synchronous activity at this level would not be evident, to some extent, at higher levels. Generally speaking, the fact that there are anatomical and probably functional interactions between M and P channels does not negate Ungerleider & Mishkin's basic distinction.

Goodale & Milner (1992) argue that despite the non-independence of these pathways anatomically, there is still an important functional distinction based on identification vs. visually based action involving objects. They suggest that

> It seems plausible from a functional standpoint that separate processing modules would have evolved to mediate the different uses to which vision can be put. This principle is already generally accepted in relation to 'automatic' types of behaviour such as saccadic eye movements... and it is possible that it could be extended to other systems for a range of behavioural skills such as visually guided reaching and grasping, in which close coordination is required between movements of the fingers, hands, upper limbs, head and eyes. (p. 21)

The idea that there are feature detectors which work in parallel has been discussed, but there is also a temporal aspect which needs to be mentioned. Bullier (2001) asserts that although M and P channels ultimately converge, M-channel mediated activity in V1 precedes P-channel mediated activity by about 20 ms. M neurons are highly myelinated and they appear to serve a network of cortical areas, which Bullier refers to as 'the fast brain'. Bullier suggests that activation of these circuits may constitute a 'first pass' analysis. Bullier explains that

> '...because of the similarity in latencies to visual stimulation in areas V1,V2 and MT and the speed of feedback axons, activity in area MT is in a position to modulate the early part of the responses of neurons in areas V1 and V2 that could therefore act as active blackboards for computations done in area MT.' (p. 97).

This introduces the idea that processing is both parallel and iterative, with feedback connections influencing processing at various stages. The fact that MT receives early projections means that it may inform dorsal stream processing and potentiate or guide movement.

Dorsal stream activity may also influence ventral stream activity and Bullier theorised that '...because of the rapid activation of neurons in the dorsal stream, it is likely that they can influence responses of neurons in the ventral stream mainly by retroinjecting information in areas V1 and V2' (ibid). Central to Bullier's argument is the idea that fast, low spatial frequency information provides a global percept that is integrated with a more detailed, local analysis in order to provide balanced processing. If the process of perception was simply feed-forward, Bullier argues that it would not be able to resolve visual anomalies like occlusion, shadows and reflections and facilitate balanced processing. Areas like V1 and V2 contain neurons that effectively have high magnification factors (p. 97) and cannot integrate larger perceptual elements.

An important difference between this model of visual processing, and that proposed by Bar, 2003; Kveraga *et al*. 2007b; Kveraga et al., 2007) is that the back-projections do not necessarily involve higher cortical, or frontal areas of the brain. However, Bullier's model allows for dorsal stream and higher order activation which precedes ventral stream processing and provides feedback which introjects global perceptual features to facilitate processing in this pathway. The difference in emphasis is partly between action (Bullier) and recognition (Bar, Kveraga, *et al*.) because Bullier's model involves activation of dorsal areas which are involved in encoding motion signals. It seems possible therefore, that actions may be potentiated by visual input, before the arrival of P signals in V1, and that they are not product of, or secondary to neural processes which serve object identification. The Bar, Kveraga, *et al*. model involves back-projections from frontal areas to ventral stream areas, like the inferior temporal cortex and one of the implications seems to be that action follows from, or after identification.

If object recognition is slower than dorsal stream activation, the major processing task, if control is to be exerted, is inhibition of a motor response that has already been potentiated. Munoz & Everling (2004) describe a task where voluntary control of eye movements involves suppression of an involuntary (pro-saccadic) response in favour of a voluntary (anti-saccadic) response. Their paradigm shows that inhibitory control is time consuming in the case of an 'abort' process, where a presented stimulus requires an incongruent response. As they point out, this may be seen in a comparable task, such as the Stroop task where participants are presented '...with the names of colours printed in colours and are instructed to name the print colours and ignore the words. Reaction times are faster when the print colours and colour names are compatible rather than incompatible' (p. 219). Anguera & Gazzaley (2011) also make this point and suggest that 'Both motor inhibition and sensory suppression are believed to be mediated by top-down control processes originating from the prefrontal cortex...' (p 230).

1.4.4 Perspectives on the neurophysiology of colour perception

The idea that the M system receives only input from rods and the P system only receives input from cones is simplistic. The basic distinction between *X* and *Y* (tonic/phasic) cells does not relate precisely to rod and cone photoreceptor types. It was pointed out by Kaplan & Shapley (1982) that 75% of the cells in the magnocellular layers of the LGN in the macaque

are *X* (tonic) cells, but the vast majority (99%) of *Y* cells are found in the magnocellular layers. Dreher, Fukada & Rodieck (1976) believed that a '...special feature of monkey tonic cells is that they are colour opponent...' (p. 434).

There is a consensus that the retinal ganglion cells which project to the parvocellular division of the LGN are colour opponent (Callway, 2005; Nassi & Callaway, 2009) and that the cells which project to the magnocellular division are 'spectrally broad band' (Klistorner, Crewther & Crewther, 1997) but not colour opponent. Essentially, cells in the M layers of the LGN are achromatic, whereas (midget) cells in the P layers of the LGN show either red-green, or blue-yellow opponency (Callway, 2005).

According to Jacoby, Stafford, Kouyama & Marshak (1996), about 20% of parasol cell signals are derived from diffuse bipolar cells which make contact with L and M (long frequency/red and medium frequency/green) cones - see Figure 8.

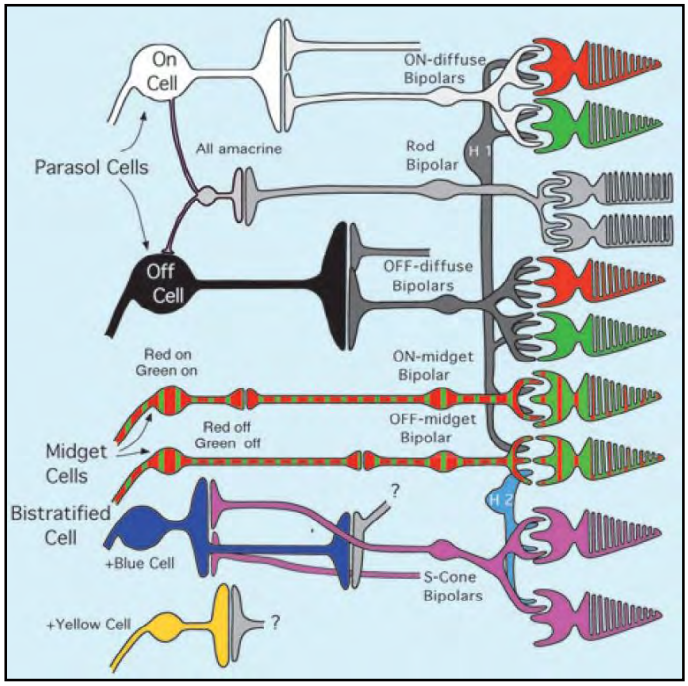


Figure 8. Retinal connectivity (from Lee, 2011). Note that parasol cells receive input from amacrine cells and also cone bipolar cells.

Another major signal source for parasol cells is the rod-type photoreceptor (Jacoby & Marshak, 2000). Chan, Martin, Clunas, & Grünert (2001) found that in the Marmoset, the rod signal pathway is as follows: (1) bipolar cells synapse onto rod cells, (2) amacrine cells (AII type) receive input from bipolar cells and provide a 'scotopic signal into the cone pathway...' (p. 227). 'Scotopic' refers to the signals generated at low ambient light levels. Thus, it

appears that rod receptors provide input to what is certainly a colour opponent pathway. According to Jacoby, *et al.* (1996) 80% of the input to parasol cells in the primate retina is from amacrine cells.

An interesting aspect to this discussion of scotopic signal inputs is in Martin's (1998) review of colour processing in the primate retina. That red-green opponent midget cells exist is well established in the literature. However, despite expectations that blue-yellow opponent cells would have similar morphology to the red-green cells, it seems clear from this, and earlier research (Dacey & Lee, 1994 - in *ibid.*) that this signal was produced by small-field (or simply 'small') bistratified ganglion cells (SBCs). These cells are interesting in that according to Martin, they receive excitatory input from 'S' cone bipolar cells (i.e. short wavelength cones, sensitive to blue light) and inhibitory input from diffuse bipolar cells which receive input from L and M cones. This effectively constitutes blue/yellow opponency since red and green combine additively to produce yellow sensitivity. Martin asserted that there was little evidence that parasol cells received input from S cones, and concluded that both the ON/OFF signals were derived from diffuse bipolar cells which receive input from L and M cones. Lee (2010) agrees that this idea has been refuted. However, there is some evidence that under certain conditions, SBCs and parasol cells may receive input from a common source.

Bordt, Hoshi, Yamada, Perryman-Stout & Marshak (2006) indirectly introduce a caveat to Martin's conclusion, although this only becomes evident in more recent research. Their study focused on the different inputs to ON and OFF parasol ganglion cells and concluded that there were subtle differences in terms of the signal sources. They examined an OFF parasol cell from the mid periphery of the retina and found that with the increase in eccentricity (or off-centredness) of the cell, there was a decrease in bipolar cell input (DB2, DB3), and an increase in amacrine cell input. Jacoby & Marshak (2000) point out that some of the DB3 cells which are presynaptic to the amacrine cells receive input from rod bipolar cells. One of the major functions of the amacrine inputs, according to Bordt, *et al.*, is to provide OFF parasol cells with information about the global properties of objects (p. 10), presumably because of the large receptive field provided by its multiple inputs. The caveat is suggested by Field, Greschner, Gauthier, Rangel, Shlens, Sher, Marshak & Litke (2009) who determined that (SBCs) receive input from rod cells which exhibit the same sign as S cone input (p. 1159).

These researchers' rationale for examining possible rod input to SBCs was partly logical. If SBCs do not receive rod input, approximately 10% of the axons in the optic nerve carry no useful information under scotopic (low light) conditions (p. 1159). Their research showed that SBCs do receive rod input via (AII) amacrine cells which '...form gap junctions[1] with ON cone bipolar cells that provide excitatory input to RGCs[2]...' (p. 1160). The finding of a gap junction path from rod to SBCs may have important implications for the altered perception of colour at lower (mesopic) light levels. They also suggest that this might explain the shift towards perceiving bluish hues when light levels approach the threshold where photopic (cone mediated) systems cannot function without rod input.

It also appears that SBC receptive fields are larger when scotopic conditions prevail (*ibid.*), and that this reverses under photopic conditions. It is interesting that this study recorded similar signals from both SBCs and ON parasol cells in low scotopic conditions. These authors suggest that there is a shift towards rod mediated vision in the various visual systems, including the magnocellular system, under scotopic conditions. Field, *et al.* believe that the visual system probably utilises all, or most fibers in the optic nerve under scotopic conditions. This was the logic on which part of their rationale was based. They do acknowledge studies which suggest that only some fraction of ganglion cells types are active under scotopic conditions, but they support their contention with the fact that '...AII amacrine cells form synapses with many cone bipolar types...' (p. 7).

The idea these authors advance is that as light levels change, there is a shift towards different photoreceptor based systems. Under normal light conditions, input is biased towards cone-based signals via various bipolar circuits. This is true of both the midget and parasol ganglion cells so presumably it should be true of both parvocellular and magnocellular systems. They did in fact find the expected high contrast sensitivity of SBCs and ON parasol cells in both photopic and scotopic conditions. Under mesopic conditions, there is input from both cone

[1] A gap junction occurs where adjacent cells are connected via membrane proteins which are arranged in such a way that they line up to form a channel between the cells. These junctions do not utilise neurotransmitters but allow for the direct, rapid conduction of an action potential between cells.

[2] Retinal ganglion cells

and rod receptors and in scotopic conditions input is coupled with rod receptors via AII amacrine cells. The broad implication is that there is no major redundancy in visual circuits when light conditions change from broad daylight to near total darkness.

Some support for this idea may come from Greschner, *et al.,* (2011). These authors found substantial, and significant correlations between different cell types which did not vary between photopic and scotopic conditions. They were surprised to observe that '...small bistratified cells, which receive ON input from S cones, fired synchronously with ON parasol and midget cells, which receive ON input primarily from L and M cones' (p. 75). This supports the notion that the different parts of the visual system work in parallel.

Another important finding of this study is that under high levels of light, SBCs showed the expected primary colour opponency. At lower (mesopic) levels this was altered and there was a shift to increased green sensitivity and although the response was colour opponent, suggesting there was still significant cone input, they attribute this increased green sensitivity to rod input. At lower light levels however, they found that the spectral sensitivity was compatible with that of rod cells. Figure 9 summarises this phenomenon.

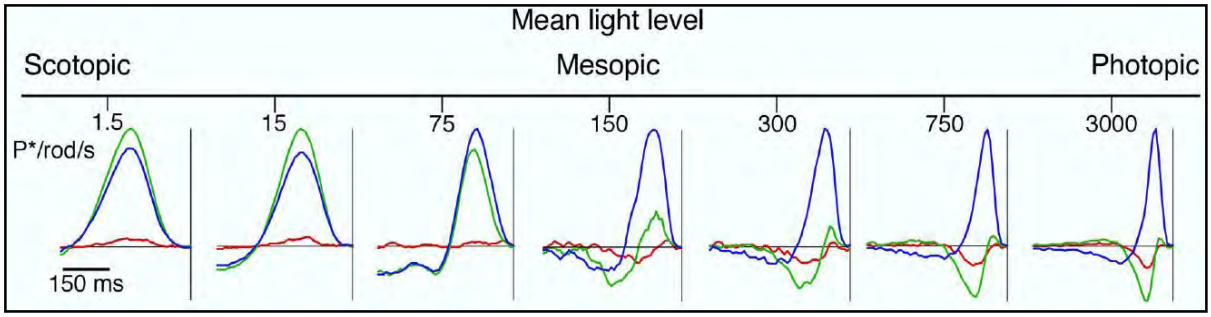


Figure 9. Spectral tuning of small bistratified cells (from Field, et al., p. 20). Note that blue/yellow opponency is evident under photopic, but not scotopic conditions.

In simple terms, it seems that as light levels dim towards the mesopic level, there is a tendency for scenes to appear 'bluish' and it becomes more difficult to distinguish red and yellow colours. In scotopic conditions, it is not possible to perceive colour. Because the visual system is not able to utilise colour opponent bipolar systems and switches to a rod-based signal source, sensitivity to longer light wavelengths is drastically reduced. Under these conditions, because sensitivity to blue and green light is similar, one would only be able to discern the difference between green and blue as a faint contrast - for example, a blue flower against green leaves under moonlight. A red flower would appear dark, and a yellow flower would look gray.

1.4.5 Implications for heterochromatic flicker photometry

It has been assumed that the lack of colour opponency in the M system (especially between L/red and M/green) is the basis on which the contribution of this channel may be subtracted (Hubel & Livingstone, 1990; Diller, Packer, Verweij, McMahon, Williams & Dacey, 2004; White, *et al.,* 2009; Lee, 2011). Practically, many researchers face a similar problem in determining isoluminance values for their participants and use fairly coarse adjustments on their equipment. For example, Kveraga (personal communication, 2010) stated that 'I also set the monitor output at slightly over 50% brightness, though that of course depends on the system.' Another problem is related to individual differences in perceived isoluminance (White, *et al.*, 2009). The range in which green/red contrasts are seen as isoluminant seems to be quite narrow, according to Kveraga. Kveraga stated that '...red has to be brighter than green (assuming somewhat linear computer color output) due to the sensitivity function of our color vision, and be in a fairly narrow range.' (*ibid.*).

The relatively coarse adjustments and context-specific solutions which are not easily applied to other contexts pose a practical challenge, but one wonders if there is also a theoretical aspect which has been overlooked. It seems clear that the M system receives much of its input from amacrine cells, which receives inputs from DB3 cells and rod bipolar cells. This arrangement is complex as some of the AII amacrine cells are presynaptic to the DB3 cells, and some occur postsynaptically as well. In short, although most of the parasol cell inputs are from amacrine cells, there are a significant number of direct inputs from DB cells (DB2, DB3, DB4, DB5). What changes occur in the M system when there is a shift from photopic to mesopic conditions? It seems likely that there would be a corresponding shift towards rod input, or away from bipolar input as this occurs and this may be associated with changes in colour sensitivity.

A second concern stems from observations that there is a possibility of a chromatic input to magnocellular cells, weak at low temporal frequencies, but stronger at higher frequencies (Lee & Sun, 2009). These authors suggest that this results from an interaction between two midget cell types with different sized receptive fields and is not the product of non-linear summation of M and L cones (p. 1). This signal, according to Lee & Sun, has been described as a response to 'red-green modulation' and is an 'excitatory' response to 'equiluminant' chromatic borders. It is related to the difference signal between L and M cones, and as

temporal frequency increases, the signal increases up to about 20 Hz. Lee & Sun suggest that this signal feature enhances motion signals in the magnocellular pathway '...at or near equiluminance.' (p. 2). They conclude that although the magnocellular system may '...provide the substrate for photometric tasks such as heterochromatic flicker photometry...' '…they respond to equiluminant red-green temporal alternation or spatial borders.' (p. 15). One of the functional implications is that this signal is likely to have an enhancement effect in 'natural scenes' of 'responses to red-green borders that are close to equal luminance'.

A controversy about whether the M system is inactive at isoluminance has persisted for at least 25 years. Livingstone & Hubel (1988) reflected this, although they remarked that 'If a particular magno cell sums red and green inputs, there will be a red : green ratio at which the red and green will be equally effective in stimulating the cell. This need not imply that every magno cell has the same ration of red to green inputs and therefore necessarily the same equiluminance point.' (p. 745). It is interesting that their illustrations of depth and shape loss due to isoluminance use blue/green colouring - see Figure 10.

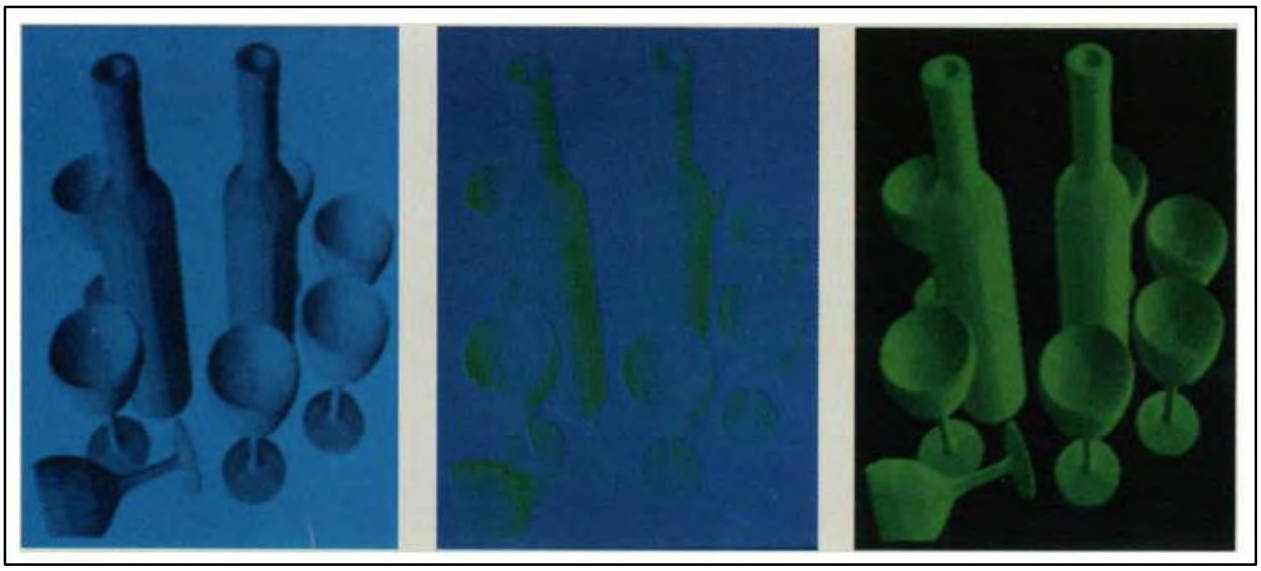


Figure 10. Loss of 3-dimensional shape due to 'equiluminance' (centre picture). From Hubel & Livingstone, 1988, p. 744). In the first and third columns, 3-dimensionality is easy to perceive, but in the centre picture, it is not easily evident.

Although Crook, *et al.*, (2008) found strong evidence that of a Y-cell projection to the superior colliculus, they state that this signal is non-opponent. Their characterisation of Y-cells involves the F2 (frequency doubled) harmonic (not the non-linear spatial summation criterion suggested by Shapley, Kaplan & Soodak, 1981), and they argue that the signal source for these cells is derived from cone bipolars. Other research points to the possibility that the superior colliculus does receive colour-related signals and was discussed earlier

(White, *et al.* 2009). White, *et al.* acknowledge received wisdom that the saccadic system is driven by an achromatic/luminance system thought to project from the magnocellular pathway, but they argue that the colour-related signals might involve a different pathway from this purportedly luminance based one. It may be worth speculating that there is some limited chromatic information available from the parasol/magnocellular system under certain conditions, especially taking Lee & Sun's (2009) findings into account. Indeed, the work of Kaplan & Shapley (1982) which distinguished X and Y cells on the grounds of spatial summation showed that 75% of cells in the magnocellular layers were X-like. Perhaps the collicular projection is a subset of one of these cell types.

There appear to be enough suggestions, from different sources and with a plausible neuroanatomical rationale, which suggests that the M system might receive some minor chromatic input to explain the finding of colour-related signals with 'limited color specificity' (White, *et al.,* 2009, p. 12159) at the superior colliculus. This may also qualify the opinion of Crook, *et al.*, (2008) that the projection is not colour-opponent.

If there are chromatic inputs to the M system, it seems likely that the range in which red/green isoluminance points can be determined is narrow and that this would depend on ambient light levels which appear to influence the signal source for components of the visual system (bipolar vs. rod/amacrine input). Lee & Sun's findings also raise concerns about using red/green heterochromatic flicker photometry if there is an 'excitatory' response to equiluminant chromatic borders. It may be that the possible range in which isoluminance points can be found is broader in the blue/green spectrum. I will now discuss this theoretically and practically.

Since the majority of parasol inputs seem to be rod based, it seems to follow that their responses to colour would be determined to a large extent, by rod spectral sensitivity. This falls between the maximum sensitivity of S and M cones (430 and 530 nm respectively), and is held to be about 495 nm (Gazzaniga, 2005). This leads to a prediction that the wavelengths which are least likely to be distinguished (perceived in terms of contrast) fall between these wavelengths. It can be speculated that the likelihood of whether red/green contrasts, or blue/green contrasts will be perceived as isoluminant by the M system depends to some extent on the ambient light levels and hence the amount of input from cone bipolars, but to a greater extent on the higher relative contribution of the rod system.

The most viable isoluminance solutions in this research seemed to involve green/cyan colours. Items coloured in this way were especially difficult to see when the values were high (i.e. close to the maximum light levels for a computer screen of 255). This is illustrated in Fig. 11 (LHS) where the colour values were RGB(0,216,230/0,255,0) which produces the cyan background and green image contours. Compare this with the RHS graphic where the colour values were RGB (0,141,0/150,0,0).

Figure 11. Cyan/Green vs. red/green colours. Note that the colours in the LHS picture make the face more difficult to recognize.

It is possible that the bright cyan/green combination results in the rod system being saturated, and this results in a signal to the magnocellular system which is close to null. The cyan/green picture is very difficult to see as it tends to shimmer and is visually confusing. This is reminiscent of a comment by Livingstone (1988), 'An object that is equiluminant with its background looks vibrant and unstable. The reason is that the parvo system can signal the object's shape, but the magno system cannot see its borders and therefore cannot signal either the movement or the position of the object.' (p. 83).

1.4.6 Conclusion

I have given a review of the neurophysiology of visual perception and discussed some of the important historical developments which help shape our understanding of the visual system. The first major distinction I discussed was the classification of cells as X- or Y-like, and the criterion of the receptive field which seemed to distinguish them. The relationship between this classification, and two major visual systems (parvocellular, magnocellular) is not clear when the criterion of spatial summation (linear, non-linear) is used to support the idea that the P system receives input from X cells, and the M system receives input from Y cells, and it

appears that 75% of cells in the M system layers are X-like. A more robust characterisation is based on an F2 (frequency doubled) harmonic and perhaps captures the difference between tonic and phasic cells. P and M cell properties seem to fit this characterisation of tonic/phasic, respectively.

I reflected on the idea of parallel pathways and the contribution of the M system to area MT, and the idea that movement, or action-related information is available very early to the dorsal system. An aspect of this idea is that fast, low spatial frequency information which arrives before the more detailed, high spatial frequency information, influences the way it is processed by providing a global percept. In this way, Bullier (2001) suggests that various levels of detail are integrated into a visual percept. A possible implication of this idea is that this global perceptual information which arrives rapidly and activates areas like MT, may be available to potentiate, or guide actions. This idea may be consistent with the dissociation between identification of objects and actions that involve objects which Mishkin & Ungerleider (1983) and Goodale & Milner (1992) described. It may also be possible that action-related information, such as an object's movement or location is available before the object can be completely identified. In this case, the role of inhibition is critical, so that an incorrect response can be inhibited after more detailed information about what the object is becomes available to influence a decision on whether to act, or what to do. This idea is slightly different to that of Bar, Kveraga, *et al.,* which emphasises identification as the primary issue.

I discussed the neurophysiology of colour perception and the different retinal cells which mediate red/green and blue/yellow opponency. A related discussion concerns the derivation of an achromatic signal and the relationship between the chromatic and achromatic systems. These systems are highly interrelated and their activities are correlated, but they provide different contributions to vision. There are several lines of evidence which suggest that the various visual systems function in parallel, even when there are major shifts in the availability of light. It was suggested that there are different processing modes associated with these changes in ambient light levels and that there is essentially no major redundancy in the visual system, even though some photoreceptors are not responsive to very low levels of light and others become saturated at high levels. The relative contribution of different photoreceptor types varies with changes in light levels and the associated circuits appear to 'switch' between photopic, mesopic, and scotopic processing modes. The main signal source

in photopic conditions is the cone and various bipolar system elements. In mesopic conditions, there is increased rod input and this begins to bias colour perception towards perceiving bluish colours. Under scotopic conditions, both the parvocellular and the magnocellular systems are active, but the main signal source is derived from rods. Because rod cells are not colour opponent, it is not possible to see colour (for example at night under moonlight), and the receptive field of the various cells becomes larger (possibly explaining why it is not possible to read a book under moonlight).

An important implication is that parasol cells receive input from cones, especially under photopic and mesopic conditions. There is evidence of some chromatic sensitivity in this pathway, especially to red/green contrasts which takes the form of an 'excitatory' response to 'equiluminant' chromatic borders. This is especially relevant to heterochromatic flicker photometry, as the signal increases up to about 20 Hz. It seems that red/green isoluminance solutions are difficult to determine, that they probably occur in a narrow range and they need to be determined individually.

It seems practically and theoretically possible that isoluminance in the blue/green range is more viable. The rationale is that this colour contrast is close to the range in which rods have maximum sensitivity and if they are saturated, their contribution to the magnocellular system is close to zero.

1.5 Implicit cognition investigated

The broad intention in this research project is to replicate an existing methodology, and then build on it to study the role that this visual perceptual sub-system plays in mediating response bias in the Implicit Association Test (IAT) by using a methodology which is intended to selectively cancel out the contribution of the M system which is hypothesised to mediate a reflexive, or more automatic response style.

1.5.1 A general neurobiological framework for Implicit Cognition

There is general agreement that rapid visual feature processing occurs in the brain, and that this is mediated without conscious effort (Crick & Koch, 2003), and this has been a thread running through the preceding discussion. I have briefly discussed Crick & Koch's idea of 'zombie mode', sensory processing contrasted with a more abstract, reflective mode of

cognition associated with consciousness. They argue that we are not conscious of neural activity in primary cortical areas, such as V1. Crick and Koch suggest that many actions that occur '…in response to sensory inputs are rapid, transient, stereotyped and unconscious.' (p. 120). In contrast, conscious thinking takes more effort, and focussed attention. One possible way of understanding consciousness, is that it emerges at the apex of a hierarchy of feature detectors, and represents highly complex and abstract information resulting from a 'feed-forward' flow of information (*ibid.*) and the co-activation of vast networks of neural feature detectors. Crick and Koch suggest that conscious experience results from the attentional selection and amplification of the activity of massive 'coalitions' of neurons. In order for an event to achieve conscious representation, there has to be sufficient activation between these neuronal assemblies. A stimulus can be selectively perceived and other objects ignored, so that when some pattern of neural feature detectors is activated and corresponds with an object representation of an important stimulus, (for example, a masked gunman running out of a bank) in the visual system, it can recruit attention very quickly.

1.5.2 Dual Systems theory as a framework for implicit cognition

I will introduce Dual Systems theory as a useful conceptual framework for the IAT, which will be discussed shortly. Human behaviour is organised according to various needs, priorities and often, urgencies. Some theoretical frameworks imply conflict, or opposition between different cognitive systems (e.g. Hofmann, Friese & Strack, 2009), and this distinction often falls along the lines of a conscious/unconscious thought process, although added to this, is probably an approach/avoidance system which adds many nuances (Davidson & Irwin, 1999).

Hofmann, *et al.* draw on Dual-Systems theory (Baumeister & Heatherton, 1996, in *ibid*), and theorise two opposing systems of *impulse* and *self-control*. Impulses arise from a motivation to respond to some need, fear, or important situational contingency (e.g. to eat, escape pain, or to engage in some pleasurable activity). The impulsive system predisposes towards immediate action and engagement, whereas the reflective system tends to inhibit some of these behaviours in line with longer term goals, strategies, values and concerns. The reflective system has to do with conscious reflection, strategising and planning, it therefore exerts a degree of control over the impulsive system. As Hofmann, *et al.,* put it '…these models share the general assumption that structurally different systems of information

processing underlie the production of impulsive, largely automatic forms of behaviour on the one hand and deliberate, largely controlled forms of behaviour on the other' (p. 164).

This dichotomy has already been alluded to in the previous discussions, which highlighted aspects of the visual system and the way that information is processed in certain, structured ways (e.g. the coding of colour, contrast, movement, emotional salience), and low level perceptual analysis tends to evoke forms of behaviour, which according to Crick and Koch, '…are rapid, transient, stereotyped and unconscious.' (in Mutalik, 2003, p. 120), for example, loud noise or sudden movement. It is fortunate that low level perceptual processing and various regulatory systems, such as proprioception occur without conscious monitoring, as this frees up cognitive resources so we can do a variety of things simultaneously (e.g. walk and talk, listen to the radio and drive – see Bargh, 1994). Conversely, controlled forms of behaviour are accompanied by states of awareness, in which sensory or other kinds of information are more fully integrated into our conscious thought (e.g. the smell, and taste of food), due to focussed attention. Planning, and controlling behaviour involves elements of inhibition, weighing up alternatives, anticipating the various possible outcomes, costs and benefits.

Hofmann, *et al.* suggest that behavioural impulses arise from the 'activation of certain associative clusters in long-term memory by perceptual or imagined stimulus input' (Metcalf & Mischel, 199l, Strack & Deutsch, 2004 in Hoffman, *et al.,* 2009). The probability of the behaviour being activated depends on affective arousal, previous conditioning and presumably, the status of various innate systems (appetitive) like hunger, thirst and sexual desire. Hoffman *et al.* suggest there is an associative system that determines the probability of activation and a behavioural response when a particular stimulus is encountered. In other words, a system by which associations are formed and strengthened between an item, such as chocolate and the 'behavioural schema' (to eat it) which is activated. An aversive reaction to chocolate will diminish the probability that one will eat it in future, whereas a positive reaction will strengthen it. They also state that this process by which these associations form, do not require attentional resources to function and do not depend on whether the person approves, or rejects the fact and implication of this associative link (*ibid.*). This 'associative cluster' may be 'reactivated quickly by perceptual input' (p. 165) and become automatic.

The reflective system described by Hofmann, *et al.,* involves a higher degree of phenomenal awareness and a strategic, deliberate and conscious mode of thinking. It also involves conceptual, abstract, reasoning, planning, reflection, for the maintenance of goal directed behaviour and flexibility in how goals are implemented.

Conflict between the impulsive and reflective system is one of the important implications of dual-process theory, but another may be that there is a degree of dissociation between these systems. The notion of an 'associative cluster' captures a link between a biologically relevant stimulus (e.g. food) and the behavioural schemas and programs which are activated in response to it (i.e. acquiring and eating the food). These schemas do not necessarily reflect the idealised view one may hold of oneself. Thus, one may consciously be aware of good health practices, but still be tempted by the smell and appearance of junk food. The conscious construction of oneself as being healthy is at odds with the appeal of unhealthy food and with diminished self-control due to stress, hunger, or situational factors, it is easy to succumb to temptation.

It is the dissociation between behavioural dispositions, and consciously held attitudes that has given the IAT its attractiveness as a research instrument. Expressed racial attitudes often contradict actual patterns of behaviour – e.g. showing preference for in-groups in many behaviours and contexts. Even at early stages of visual perception, exposure to racial out-groups seems to trigger arousal in limbic nuclei such as the amygdala (Lieberman, Hariri, Jarcho, Eisenberger & Bookheimer, 2005). This is almost certainly inconsistent with attitudes that would be overtly expressed in various contexts where one would prefer to be regarded as egalitarian, and unprejudiced. The activation which occurs in response to an out-group may therefore be strikingly different to the beliefs that participants hold about themselves that they are not prejudiced.

It has thus been argued that there may be conflict between 'impulsive' systems which produce relatively automatic and (usually) adaptive behaviours (e.g. turning away from something/one that is aversive, or perhaps just unfamiliar), vs. reflective systems, by which one can inhibit behavioural impulses in favour of more deliberate, strategic, and considered responses.

1.5.2 The Implicit Association Test

The Implicit Association Test (IAT) purports to measure associations between evaluative and other concepts and social objects which exist at the level of automatic response and are not necessarily available to conscious awareness (Greenwald & Banaji, 1995). These associations are relevant at the stage of behaviour when one acts towards a person, object, or social entity in such a way that bias is revealed. These behavioural predispositions (Rosenberg & Hovland, 1960; Yuker, 1965, in Wood, 2003), operate in such a way that they '…influence decision making and behaviour in social situations' (Wood, 2003). The concept of 'attitudes' is an important one in psychology, and generally these have been measured using self-report instruments, such as questionnaires. There seems to be relatively poor association between self-report measures, and actual behaviour however, especially with regard to stereotypes. People do not always seem to be able to, or willing to accurately report their attitudes on socially sensitive topics, like racial prejudice. The IAT uses response latency as a measure of task difficulty in sorting different exemplar words or pictures into various categories, along with congruent or non-congruent attributes. For instance, one of the classic IAT experiments involves pairing lists of flower and insect exemplars (i.e. concepts) with pleasant and unpleasant word lists (attributes). When the concepts and attributes are mapped incongruently onto a response key (e.g. sorting insects with pleasant words), response latencies are longer, than when congruent concepts and attributes (e.g. flowers and pleasant words) are mapped onto a response key. The IAT may therefore be understood as providing a measure of task difficulty in categorising exemplars of a certain category (e.g. flowers or insects), with polarised attributes (e.g. good or bad). The logic of this seems close to being tautological in that short reaction times are seen as evidence of preferential congruence, whereas relatively longer reaction times are interpreted as indications of preferential incongruence. Reaction time variation could reflect other kinds of congruence or incongruence, however such as semantic, or conceptual incongruence).

1.6 Conclusion

I have introduced the overall theme that there is an intimate relationship between visual sensory information and motor systems which act upon it (Engel, Fries, Konig, Brecht & Singer, 1999). There also appear to be visual-motor links which are important to imitation which is suggested to be an important developmental aspect of empathy. Intelligence, it is claimed, is an essentially social capacity which is linked to fitness and adequate social

functioning in a complex social environment. Emotion provides a motivational component to behaviour, and informs cognition in important ways. It provides survival-related information which biases cognition relevantly. There appear to be early appraisals of emotionally significant information which prepare the individual for rapid action, especially in relation to threat, or danger.

A rough framework for understanding the structure of the brain in relation to information processing and consciousness was suggested and the basic distinction was drawn between information processing in primary, secondary and tertiary sensory areas which generally occurs towards the back of the brain and the more highly integrated, abstract information which is available at the front of the brain and which provides a basis for reflection, planning and organising behaviour.

The role of attention systems in gating information in a bottom-up and top-down manner was discussed in relation to various needs for both fast, reflexive responses and more long-term, reflective decision making and response orchestration. Consciousness was seen as a mode of cognition in which the individual is able to conceive of her/his self as a 'self' in relation to other 'selves' who are also conscious and engaged in pursuing their own goals, which possible involves deceptive posturing. 'Mind-reading' is therefore an essential aspect of consciousness. While language is an important abstract code for representing the world, oneself and problem solving, it was suggested that language is not synonymous with thought, as many non-human, non-language speaking animals are capable of complex problem solving, especially in relation to comprehending social variables and deceiving other players.

Since the world yields far more information than can be utilised, attention is critical for being able to focus on relevant information, while non-relevant information is disregarded. Nonetheless, sensory information that does not achieve conscious representation is still processed by the various sensory channels and many complex behaviours can operate in a non-conscious, relatively automatic manner. Response automaticity is thus an important capacity which frees up cognitive resources so that numerous activities and cognitive processes can be managed simultaneously. In this sense, many behaviours and sensory and cognitive activities are 'implicit'. An important focus for this dissertation is the visual subsystems which are argued to underlie different styles of cognition and behaviour. It has been hypothesised that the two visual pathways (M/P) have temporal and spatial information

processing characteristics which mediate response styles which tend to be reflexive vs. reflective. It was important to try to map these widely accepted ideas onto what is known about the neuroanatomy of the retina and visual system, but recent literature reveals details which to some extent, seem to have been oversimplified. A considerable literature has accumulated which implicates M-system dysfunction in various clinical conditions such as dyslexia and schizophrenia. The M theory of dyslexia was a source of some experimental hypotheses and conjectures about the role and characteristics of the M system that have informed aspects of this research project. However, the most important aspect is the theory that visual object recognition is facilitated by a top-down visual perceptual pathway which is derived from the M pathway. The essential idea which this dissertation explores is that the M system carries information which facilitates object recognition, and provides a basis for perceptual and cognitive heuristics which potentiate rapid behavioural responses. This is thought to be relevant to decisions which categorise and organise everyday social objects and inform routine, or conventional responses. It seemed reasonable to hypothesise that the M system could play a role in mediating response bias in the IAT, and the aim of this research was to test this idea.

Chapter 2. Exploring properties of the magnocellular system

2.1 General introduction

The aim of this experiment was to investigate ideas about the purported role of the magnocellular (M) system in word recognition and more broadly, to test a computer-based methodology for dissociating the contributions of the M and parvocellular (P) to word recognition. This is the first in a series of experiments which tested various ideas and computer-based techniques.

2.2 Experiment 1: The role of the M system in reading

2.2.1 Introduction

The general theoretical areas which this experiment spans are reading theory, and more generally, visual perception theory (Bar, 2003). This experiment is an application of some ideas from Chase, *et al*., (2003) and an attempt to replicate their finding which suggests that the M system facilitates reading.

According to reading theory, the magnocellular (M) pathway in the brain has an important role in the fast transfer of low resolution information (LSF) to an area of the brain thought to be involved in top down processing of visual information. It is suggested that the M system plays a role in reading by projecting diffuse, LSF information that provides orthographic information which facilitates fast word recognition. Chase, *et al*. argue that the spatial frequency characteristics of '...channels used for text processing are well within the sensitivity range of the M pathway.' (p. 1213). Their basic argument is that the M system plays a primary role in reading, and their review of a variety of literature appears to support this. For example, Legge, 1978 (in Chase, et al., 2003), '...showed the low SFs (0.375-1.5 cycles/deg), that define the global pattern or shape, are extracted rapidly in 60-80 ms. Higher SFs (6.0-12.0 cycles/deg) are processed more slowly, requiring 150-200 ms.' (in *ibid*., p. 1213).

Chase, *et al*. conducted a series of experiments which support this model of reading, and concluded that the M system facilitates reading (in terms of speed and accuracy) under normal light conditions. A further elaboration of their findings is that the M system may be more responsive to light of higher frequencies than the parvocellular (P) system. They argue

on the basis of theoretical predictions and experimental evidence that the M system is more sensitive to light in the blue end of the spectrum, and it may be suppressed under certain conditions, such as reading text under red light.

The focus of this experiment was to test the relative word recognition accuracy under two experimental conditions: words presented against a red background, vs. words presented against a light blue background. The rationale of the experiment was to replicate the findings of Chase et al., and develop an understanding of the effects of different background colours in facilitating, or inhibiting recognition speed and accuracy. It was hypothesised that recognition would be facilitated when type was presented against a blue background, and response latency would be reduced. Conversely, it was expected that recognition accuracy would be reduced when type was presented on a red background, and response latency would be increased. The dependent measures were response latency (ms), and d'. 'D-prime' (d') is a standardised measure of signal detection sensitivity which takes noise and perceiver bias into account. The measure takes 4 response possibilities into account (hits, misses, false alarms/false positives and correct rejections/true negatives) and it shows how accurately a signal can be distinguished from noise. The d' score has a mean of 0, and a standard deviation of 1 and shows the separation between the signal and noise plus signal distribution curves. When d' = 0, the two curves are highly overlapped and detection is at the level of chance.

2.2.2 Participants

Participants (n = 18: 13 females, and 5 males) were recruited from an undergraduate class, and although they did not receive financial remuneration, they received course credit for doing the experiment. The experiment took a total of about 30 minutes. All data was coded with an anonymous ID which the participants chose, and recorded for themselves, so that they could identify their own data. Participants were informed that they could stop the experiment at any stage, by pressing a designated key, and that they were free to do so if they found it unpleasant for any reason, or simply wished to decline participation. They were given course credit for participation and their identity was not linked to their results. Ethical approval for this experiment and all the experiments which followed, was granted by the Research Ethics Committee in the Department of Psychology at the University of Cape Town.

2.2.3 Materials

The experiment was run on computers with the *Microsoft Windows XP* operating system, and the code was written by D. Mansfield in Delphi for Microsoft Win32 using the *Borland Developer Studio 2006* Integrated Development Environment (IDE).The software was installed on 6 workstations, and ran using configuration files, and stimuli in the installation sub-folders. Configuration files scripted the program exposure durations, block timing, mask characters and background colour definitions.

The blue background was RGB(185, 209, 235), and the red background was RGB(255,0,0). These are decimal equivalents of the hex values that define screen pixels. Screen pixel colours are defined by 4 1-byte values (red, green, blue, plus another byte which defines transparency). Each byte expresses the brightness of the respective colours of a pixel (0 = no illumination; 255 = full illumination). This arrangement allows for 24-bit colour depth, or 16777215 colours. The red background was a pure red, and the blue background was pale blue (see Figures 12, 13).

Figure 12. Experiment 1. Red word background designed to inhibit the contribution of the M system.

Figure 13. Experiment 1. Blue word background designed to enhance M system sensitivity.

The stimulus words consisted of a combined list of 100 correct words, and 50 pseudo-words. The correct word list was drawn randomly from an online word list (Kelk, 2003) and the pseudo-words were generated from a website (http://www.soybomb.com/tricks/words/) which creates nonsense words with legitimate English word phoneme patterns. The page generates fresh lists on demand. They were chosen with length and phonemic plausibility in mind. These lists were combined in random order for each trial, and the same source word lists were used for all trials but combined and ordered randomly each time.

2.2.4 Procedure

The experiment was intended to test word recognition accuracy under difficult conditions. When this experiment was designed no local (South African) data existed that could provide

guidance on what SOA (stimulus onset asynchrony) or ITI (inter-stimulus interval) would be appropriate. The values that were used in this experiment were based on extensive initial experimentation and dry runs with volunteers. A low number of targets vs. non-targets (50/100 respectively) was selected in order to increase the task difficulty and to discourage random responding. SOA and ITI timing was designed to make the task challenging and to test the limits. The task difficulty was moderated by using a calibration procedure so that inter-individual differences could be accommodated by varying the stimulus exposure time.

The experiment was done in the following stages:

2.2.4.1 Practice Phase

The word list, consisting of 100 valid words and 50 pseudowords was presented at a rate of 1 word every 1000 ms. The instructions were to press a key every time a miss-spelled or pseudoword was seen, but to do nothing when a valid word was presented. The program recorded keystrokes, and measured the response latency (i.e. the time between the stimulus presentation and a key press). All trials were preceded by a fixation point 'X'.
Visual masking was used, and the sequence of presentation was as follows:

1. Word (100 ms)	2. Mask (100 ms)	3. Blank screen (800 ms)
shepherd	**XXXXXXXXX**	

This effectively gave a window of 1000 ms, in which a response could be made. The keyboard was disabled after the 1000 ms response window, until the next stimulus was loaded from an internal array and the next trial began. In the Practice Phase, the exposure time began at 100 ms and if performance exceeded a predetermined level (hit rate >0.85), the exposure would decrease by 3 ms, until the hit rate threshold of 0.65 was reached, after which the exposure time was increased by 2 ms every trial. No participant managed to achieve a reduction of exposure time, suggesting that the calibration algorithm was overly strict. Feedback was given to participants by means of a red or green indicator, whether their response was correct or not (green = correct; red = error). Feedback was only given in the Practice Phase. Stimuli were presented in black type on a white background.

2.2.4.2 Phase 1

The same word list was used and randomised in a different way from the Practice Phase. The background colour of the target text was set randomly to either blue or red[1]:

1. Word 100 ms	**2. Mask 100 ms**	**3. Blank screen 800 ms**
joker	**XXXXXXXXX**	

2.2.4.3 Phase 2

The same list was used once more, but randomised into a different order again. For each trial: Practice, Phase 1, Phase 2, each word list (the valid word list and the pseudo word list) was shuffled into random order and combined in a random order to form a new list of 150 words and the background colour was set to either light blue or red, depending on how it was determined for the previous phase.

1. Word 100 ms	**2. Mask 100 ms**	**3. Blank screen 800 ms**
lionist	**XXXXXXXXX**	

Participants were provided with feedback in the form of graphs depicting their raw score rates which they could save for their own interest. Individual debriefing, and explanation was provided by the researcher on request. Refer to Appendix 1 (on media disk or Dropbox) for screen captures of the program dialogues, layout and instructions.

2.2.5 Results

A total of 17 participants completed the experiment (12 females, and 5 males). The data was assembled from the coded text files and the d' scores calculated using a custom program written in Delphi for Microsoft Win32 by D. Mansfield. The data was analysed using SPSS version 21.

Cases with hit rates of < 0.5 for the Practice Phase were excluded, and also cases where the Practice Phase was terminated prematurely. The rationale for doing this was that low hit rates

[1] This approach of randomly ordering trials was done to prevent expectations about trial ordering from interfering with performance. This was particularly a concern for larger studies that were run over longer periods of time, as participants might tend to collaborate and discuss expectations or experiences about which blocks were hard or easy.

suggested that the task difficulty level was too high, and/or that the level of effort was low. Low hit rates, combined with low false alarm rates suggested poor engagement or invalidity, and this criterion resulted in less variability in false alarm rates. Unfortunately, the sample size was reduced as a total of 8 participants were eliminated. Although in a small sample eliminating cases can reduce statistical power, on the other hand noisy data can easily confound existing effects.

Grouping participants as 'Selected' vs. 'Not Selected' on this criterion resulted in a final sample of $n = 9$. An ANOVA was constructed with 2 independent variables and 2 dependent variables (Selected/non-selected BY Response Type – Hit Rate/False alarm rate). Figure 14 shows an interesting pattern. The 'Not Selected' group had lower hit rates, but similar false alarm rates. An analysis of variance (ANOVA) confirmed this: $F(1, 15) = 6.5$, $p = 0.023$.

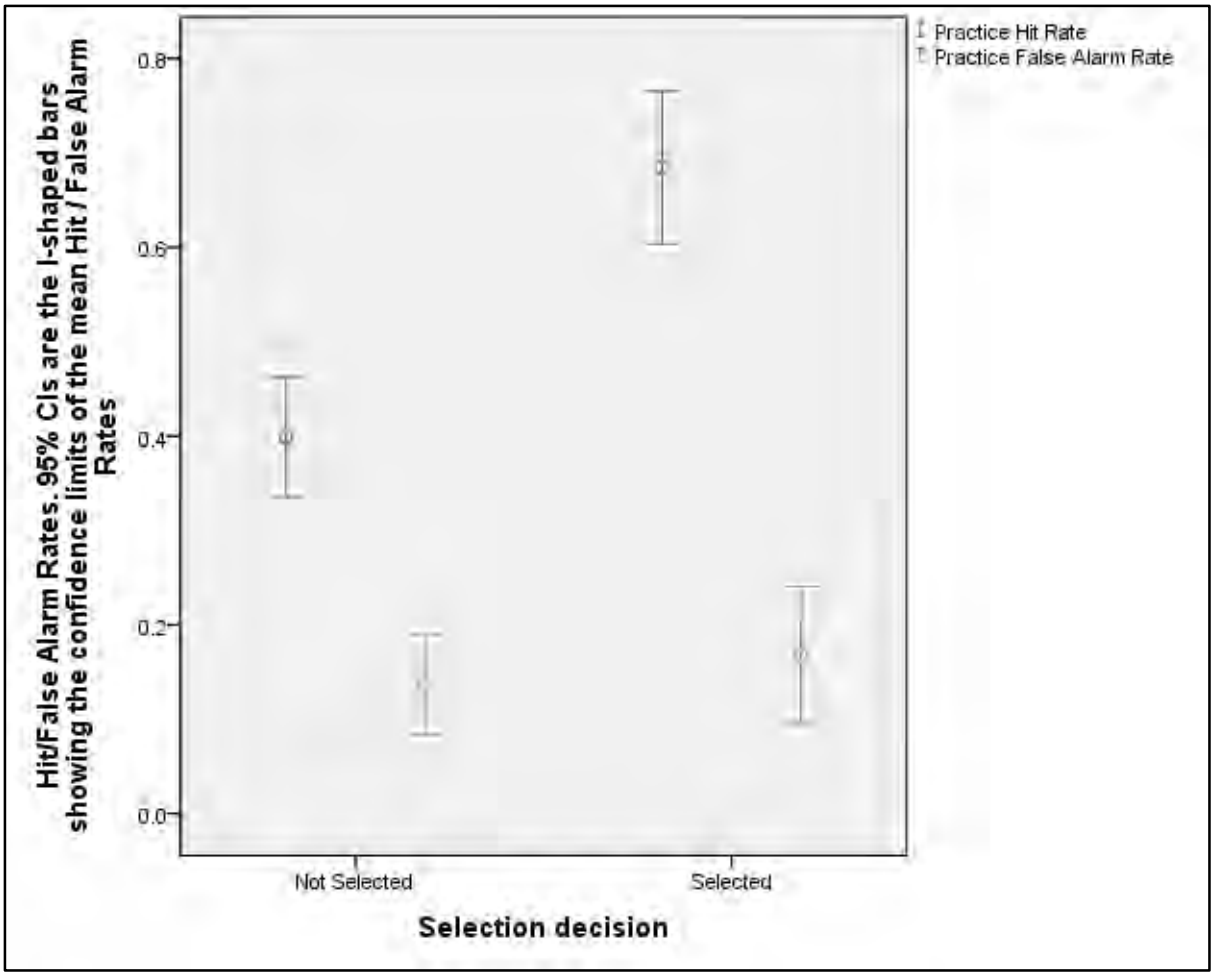


Figure 14. Experiment 1. 95% confidence intervals of mean hit rate and false alarm rate practice scores grouped by selection decision on the x-axis.

A Paired Samples t-test shows a statistically significant difference between the two conditions $t(8) = -2.53$, $p = 0.035$ (2-tailed). Because of the small sample size the Paired Samples t-test was repeated to run with 10000 bootstrap samples and the significance estimate was $p = 0.047$ (2-tailed). While the sample size is small ($n = 9$), there appears to be a clear pattern, consistent with theoretical predictions.

2.2.6 Discussion

This is a significant result, although the final data set was probably too small. Some lessons were learned from this experiment. Firstly, the randomisation procedure which swapped the blocks of trials was not satisfactory as 8 of the 9 experiments were run in the same order. This was due to a 'Randomize' instruction which initializes the random number generator with a random value being given twice in succession by the main program and a function which was called to shuffle the word list for each presentation. It is possible that there was a confound due to the non-random ordering and that there was a learning effect between the two conditions (red/blue text backgrounds).

These results may constitute a valid replication of the findings of Chase, *et al.,* but the sample was small and there were problems with the presentation ordering. The task was probably too difficult and the overall conclusion is necessarily tentative.

2.3 Methodological and practical challenges

2.3.1 Practical reflections and modelling

The discussion in section 1.4.6 suggested that the approach used by Kveraga *et al.* (2007b) to achieve isoluminant colour solutions would be difficult to replicate and that there may be inherent problems with using a red/green colour solution. I will describe a number of models that were used to try and replicate and test the methodology of Kveraga *et al*., and generate possible alternative solutions.

While there were some interesting aspects of the first research project, the findings were perhaps of less interest than lessons learned in the process of designing and implementing this experiment. The results suggested that detection accuracy was degraded by presenting text against a red back-ground. An important overall objective was to replicate the Kveraga, *et al*. experiment, so their methodology needed to be explored and adapted for local conditions

which involved using workstations in a computer lab, instead of projecting images into a head-set in an fMRI scanner. A critical objective was to develop software to do the isoluminance calibration procedure, image processing and run the experiment following their protocol as closely as possible.

The approach of Kveraga, *et al*., was to use line drawings, and transform them into low luminance contrast presentations (M-biased), or 'chromatically defined and isoluminant...' presentations ('P-biased') (p. 13233). These authors chose red-green for their isoluminant 'P-biased', presentations. Their method for determining the isoluminance point was to use flicker photometry, which displays the stimulus rapidly alternating between the two colours. They reported that in the frequency range between 12 - 20 Hz, '...the flicker caused by the luminance differences between these two colors seems to disappear for a very narrow range of luminance values, and the two colors fuse' (p. 13234). Their experience showed that an average alternation frequency of 14 Hz gave the best estimates. The range in which the flickering stopped, was very narrow, and it varied between participants. It thus had to be calibrated for each participant.

Practically, the procedure according to Kveraga, *et al.* was: 'Subjects were required to report via a key press whether the stimulus was flickering or appeared steady. Depending on the response, the output of the red gun was adjusted up or down in a pseudorandom manner so that many passes over the isoluminance point occurred during the procedure. The average of the values in the narrow range when a subject reported a steady stimulus became the isoluminance value for the subject used in the experiment. Thus, isoluminant stimuli were defined only by chromatic contrast between foreground and back-ground' (p. 13234).

A prototype image processing algorithm was developed using Embarcadero RAD Studio (2010), with Delphi for Win32. The file format used was bitmap (.bmp), since these are uncompressed images which do not suffer from compression artefacts, are faster to load and the format is easier to manipulate.

After the algorithm was developed, and some experimentation done, it proved difficult to find the general range of values in which the isoluminance point could be found. Kveraga (26 November, 2010, personal communication) gave some practical details: '...green was held constant at around 141 (on the RGB scale of 0-255), while red went up and down depending

on the response ("flickering or steady?"), crossing the isoluminance range several times, and the average usually ended up being plus or minus couple points around 150. The background for finding isoluminance was middle gray, I think around 127 or so. The fused drawings do not have an easily definable color, they're sort of grayish-yellowish I guess, but it's hard to say.' (See Appendix 2 for verbatim record of this correspondence).

Another source of variability lay in the fact that the monitor brightness settings were adjusted. According to Kveraga (26 November, 2010, personal communication), the monitor output was set at '...slightly over 50% brightness, though that of course depends on the system...' While this provided a starting point, the values needed to be more precise. Figure 15 shows an image transformed using the values suggested by Kveraga. Whether the colours are isoluminant or not is difficult to say.

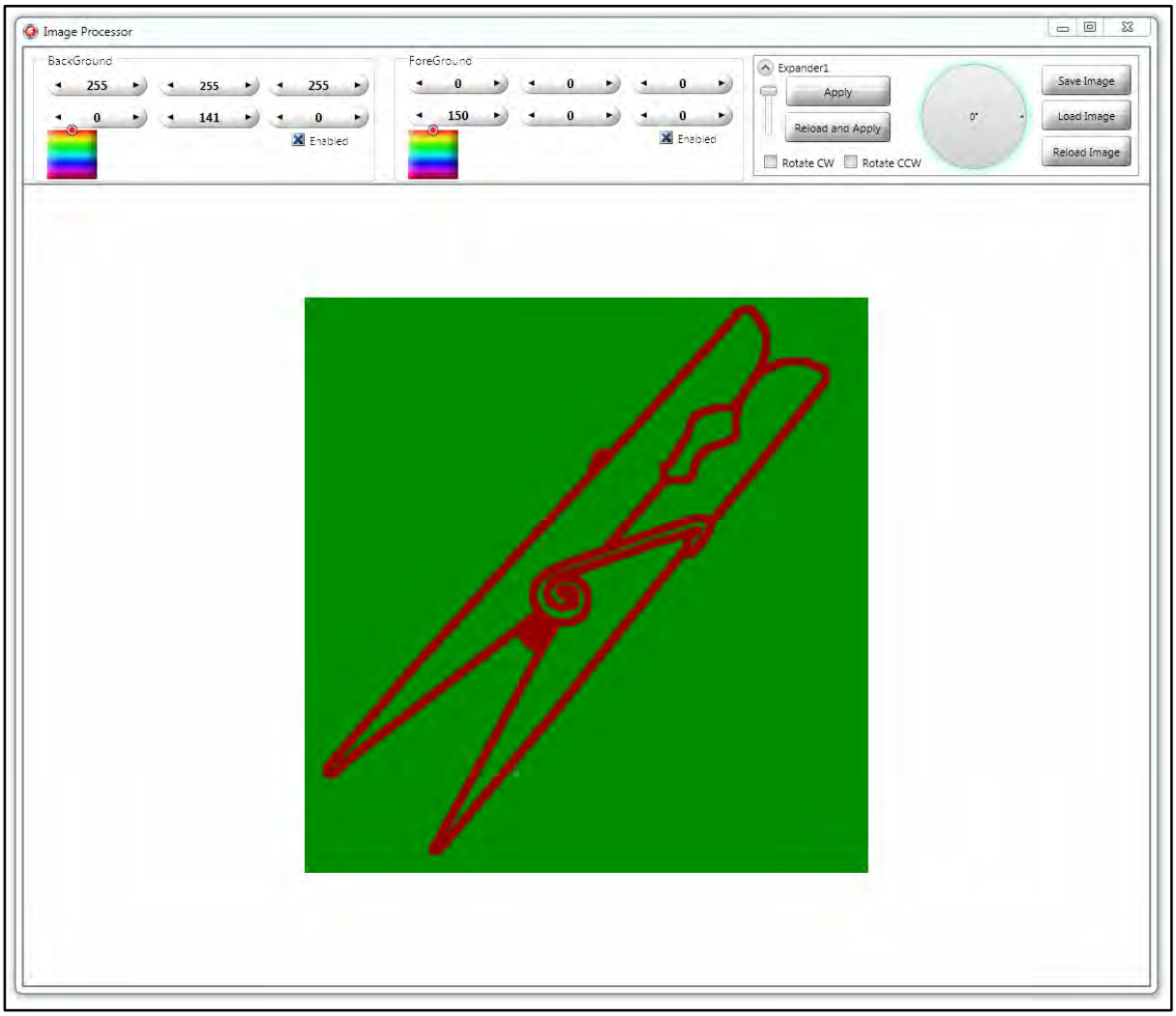


Figure 15. Image processing model using colours suggested by Kveraga, 2010. In this case, the 'Background' colour were transformed from white - RGB(255,255,255) to green – RGB(0,141,0) and the 'Foreground' colour from black – RGB(0,0,0) to red – RGB(150,0,0).

Software was developed to model the presentation format for calibrating isoluminance (Figures 16, 17). One would select a background source colour value, (usually RGB(255,255,255) = pure white) and transform it to a selected destination value (RGB(0,141,0) = green. The foreground source colour (RGB(0,0,0) = black), was transformed to RGB(150,0,0) = red. In this model, the colour of the text could be set to flicker between the two values below at any value between 1 and 16 Hz:

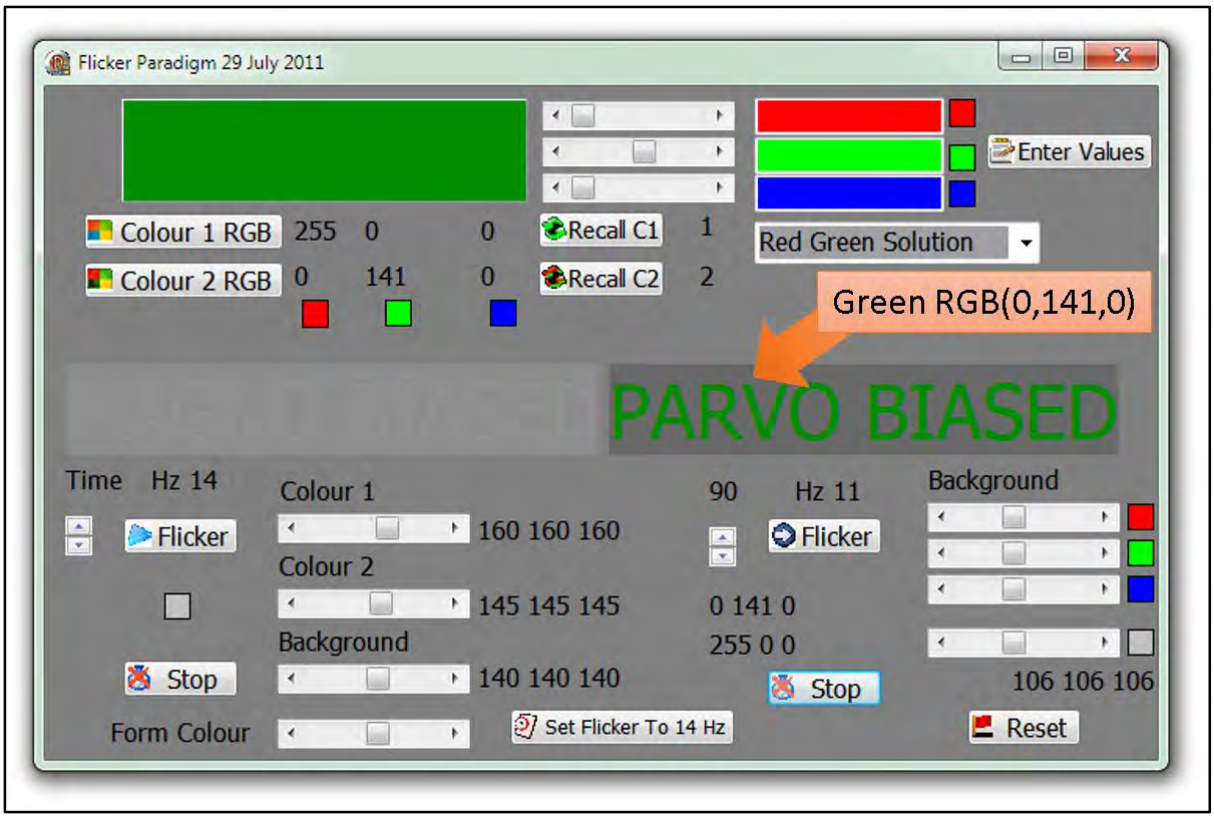


Figure 16. Flicker Model showing green: RGB(0,141,0) on the RHS, middle of the figure ('PARVO BIASED') text.

Figure 17. Flicker Model showing red: RGB(151,0,0) on the RHS, middle of the figure ('PARVO BIASED') text.

The flickering was very obvious, and the colours were darkened to try and approximate the suggestion that the monitor output be dimmed by about 50%. This probably is not a linear reduction but to illustrate the effect, Figures 18-19 show the result of an adjustment using a linear factor of -0.4:

Figure 18. Flicker Model showing red: RGB(60.0.0) on the RHS, middle of the figure ('PARVO BIASED') text.

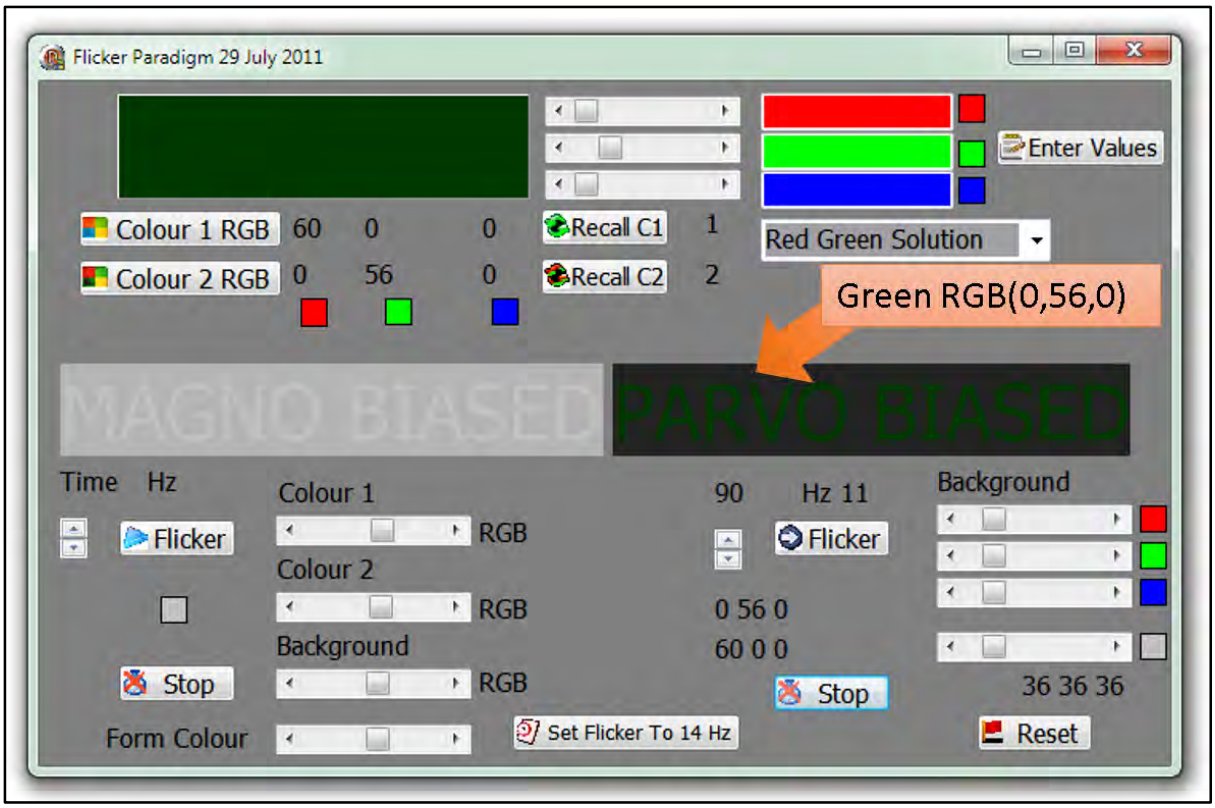


Figure 19. Flicker Model showing green: RGB(0,56,0) on the RHS, middle of the figure ('PARVO BIASED') text.

Flickering was still very obvious in this simulation and the background was adjusted to various values - shown here as RGB(36,36,36). Clearly, there was no textbook value which could be applied.

The colour combination where flickering was the least evident was with green/cyan alternation. A wide range of different values produced acceptable results. The questions that arose from this modelling were both theoretical and practical. It was theoretical in the sense that one had to ask why red/green combinations were commonly chosen and not other colour combinations. Practically, red/green isoluminance seemed difficult to achieve whereas green/cyan isoluminance was relatively easy to achieve in this flicker model. Another phenomenon seemed evident: in the red/green condition, images seemed clearer at smaller scales than in the green/cyan condition – see Figure 20.

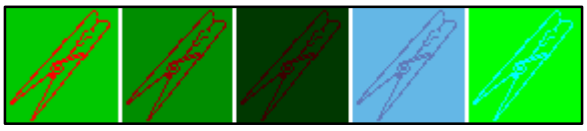


Figure 20. Scaled images in different colour combinations and brightness levels (panels are labelled 1 – 5, from left to right). The red/green images in panels 1 - 2 are easy to recognize. In panel 3, although it is dark, it is still recognizable. In panel 4, the image is recognizable in shades of blue. In panel 5, the cyan/green image makes the image very difficult to see.

These questions were explored in two experiments which follow. They were a prelude to the attempted replication of the Kveraga, *et al*. (2007b) experiment as there were questions about how to precisely calibrate the red/green values and the concern was that other combinations seemed to work better.

An article by Livingstone (1988) was part of the reason that other isoluminance solutions attracted my attention. Figure 21 illustrates how depth, and 3-dimensionality are affected by isoluminance:

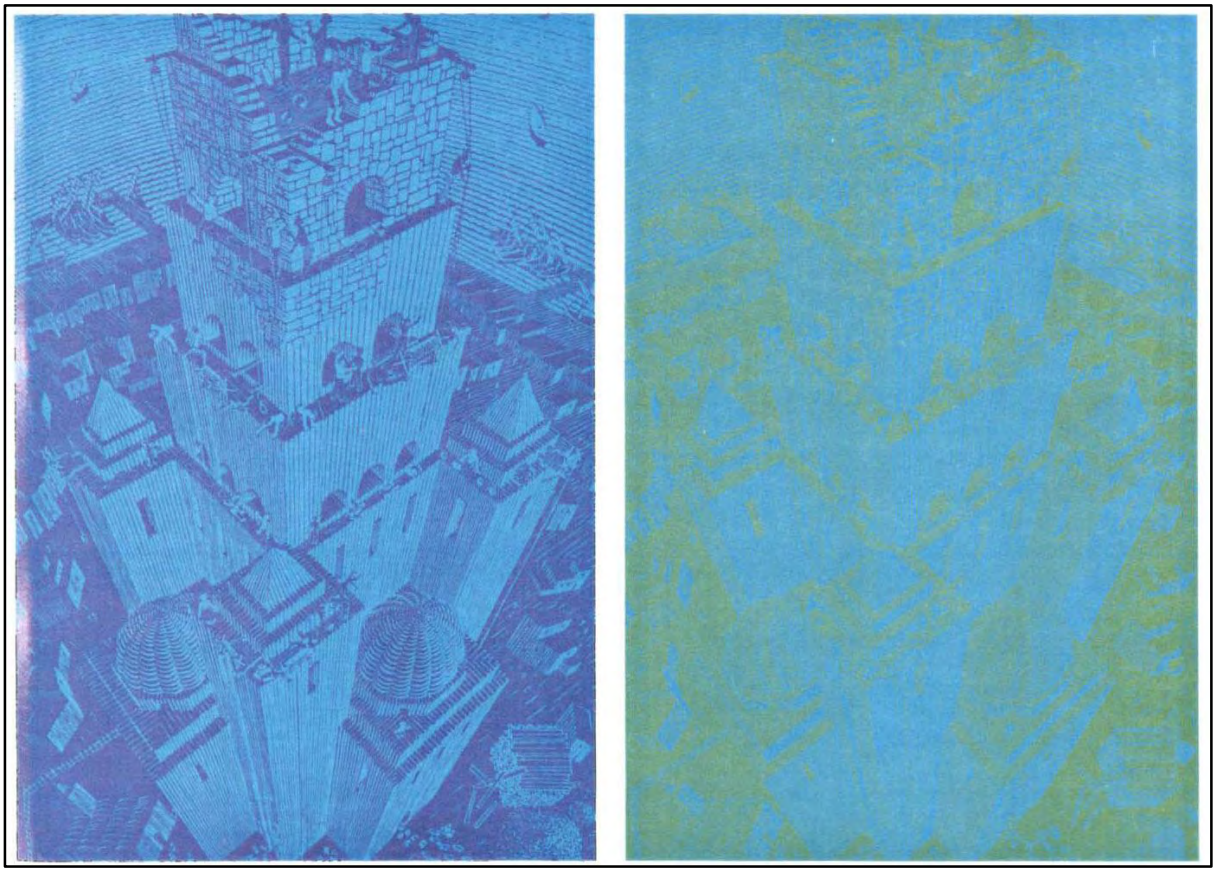


Figure 21. Tower of Babel by Escher (reproduced from Livingstone, 1988). The picture on the left hand side has luminance contrast and gives a strong impression of three-dimensionality. The picture on the right hand side shows that when the dark blue is replaced with green, the 3-d effect is lost and the picture is hard to see. This picture has colour contrast but poor luminance contrast.

The picture on the LHS has luminance contrast, and it is easy to recognise depth and 3-dimensionality. Livingstone explains the loss of depth, in the isoluminant picture on the right, as due to the relatively poor response of the M system to colour contrast. The choice of green and blue colouring is interesting as the effect is extremely clear.

Figure 22. Escher transformation: RGB(141,0,0) / (0,150,0). This picture was transformed from the same picture as in Figure 21, but the blue was replaced with red and the green altered to a different shade. This was an attempt at using the colours suggested by Kveraga, et al. (2007b).

Figure 23. Escher transformation (RGB(60,0,0) / (0,56,0). This picture was transformed from the same picture as in Figure 21, but the blue was replaced with red and the green altered to a different shade. This used the altered colour combinations suggested by Kveraga, et al. (2007b).

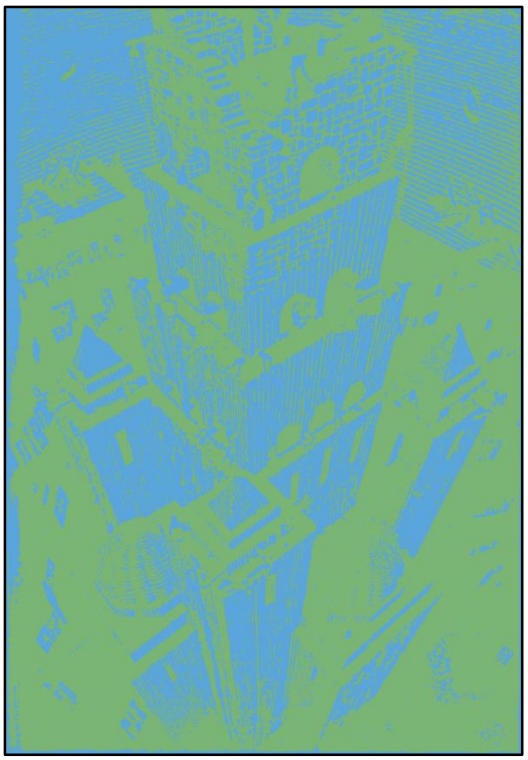


Figure 24. Escher transformation: RGB(90,165,220)/(123,181,117). These colours approximated the colours used by Livingstone (1988).

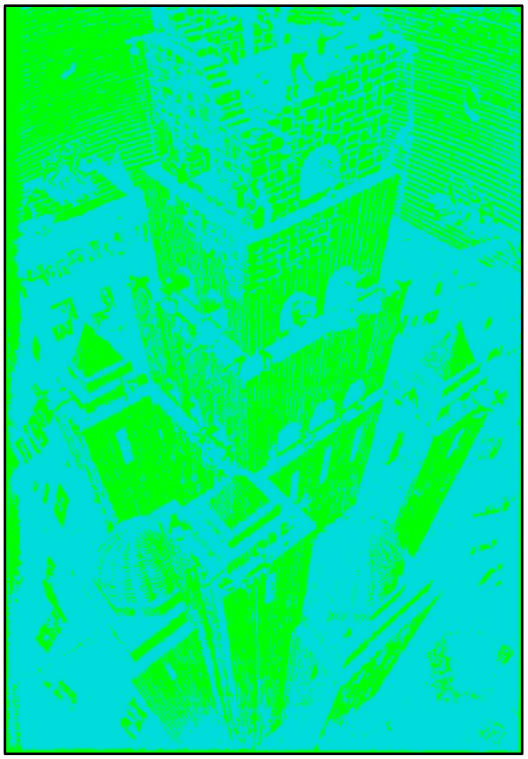


Figure 25. Escher transformation: RGB(0,220,220) / (0,255,0). The colour combination used in this transformed image was based on preliminary testing. Note that it is very difficult to perceive depth and get a gist of what the picture is.

Figures 22 – 25 show transformations of the Escher picture with various colour combinations. Figure 24 shows an approximation of Livingstone's transformation. While not accurate, it

does seem closer to Livingstone's understanding of 'isoluminance'. The image in Figure 22 and 23 seems easy to perceive and the depth and 3-dimensional cues do not seem to be affected to any significant extent. However, Figure 25 seems much more difficult to perceive and this is apparently because the contrast which gave Figure 23 and 24 clarity, defined the figure from the background and facilitated the perception of depth is diminished. The combination of cyan and green makes the image blurred and visually confusing. If the colours are reversed, the effect is still apparent - see Figure 26. The detail still exists, and this can be demonstrated by separating the blue and green components (middle and RH panel in Figure 26).

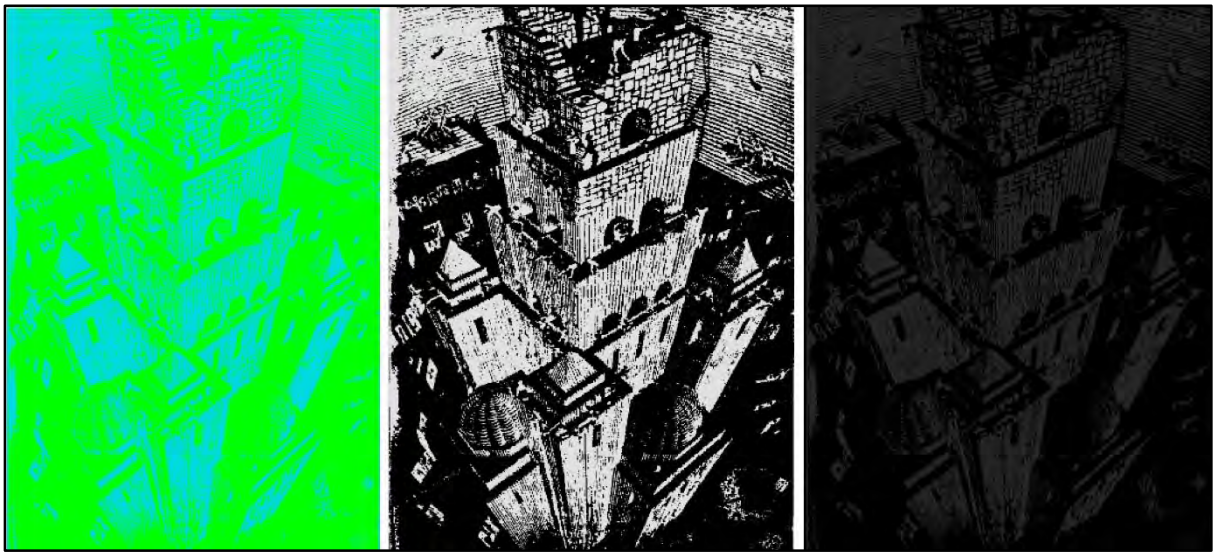


Figure 26. Reversal, blue and green channel of Escher picture. The same colours were used in the left hand panel, except they were reversed. The picture is very difficult to perceive and seems blurred. The middle panel is transformed from the left-most panel and the green is converted to black and the bluish colour to white. Note that it seems sharp and 3-dimensional. The third panel is transformed from the same image, but the colour contrast is low. This has little effect on one's ability to identify the picture and see the 3-d cues.

The flicker photometry simulation model and the various colour combinations and solutions that were tested made it seem that the cyan/green combination was most viable. This idea came from Livingstone, 1998. Extensive modelling and testing reinforced this. The next experiment tested this in a word recognition task, similar to the first one, but with words presented in a blue/green colour combination.

2.4 Experiment 2: The role of the M system in reading, a follow-up study

2.4.1 Introduction

This experiment repeated Experiment 1 which presented black type against a red/blue background, but in this experiment, different colour combinations were used for the visibility test of words since the red/green combination in the flicker photometry model seemed to be

difficult to implement, while the cyan/green colour combination seemed to work better. This preliminary test, using type instead of Kveraga, et al.'s graphical stimuli was done in order to investigate a different isoluminant colour combination before a full replication experiment was designed and implemented.

The following hypotheses were tested:
H_1 Words presented with high colour contrast, and low luminance contrast (P-biased, green type on cyan background) will be more difficult to detect than when presented with low colour contrast, but higher luminance contrast (M-biased : light gray type against darker gray background).
H_2 Response latency will be longer in the P-biased, than the M-biased condition.

An approximation of Kveraga et al.'s 'M' and 'P-biased' conditions was used, except that cyan/green was used for the 'P' condition. This was an attempt to resolve some of the difficulties raised by the previous discussion regarding colour isoluminance.

2.4.2 Participants

Participants (n = 50) were recruited from an undergraduate class at the University of KwaZulu-Natal in 2010. There were a total of 50 participants (41 females, 9 males).
They received course credit for attendance at the experiment, but were given the option of refusing participation, or withdrawing their data. They were required to sign a register by logging into the program, and then they could elect to do the experiment, or stop at any point. This experiment took approximately 20 minutes. The same arrangement was followed for entering an anonymous ID, and recording it so that they could identify their own results. The protocol was similar to that of Experiment 1.

2.4.3 Materials

The program was developed in Embarcadero RAD Studio (2009), with the Delphi for Win32 language. It was run on 11 workstations and the data was written to local ASCII files. The stimulus word list consisted of 100 correct words, and 100 pseudo-words. The previous experiment (1) used 50 pseudo-words and 100 correct words, but the hit rate tended to be low and the overall difficulty level was too high. The stimulus lists were therefore balanced to increase the number of targets to 50%. The correct word list was generated from a random

sample drawn from an online word list (Kelk, 2003). The maximum length of all words was 10 characters, and the minimum was 3. The pseudo-words were generated from a website (http://www.soybomb.com/tricks/words/) which creates nonsense words with legitimate English word phoneme patterns. The page generates fresh lists on demand. A list of 100 words was chosen, based on mainly on length and appearance. The experiment was dubbed 'The Continuous Recognition Test', as it is a type of 'Continuous Performance Test' commonly used to test sustained concentration (see Yang, Chung, Chen, & Chen, 2004; Inoue, Nadaoka, Oiji, Morioka, Totsuka, Kanbayashi, & Hukui, 1998). The colour definitions are illustrated below (Figure 27). The background colours for each condition are shown on the top row, and the middle row shows the font colour. The bottom row shows the appearance of the fields. Table 1 lists the colours as RGB values.

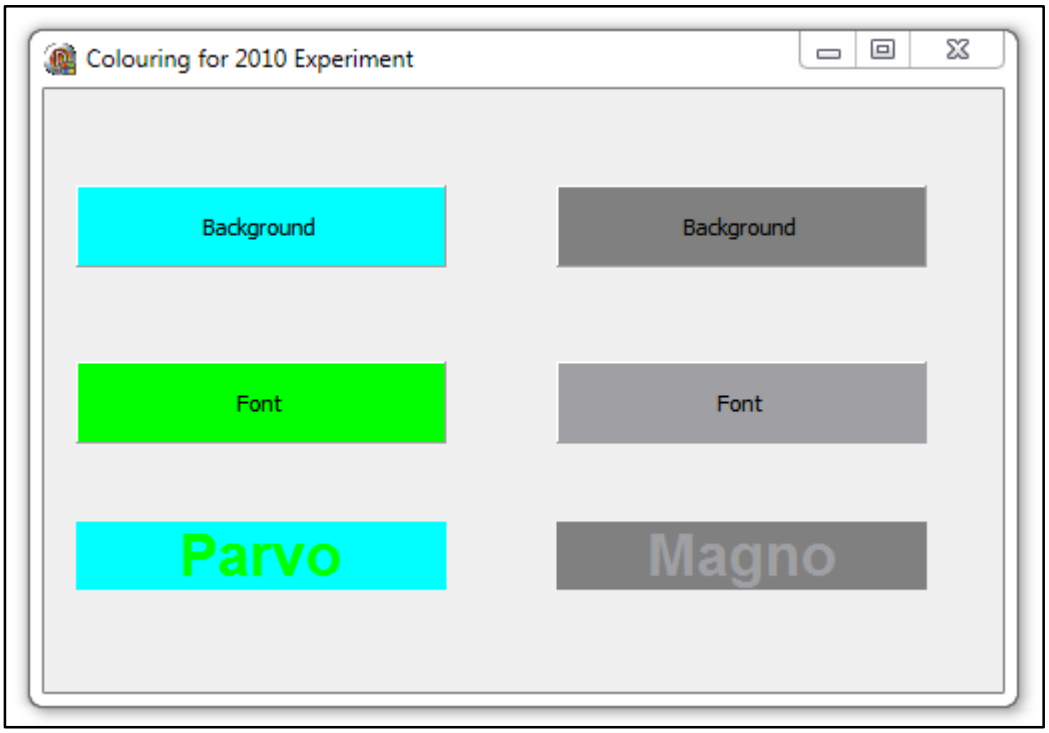


Figure 27. Illustration of Experiment 2 colour definitions. The top LH illustration ('Background') shows the colour of the word presentation. The middle row in the first column ('Font') shows the colour of the type and the bottom row of the first column shows how this was actually presented. The top RH ('Background') shows the colour of the word presentation. The middle row in the second column ('Font') shows the colour of the type and the bottom row of the second column shows how this condition was presented.

Colours for P-biased condition:
Back-ground: RGB(0,255,255)
Font colour: RGB(0,255,0)
Colours for M-biased condition:
Back-ground: RGB(128,128,128)
Font colour: RGB(160,160,160)
Font: Arial bold, 22 point.

Table 1. Experiment 2 colour definitions.

2.4.4 Procedure

The experiment consisted of 3 phases or blocks of trials: namely, a Practice Phase and two Main Phases in which the type colour and background colour were presented in random order. The login dialogue was similar to that of Experiment 1. After the information screen below was presented, an opportunity was given to decline participation at the start and at any stage of the experiment, by pressing a designated key. The sequence was the same as the previous experiment, except that the timing was altered so that the task was easier. The total trial cycle (ITI) was increased to 1200 ms in order to decrease the difficulty level, the initial word exposure time was increased to 200 ms and the mask duration was increased to 300 ms. This increased the response window to 1200 ms. Every exposure sequence was thus timed to run within a total window of 1200 ms (word exposure duration ≤200 ms; mask exposure duration = 300 ms; blank screen duration = 700 ms).

2.4.4.1 Practice Phase

A word list consisting of 100 valid, and 100 pseudo-words was shuffled for each stage. Participants were required to press a key, every time, a misspelled, or pseudo-word was seen. Keystrokes were recorded and all response latencies were measured and logged. Every time a correct response was given (i.e. to a target), a green indicator was displayed, and a red indicator for all incorrect (false alarm) responses.

1. WORD (200 ms initially)	2. Mask (300 ms)	3. Blank screen (700 ms)
shepherd	XXXXXXXXX	

For the Practice Phase, a calibration procedure was used to increase the difficulty level by decreasing the exposure time, as the participants' performance improved and conversely, to make it easier by increasing the exposure time, as performance deteriorated. The following formula was applied to calculate the word exposure time (T2) for the next two blocks (T2:=50+(LowestValue DIV 10)). The variable 'LowestValue' was the best correct response time achieved. A fraction of the best response time was added to a constant of 50 ms, to take into account practice performance, but also to maintain a reasonable degree of difficulty. Thus, if the best correct response latency was 100 ms, T2 would be 10+50=60 ms. See *Appendix 4* ('Main Program Unit') line 2235 for the calibration algorithm.

2.4.4.2 Phase 1

The same word list was used, but was re-shuffled. The condition was randomly set to either 'M' or 'P'. The exposure duration was set to T2 (see above discussion). Feedback was given for correct responses (hits) and incorrect responses (false alarms) as described for the Practice Phase.

2.4.4.3 Phase 2

The word list was re-shuffled again, and the condition was set to either 'M', or 'P', depending on the randomisation. Feedback was given for correct responses (hits) and incorrect responses (false alarms) as described for the Practice Phase. The exposure was the same as in Phase 1 (T2).

2.4.5 Results

The trial randomisation appeared to work satisfactorily. A Binomial test showed that the proportion of trials done in the M vs. P condition in Phase 2 and 3 did not deviate significantly from the expected probability of 0.5 ($p = 0.119$). In other words, the distribution was acceptably uniform. See Figure 28.

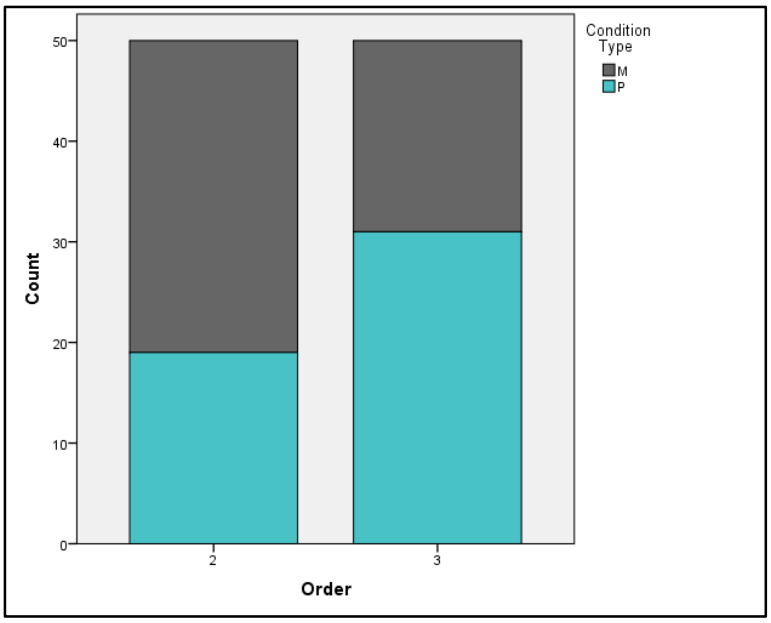


Figure 28. Frequency of presentation condition by order: Experiment 2.

One case was excluded (before the above results were tabulated). This case was excluded because of a very low hit (HR_z) rate, and false alarm (FA_z) rates for all phases (HR_z : -0.52, -0.71, -0.88) (FA_z : -1.28, -1.34, -0.77) for the Practice, Phase 1, and Phase 2, respectively. These scores stood out as being lower than those of other participants, and suggested poor engagement with the task.

Figure 29 shows the exposure times for this case in the Practice Phase and the fact that there was no improvement in performance, meant that the calibration procedure did not decrease the exposure times. The procedure was halted after 72 presentations. The Y axis is exposure time (ms).

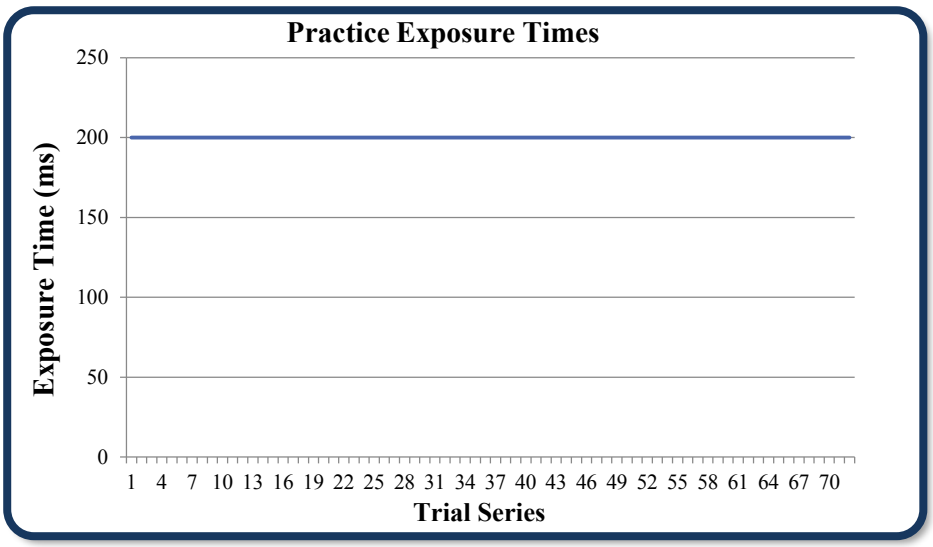


Figure 29. Exposure series for calibration procedure in Experiment 2. This participant's time did not decrease from 200 ms because no valid responses were made. This data was excluded.

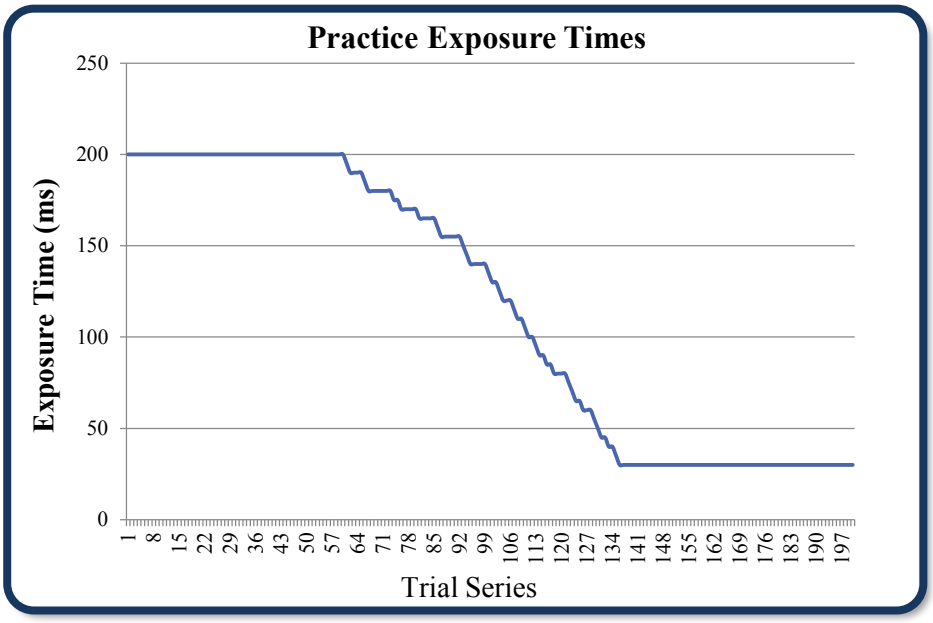


Figure 30. Exposure series for calibration procedure in Experiment 2. This plot shows a steady reduction in exposure time associated with steady and consistently improved performance.

Figure 30 shows a more typical case where there was steady improvement in performance during the practice so that the final exposure time levelled at 30 ms. This occurs when recognition performance is relatively good.

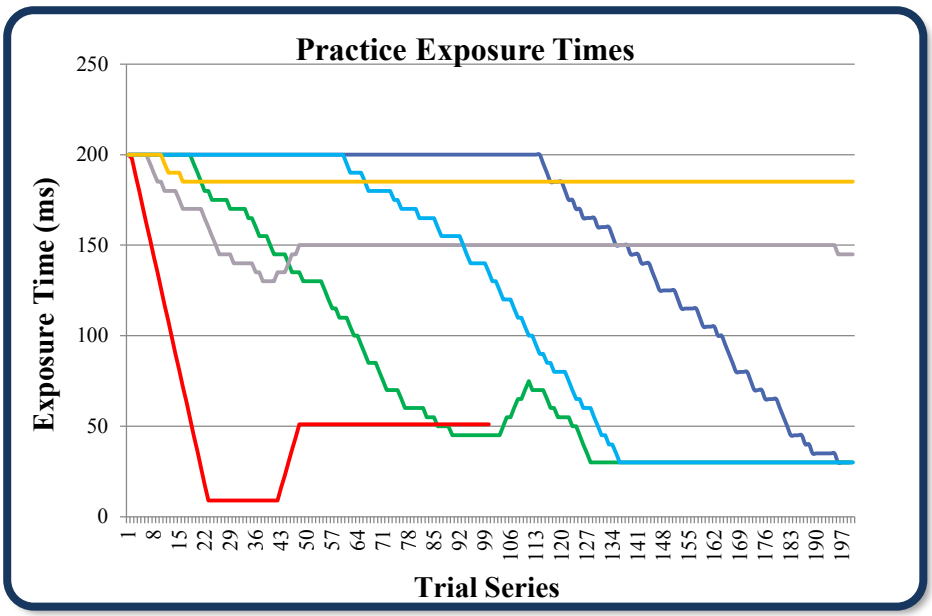


Figure 31. Various plots for calibration procedure in Experiment 2. A variety of response styles are evident: The orange series shows a very modest reduction in exposure time which does not improve. The gray plot shows better improvement, but a similar performance plateau. The red plot shows a fast and dramatic reduction in exposure time associated with excellent performance. After it dropped to just 9 ms, this participant could not sustain this level of performance and the exposure rose to a steady 51 ms. For some reason, this participant stopped the procedure. The green series shows similar, but more gradual reductions in exposure time associated with good performance. It reached 45 ms then climbed briefly to 75 ms then dropped down to 30 ms for the rest of the experiment. It is interesting to compare the green, light blue and dark blue plots which reach the same level of 30 ms with similar improvement slopes, but with differences in the point where dramatic improvements were seen.

Figure 31 shows a selection of calibration plots from the practice block. It simply illustrates various response styles and shows how exposure time was varied according to performance. There were some clear individual performance difference patterns.

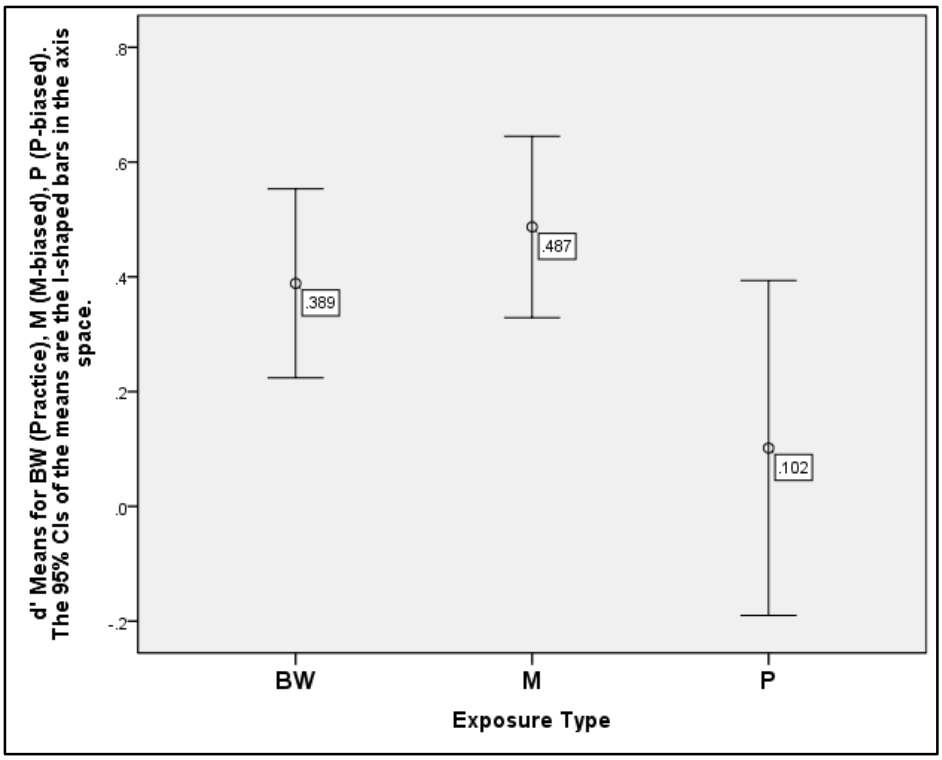


Figure 32. Experiment 2: 95% CI of d' means for Practice (BW), M-biased (M) and P-biased (P) conditions.

Figure 32 shows that d' for the Practice Phase (BW / high contrast) where words were displayed in black and white ($\bar{x}$ = 0.39, SD = 049), is at a comparable level to that of the M-biased condition ($\bar{x}$ = 0.49, SD = 0.5), in spite of the fact that the words were seen at shorter exposure durations ($\bar{x}$ = 62.3 ms, SD = 7.66 vs. calibration means). Performance in the P-biased condition is noticeably poorer ($\bar{x}$ = 0.1, SD = 0.79).

Results of a Univariate Analysis of Variance (ANOVA) show a main effect for the Exposure Type (Colour) condition when d' for P- and M-conditions is compared, $F(1, 148) = 3.8$, $p = 0.025$ (with the Bonferroni correction for multiple *post hoc* comparisons the mean difference is 0.32, $p = 0.031$). There is no main effect for the Order of exposure, $F(1, 148) = 0.06$, $p = 0.8$, nor is there any interaction between Order and Exposure Type, $F(1, 148) = 0.91$, $p = 0.34$. See Figure 33.

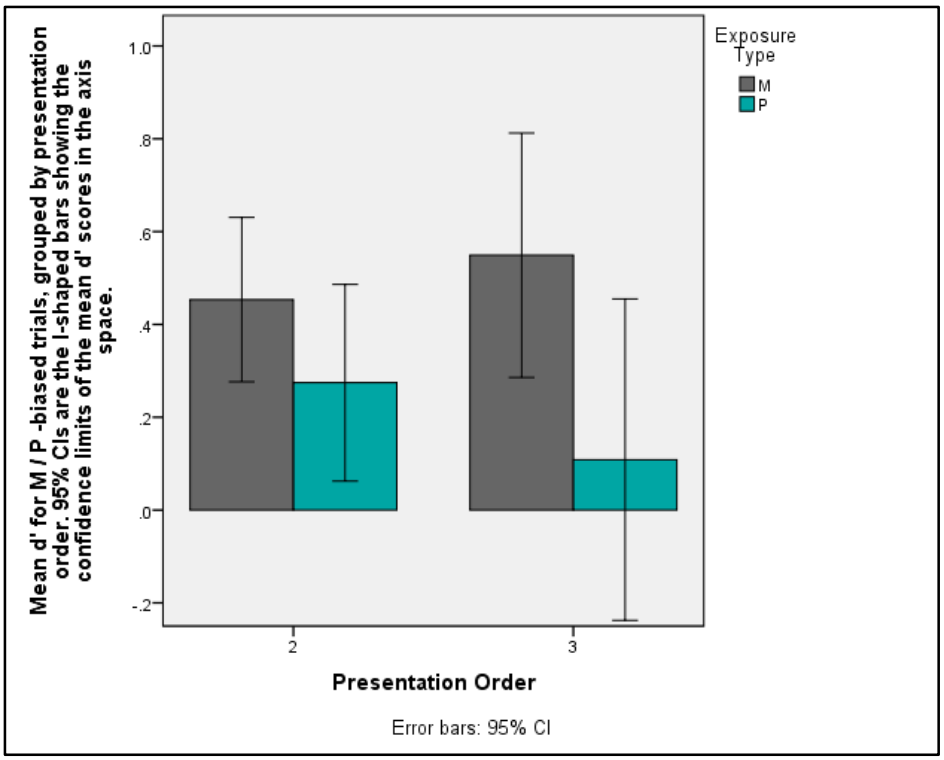


Figure 33. Experiment 2: Order by Exposure Type – error bars show the 95% CI of mean d'.

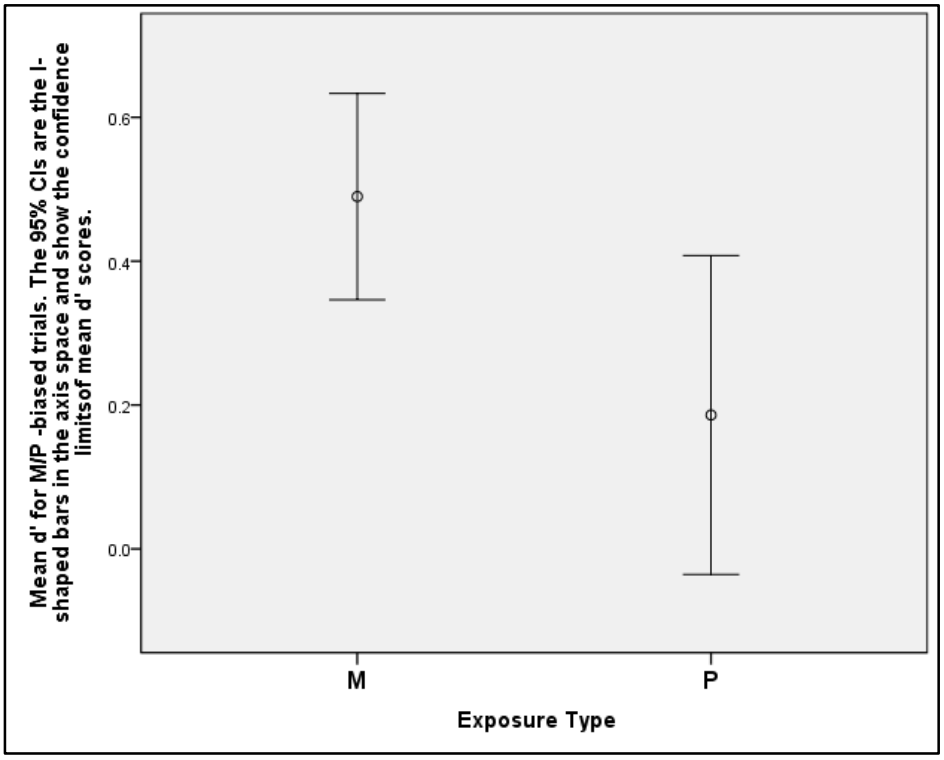


Figure 34. Experiment 2: Error bars show the 95% CI of d' means by Exposure Type (M- and P-biased only).

Figure 34 shows a comparison of the mean between the M- and P-biased conditions. There is some minor overlap between the two confidence intervals, athough the ANOVA showed a significant difference. The distribution of d' scores did not deviate from normality for either the M, or Practice condition. However, the distribution of scores in the P has a strong negative skew (Kolmogorov-Smirnov (50) = 0.26, p <0.0005). This suggested that participants' ability to detect type in this condition had a non-normal distribution (see 'Score Distribution Plots.pdf' in Appendix 1 and 'Appendix 1.pdf' (media disk or Dropbox folder).

No data transformation was helpful in dealing with this distribution problem. Log transformations cannot deal with negative values and replaced them with nulls. Other transformations, such as using the exponent exacerbated the problems. Although the ANOVA model described previously, demonstrates a significant difference between the M- and P-biased conditions (controlling for *post hoc* comparisons), because of the distribution problem, several non-parametric tests were run. Firstly, an independent samples Mann-Whitney test indicated a significant difference between the groups, Z = -2.45, p = 0.014 (2-tailed), Secondly, the Two-Sample Kolmogorov-Smirnov Test. This also showed a significant difference: Kolmogorov-Smirnov Z = 1.4, p =0.04 (Asymptotic, 2-tailed).

The fact that different individual variations and performance patterns were seen in the calibration trials in the Practice Phase has been illustrated – see Figure 29 - 31. The mean exposure time was 62.26 (SD = 7.66). However, there are interesting patterns in the distribution of the data. The distribution appears to be multi-modal - see Figure 35.

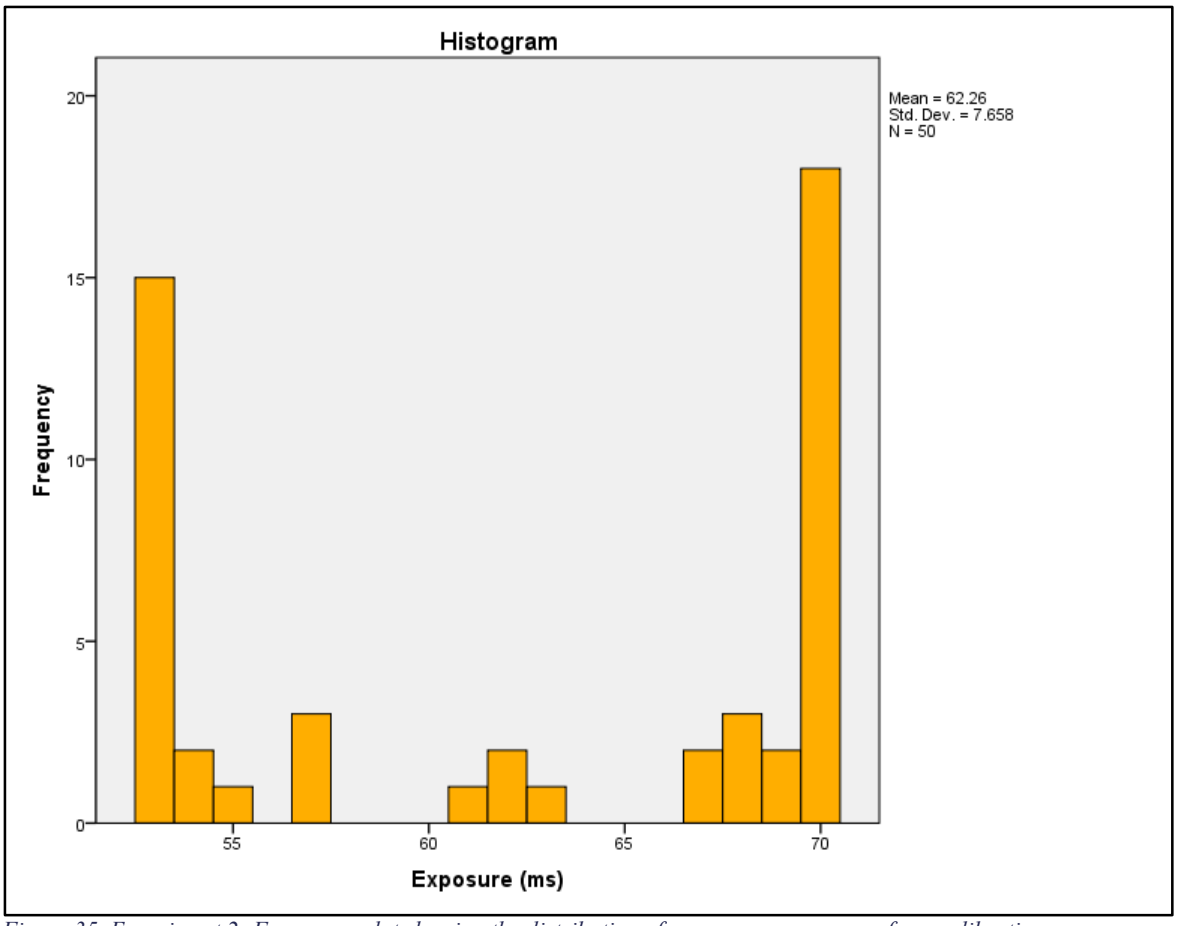


Figure 35. Experiment 2: Frequency plot showing the distribution of mean exposure scores from calibration.

The majority of participants achieved scores of 53, or 70. In order to get a score of 53, the minimum exposure time with a correct response is 30 ms (since the formula adds 30 DIV[1] 10 to a constant of 50). In order to get a score of 70, the minimum exposure time would have been 200 ms (200 DIV 10 = 20; 20 + 50 = 70). This is probably a bi-modal distribution, since the two groups make up 66% of the sample. This may signify different levels of English language familiarity.

It is interesting to note, in the M-biased condition, the median was 0.507 (50% of the group had d' scores of <0.51), whereas in the P-biased condition, the median was 0.176 thus for this condition, 78% had d' scores of <0.5. In the Practice Phase (BW/high contrast), the median was 0.43. In general, the d' scores tended to be low, which means that the task was relatively difficult. A d' score of near zero means that detection is at the level of chance, and one can only infer reasonable detection accuracy at the level of about 1 (this signifies fair

[1] The Pascal/Delphi, DIV operator used for integer division, returns the quotient and integer remainder.

discrimination of targets from noise and non-targets since d' score and Z-scores are equivalent and a Z-score of 1 is equivalent to one standard deviation above the mean). Response latencies were similar between the M- and P-biased conditions, but there was a consistent difference between target and error latencies in all conditions (Figure 36).

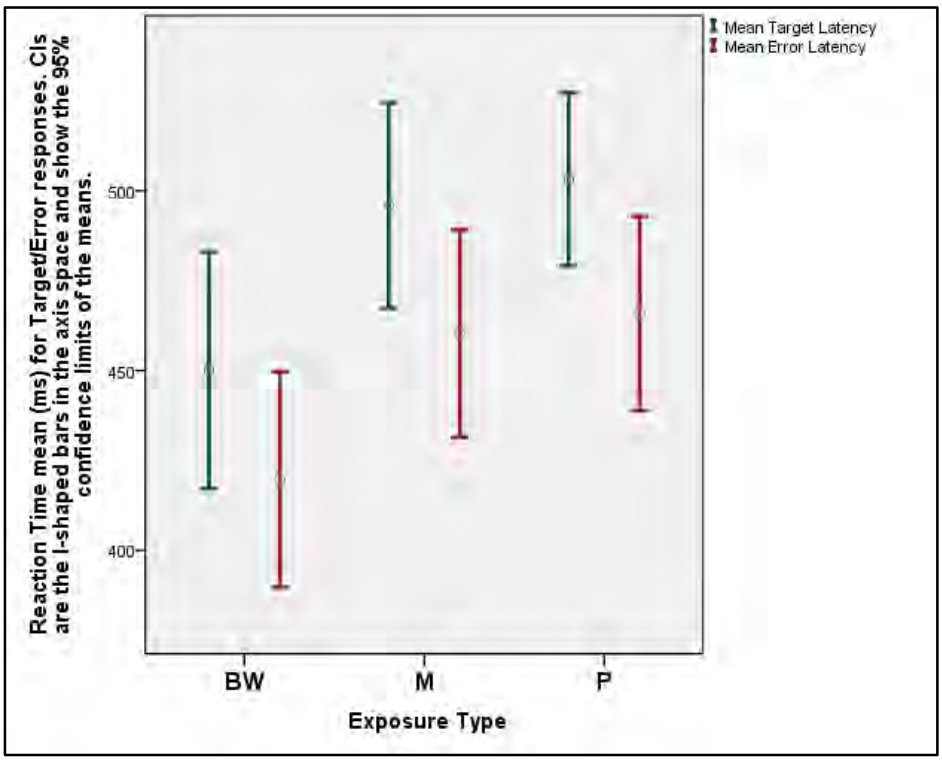


Figure 36. Experiment 2: 95% CI of mean target and error latencies (i.e. reaction times).

There was also a difference between the Practice (BW) and M/P phases, with the former showing shorter target and error latencies. A Oneway ANOVA indicated that there are group differences for both target, and error latencies, $F(2, 148) = 4.044$, $p = 0.02$ (target latencies); $F(2, 148) = 3.12$, $p = 0.047$ (error latencies). The test for homogeneity of variances was non-significant for both (Levene statistic (2, 148) = 2.6, $p = 0.08$; (2, 148) = 3.12, $p = 0.61$) respectively. The plots of the mean target latencies and error latencies are shown in Figure 37 and 38 respectively.

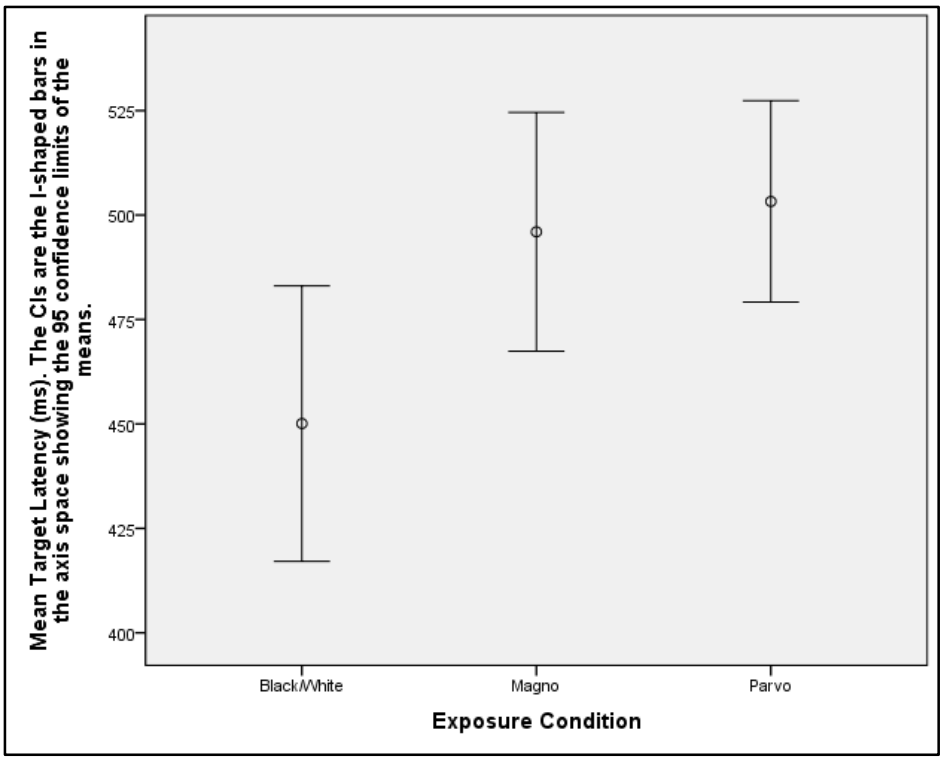


Figure 37. Experiment 2: mean target latencies for different exposure conditions.

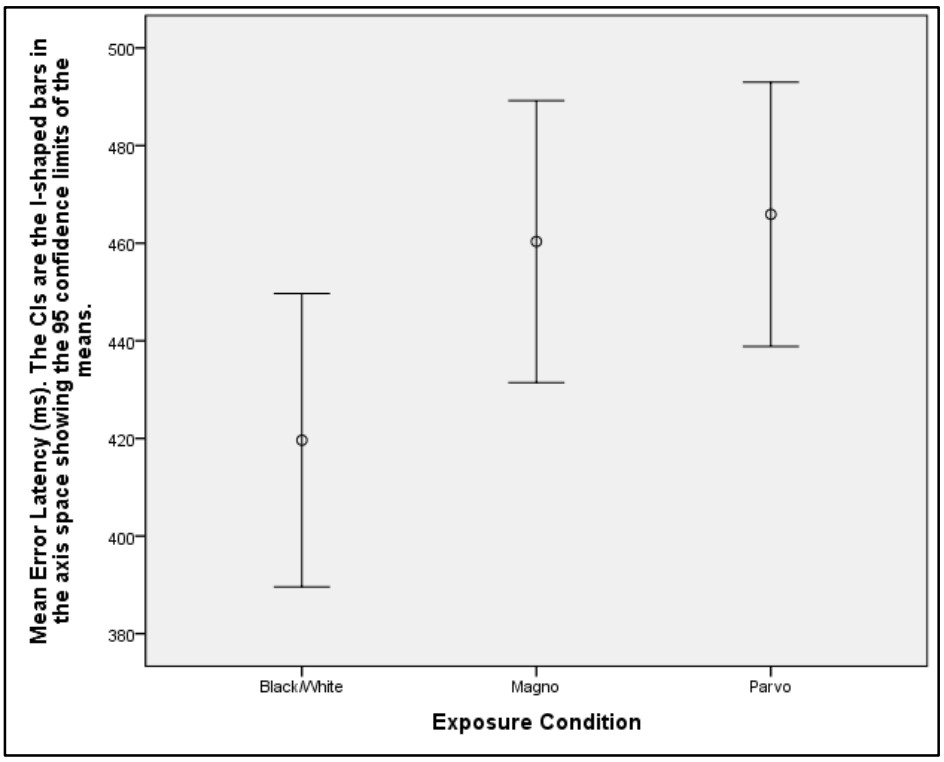


Figure 38. Experiment 2: mean error latencies for different exposure conditions.

A General Linear Model with two dependent variables (mean target latency, mean error latency) and one independent variable (Exposure condition) was run with *Post hoc* tests which compared the three levels of the independent variable (Exposure condition : Black/White, Magno, Parvo) with the Bonferroni correction for each comparison. The overall

model is non-significant (Wilks' Lambda = 0.948, $F(4,290)$, $p = 0.1$). There was a significant difference between the Practice (Black/White) and Parvo conditions for the target latencies (mean difference = -52.86, $p = 0.03$. A comparison of the Practice (Black/White), and Magno conditions is non-significant. None of the error latency comparisons was significant. As the multivariate test is non-significant, there is a reasonable probability that the comparison referred to as significant is due to chance. The most potentially interesting finding is of a significant difference between the Black/White and Parvo conditions for mean target latency.

2.4.6 Discussion

The experiment shows a difference in word detection levels between the M and P conditions. The detection accuracy between the Practice (black type on white back-ground), and the M (light gray type against darker gray back-ground) was similar, although there is a possibility of longer response latencies for the M-biased condition. This may be due to the shorter exposure times enforced in the practice condition. Although the P-biased condition involved more colour contrast, performance suggested that words in this colour condition are more difficult to detect accurately.

Gazzaniga (2005) estimated the maximum colour sensitivity of rods as 495 nm, and falling somewhere between that of the blue cones (430 nm), and green cones (530 nm). It would make theoretical sense if the M system, which receives significant rod input, is not able to discriminate colours (i.e. coding these wavelengths as luminance differences) in this region of the spectrum. This makes sense in terms of the drawings published by Livingstone, 1988 (see Figure 10, Figure 21). It also seems to reflect the subjective difficulties of trying to read pure green type against a cyan (blue/green) background.

The task was designed to be difficult and this is reflected in modest mean d' scores of 0.43, 0.49, 0.17 for the B/W, M and P phases respectively. These are not robust scores as a score of 0 signifies random responding and it is only when d' approaches ≥1 that one can have some confidence that detection accuracy is reasonable. Since a d' score is equivalent to a Z-score, a d' score of ≥ 1 is equivalent to one standard deviation about the mean (zero) and scores in this region are unlikely to occur due to chance. The experiment was designed to test detection accuracy in difficult conditions, however.

Some of the words were probably too difficult and there were a number of participants whose first language was not English. Testing word familiarity was not part of the protocol. However, it appears that the experiment was a success in that it showed increased difficulty for word detection accuracy in the P condition, compared with the M condition which presented words with relatively low luminance contrast. The problem of establishing appropriate luminance contrast levels is addressed in a later experiment.

2.5 Experiment 3: Exploring Object Recognition

2.5.1 Introduction

The objective of Experiment 3 was to test the colour combinations used in Experiment 2 for line drawings instead of type. A secondary question concerned the size at which these would be recognisable, and it was hypothesised that the M-biased presentations would be recognised at a smaller size and shorter latency than the P-biased presentations (see Figure 20). Since the M system is believed to have poorer spatial resolution than the P system, the P-biased presentations were expected to further inhibit M system function and lead to longer response latencies when the drawings were at a smaller scale.

2.5.2 Participants

The same group of participants from the previous experiment (Experiment 2) participated (n = 57: 43 females, 14 males). Age was not recorded. The same protocol was followed and participants used the same (anonymous) ID. They were informed that they could cease participation if they found it taxing, or unpleasant in any way and there would be no prejudice against them. The only requirement is that they understood the experiment, so that they could elect to write a brief report on either this, or the previous one. A full explanation was given at a scheduled lecture, and any queries were dealt with by the researcher. The experiment took approximately 20 minutes, including the briefing and debriefing procedure.

2.5.3 Materials

The program software was developed by D. Mansfield with Embarcadero RAD Studio (2009), using the Delphi language and compiled to native Win32 executable code. The experiment was run on the same 11 workstations, and this experiment was the second in the series. Data was written to ASCII files on the local hard drives of each work station.

The pictures were drawn from a Word Perfect X3 clip art library. Figures 39 – 40 are examples of the pictures presented in the Practice Phase.

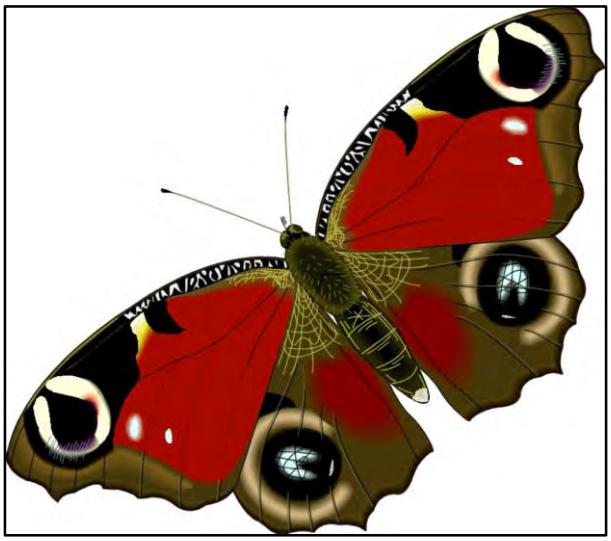


Figure 39. Experiment 3: sample item 1.

Figure 40. Experiment 3: sample item 2.

Colours for P-biased condition: Back-ground colour: RGB(0,255,0); Foreground colour: RGB(0,255,255).
Colours for M-biased condition: Back-ground colour: RGB(119,119,119); Foreground colour: RGB(145,145,145).
Surround back-ground colour: RGB(192,192,192).

Table 2. Experiment 3: colour definitions.

Since this experiment was a prelude to the main replication, the values of the isoluminant colours had not been established but the modelling presented in Section 2.3 was a source of hypotheses.

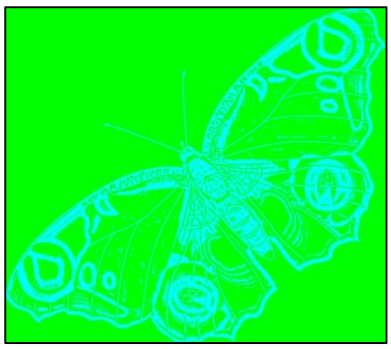


Figure 41. Experiment 3: example of parvo-biased image.

Figure 42. Experiment 3: example of magno-biased image.

2.5.4 Procedure

The experimental protocol was the same as the previous one except that line drawings were presented, instead of type. The same colouring (back-ground/fore-ground) combinations were used. A fixation '+' was displayed for 100 ms, the picture, then the keyboard was enabled, and images were presented initially at a very small size, (15 pixels – proportionally, so that the biggest dimension would be 15, and the smallest in proportion), then scaled every 5 ms by enlarging it by 4 pixels (maintaining the proportion) to a maximum of 300 pixels in width, or

height. The total trial duration including the fixation, was 1600 ms. The experiment consisted of a Practice Phase in which colour pictures were presented in random order, and two Main Phases in which M and P – biased pictures were presented in random order. In each phase, Participants were instructed to press a key if the object in the picture (in real life) was bigger than a shoe-box, see Appendix 1, 'Appendix 1.pdf', 'Experiment 3' to view screens. Although it may have been useful to use the same (ITI) timing value as Kveraga *et al.* (2007b), the current experiment was also concerned with possible differences in spatial frequency resolution between the M/P – systems and the size at which items would be recognized in the different conditions. The ITI was arranged to make the task brisk and to maximize participants' attention. Since the zooming introduced minor delays which might have allowed lapses in attention, the second fixation screen was not implemented.

A total of 40 targets (bigger than a shoe box) and 31 non-targets was presented (smaller than a shoe box) in a task modelled after that of Kveraga, *et al.* (2007b) where participants were asked to press a key whenever they saw an object presented that was bigger than a shoe box. It was intended that the target/non-target ratio would be the same as in Kveraga, *et al.* (2007b) but only 31 suitable non-targets could be found at the time the experiment was run. The same pictures were presented for all 3 phases, but in different colours. They were shuffled before each presentation, so that their order varied. The ordering of the targets and non-targets also varied.

At the end of each block results were recorded to ASCII files, with all performance data, and a summary which showed the results, such as d', hit rates, misses, false alarm rates, etc. Another file recorded specific information for each response, including the time, stimulus name, response latency, number of zooming iterations, and the width, height, and area of the picture when the response key was pressed. Phases 1 and 2 were randomly ordered.

2.5.5 Results

The randomisation of the trial ordering was satisfactory. A Binomial test showed a non-significant distribution of trials between the two conditions, $p = 0.185$ (Exact, 2-tailed test) - see Figure 43.

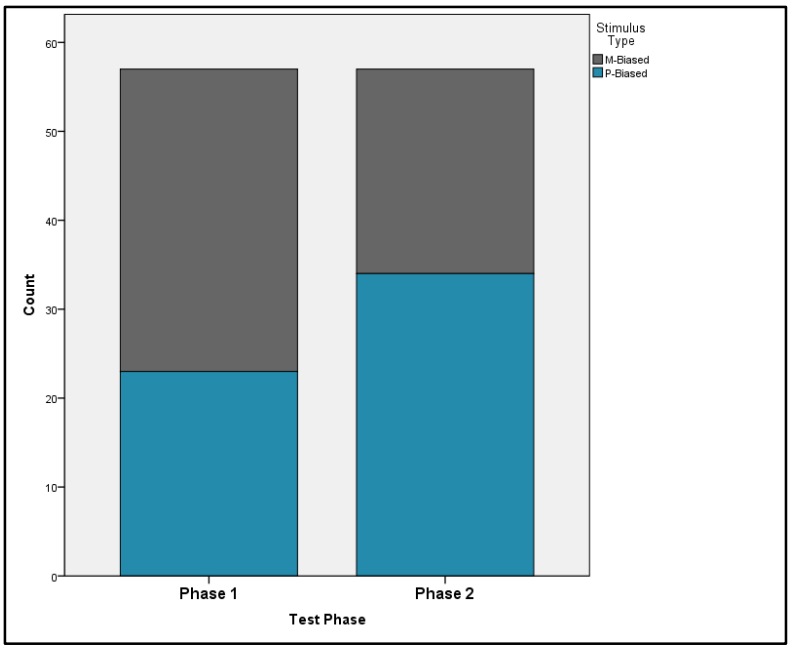


Figure 43. Experiment 3: frequency of M and P trials by presentation order.

There was a main effect for stimulus type across the 3 conditions - an ANOVA test showed a significant difference between the 3 conditions, $F(2, 165) = 6.01$, $p = 0.003$ (Figure 44 shows the 95% confidence interval of the d' means. This plot suggests that performance in M-biased vs. P-biased trials was similar (Figure 44). Performance in the M- and P-biased blocks appears to be better than in the Practice Phase (note that the lower (95%) confidence limits of the d' mean for M- and P-biased trials do not overlap with the upper limit of the d' mean for the Practice trials).

There appeared to be slight differences for target latency, and error latency (Figures 45 – 46). There also appeared to be patterns of differences associated with gender (Figures 47, 48). It is interesting to note that females showed better performance in the M-biased condition, and similar levels of performance in the Colour, and P-biased conditions. This was associated with similar target response latencies in the M condition and the error latencies were very similar for males and females in the M-biased and P-biased conditions. The fact that females had similar levels of performance to males in the Colour condition but their response and error latencies were longer, suggests they had more difficulty with this particular task. Males had target latencies in the order of 150 ms faster. However, females outperformed males quite dramatically in the M condition, although they had similar target and error latencies to males.

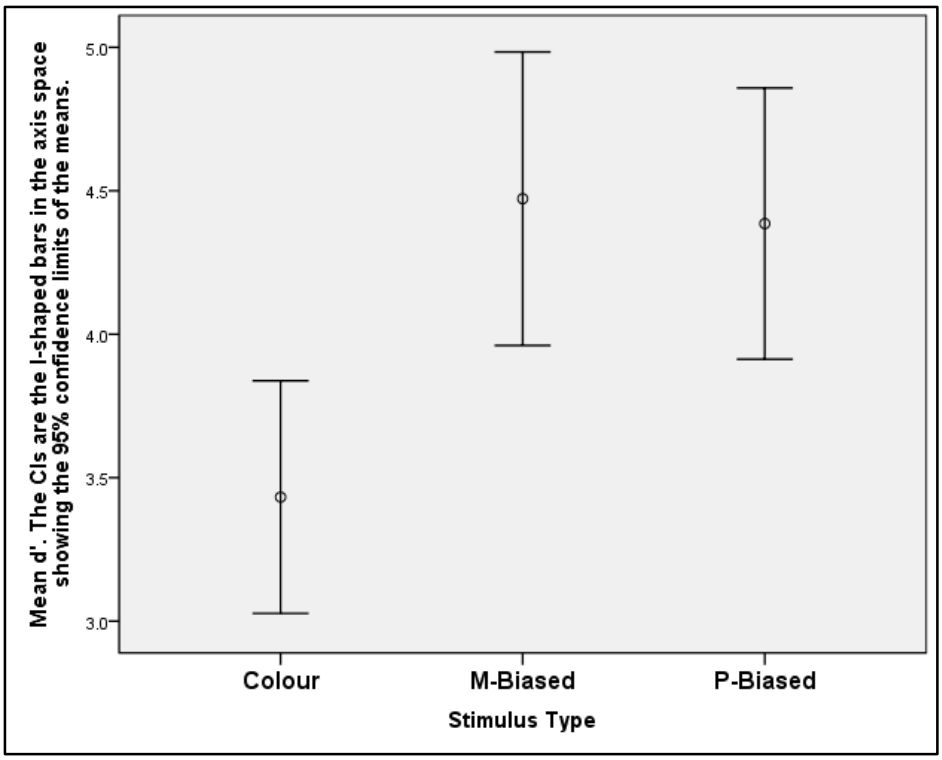


Figure 44. Experiment 3: Error bars show the 95% CI of d' for the various block means.

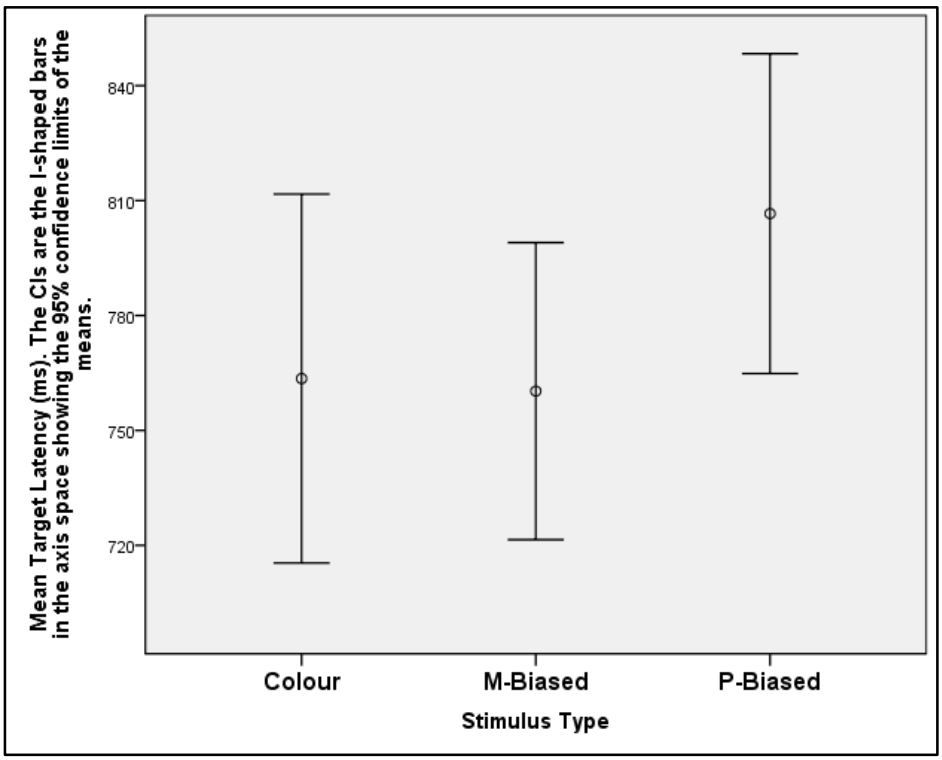


Figure 45. Experiment 3: Error bars show the 95% CI of target latency block means.

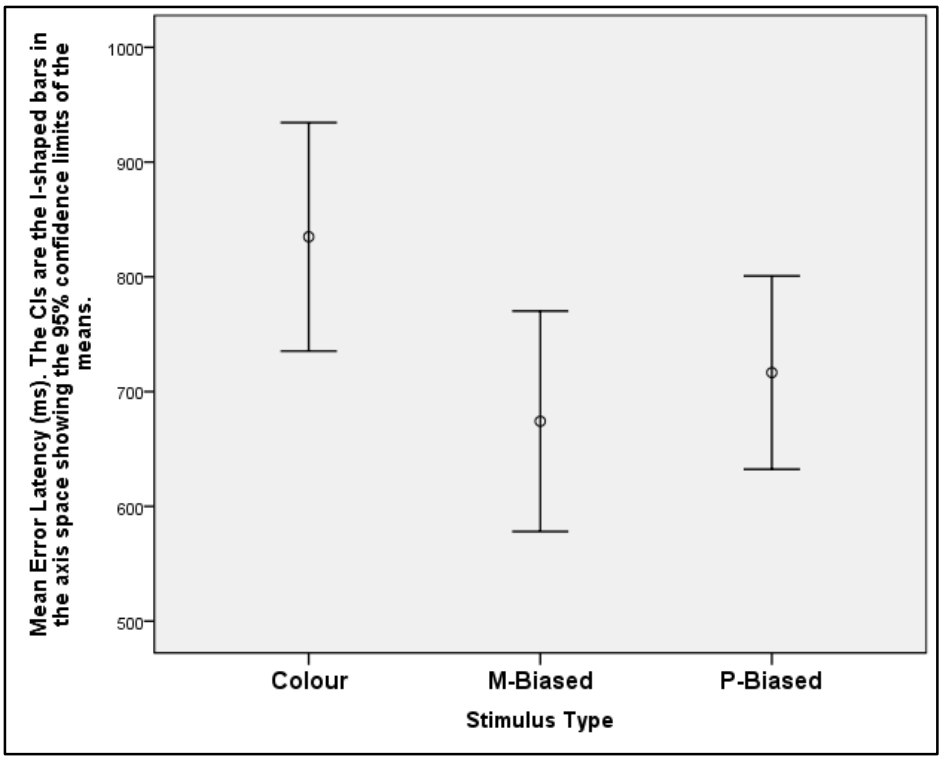


Figure 46. Experiment 3: Error bars show the 95% CI of error latency block means.

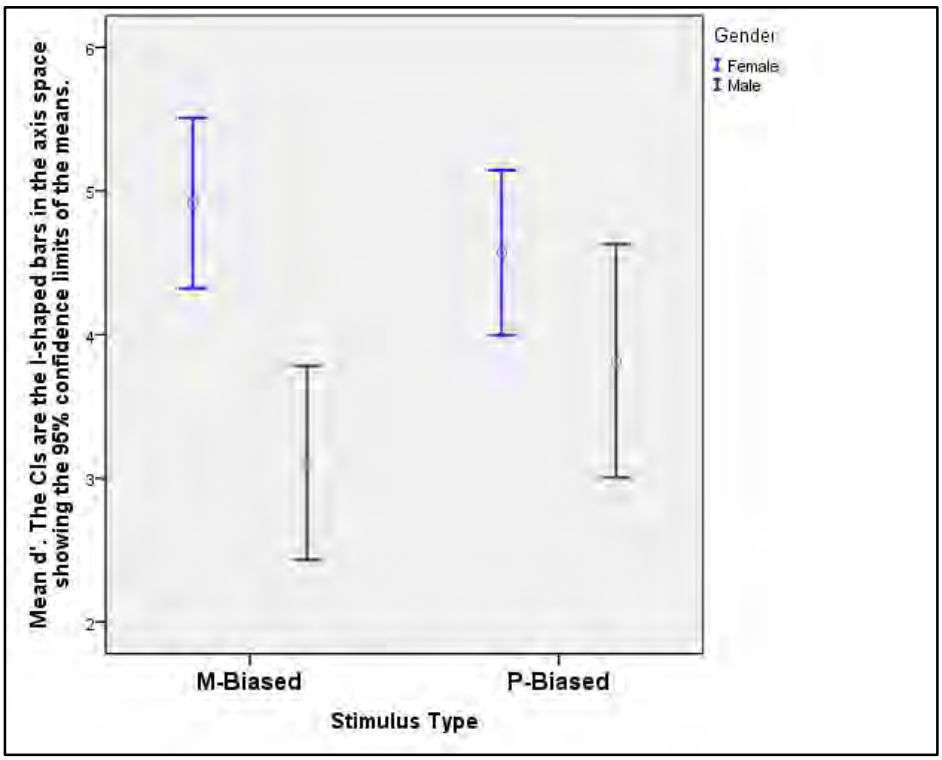


Figure 47. Experiment 3: Gender by Stimulus type: Error bars show the 95% CI of mean d' for each block and group.

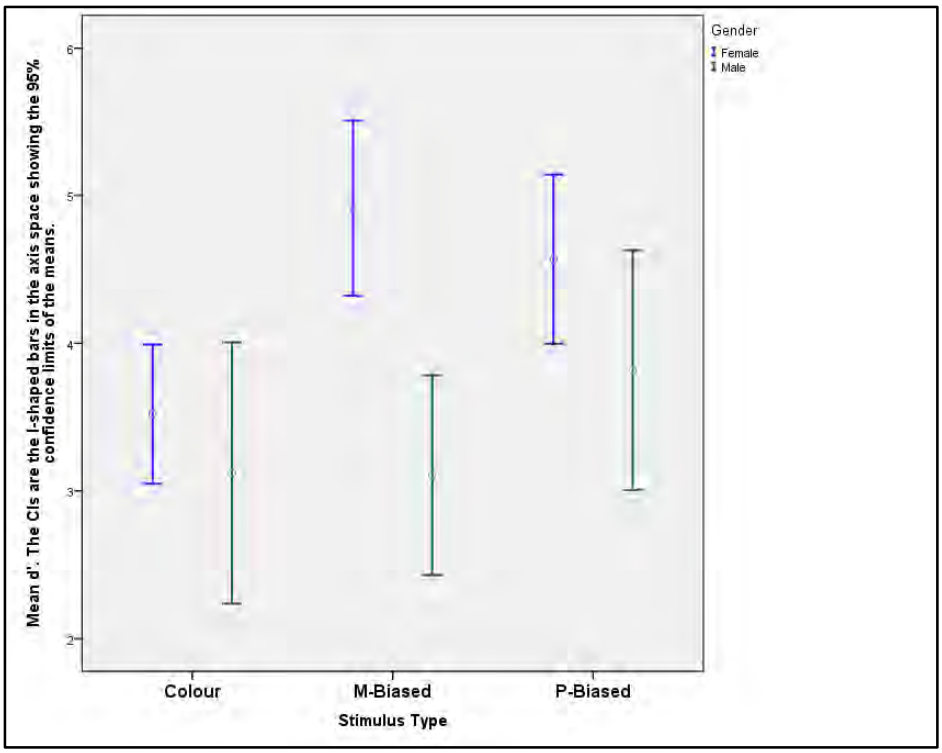


Figure 48. Experiment 3: Gender by Stimulus type (colour/M/P). Error bars show 95% CI of mean d' for all conditions.

Although these patterns appear interesting, an interesting feature needs to be described and discussed. The distribution of the d' scores, for all conditions, is not normal. It appears to be a bi-modal distribution: see Figures 49 – 52.

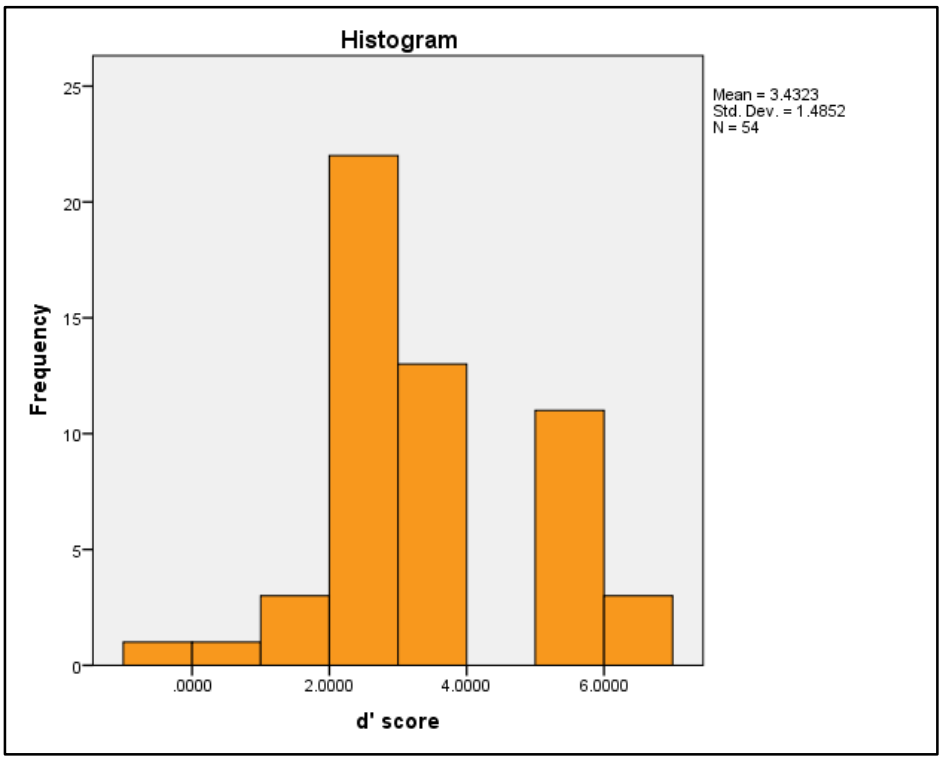


Figure 49. Experiment 3: distribution of d' for colour (practice) condition.

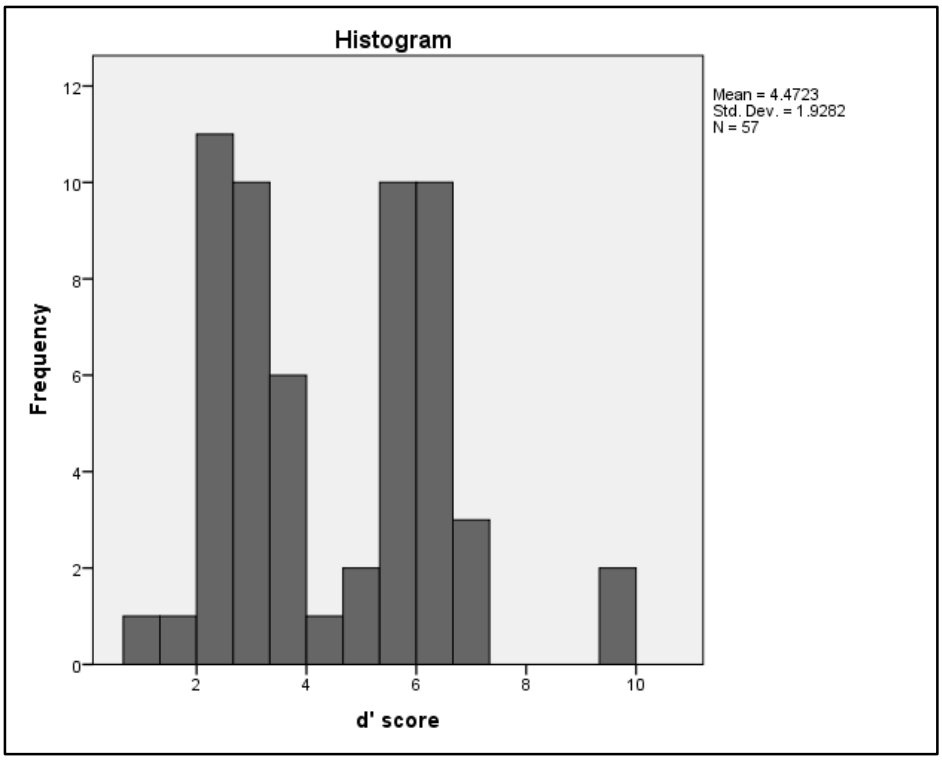


Figure 50. Experiment 3: distribution of d' for M-biased condition.

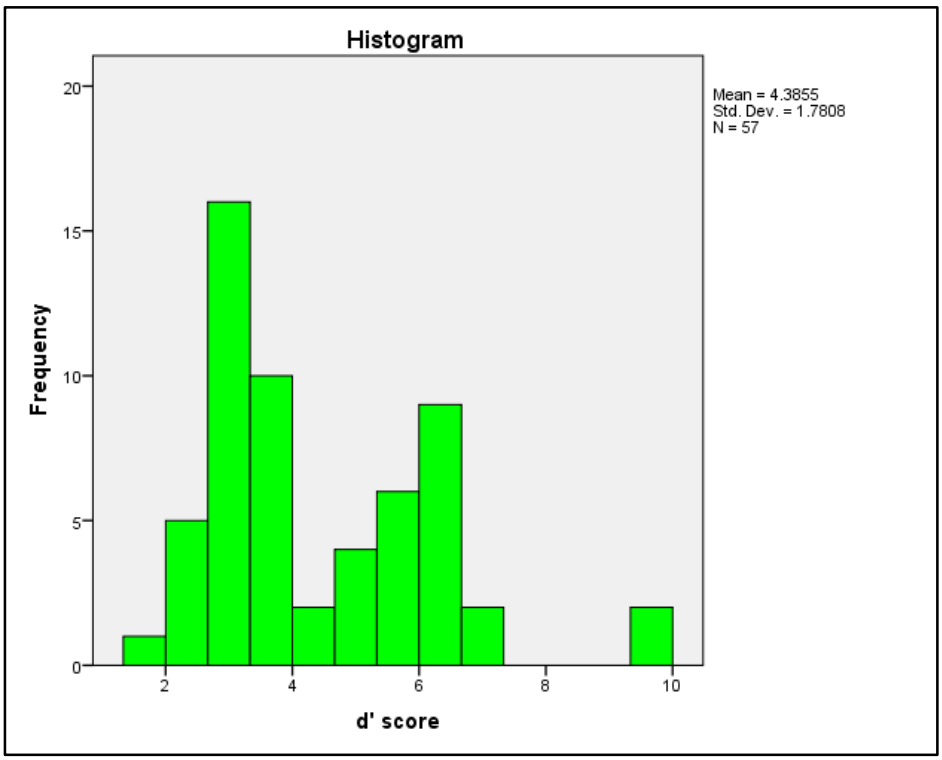


Figure 51. Experiment 3: distribution of d' for P-biased condition.

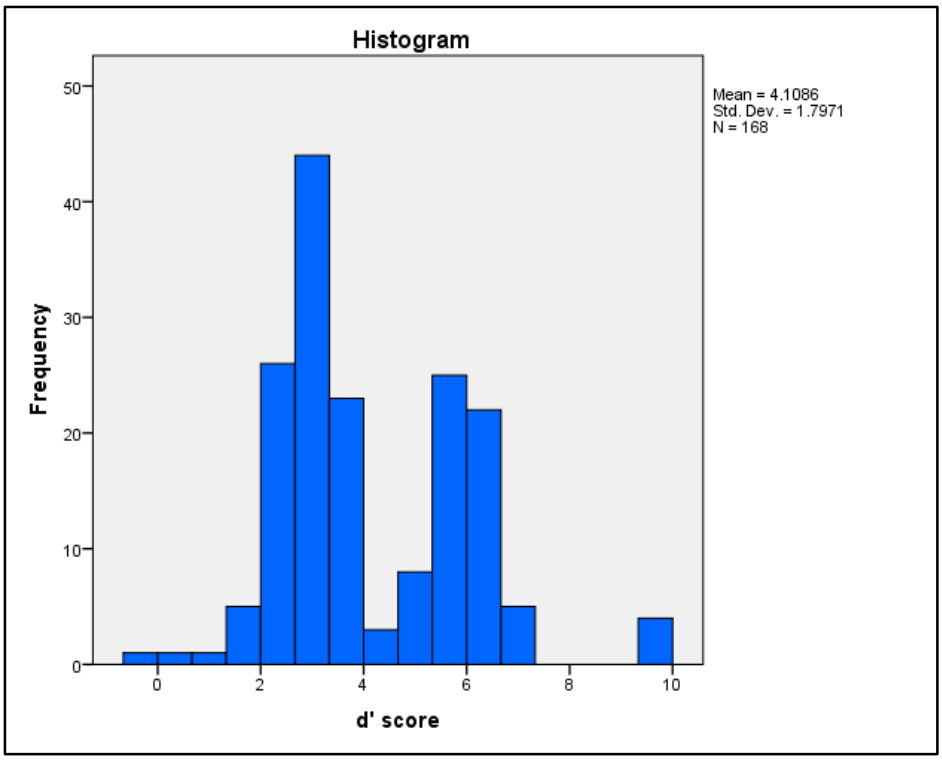


Figure 52. Experiment 3: distribution of d' for all conditions.

Part of the reason for the unusual distribution of scores is that females generally outperformed males. When the scores for all conditions were categorized as low/high with the criterion of <4 / > 4 for d' respectively, the female group had a relatively higher count of scores in the 'higher' group than the male group: Fisher's Exact test: $p = 0.04$ (exact sig., 2-sided). This tendency was even more marked for the M-biased condition: Fisher's Exact test: $p = 0.005$ (exact sig., 2-sided) (see Figure 53).

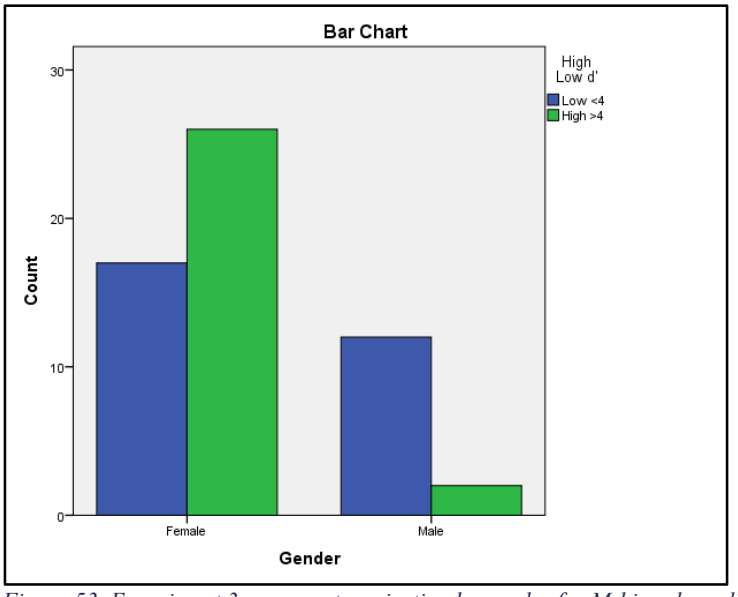


Figure 53. Experiment 3: score categorisation by gender for M-biased condition.

Indeed, none of these distributions was normal, according to the Kolmogorov-Smirnov tests of normality (all tests were significant, $p < 0.05$). It is not easy to transform the d' scores since d' values are often negative (negative log values are imaginary) so it seemed better to use non-parametric tests than add a constant to all d' values. A Mann-Whitney test indicated significant differences for all dependent variables (Mean Target Latency, Mean Error Latency, d') with Gender as the grouping variable – see Table 3:

Table 3. Experiment 3: Mann-Whitney test with grouping variable Gender.

Test Statistics[a]

	Mean Target Latency	Mean Error Latency	D Prime
Mann-Whitney U	1625.500	1897.000	1597.500
Wilcoxon W	2445.500	2717.000	2417.500
Z	-3.480	-2.470	-3.586
Asymp. Sig. (2-tailed)	.001	.014	<.0005
Exact Sig. (2-tailed)	<.0005	.013	<.0005
Exact Sig. (1-tailed)	<.0005	.007	<.0005
Point Probability,	<.0005	<.0005	<.0005

a. Grouping Variable: Gender

This test shows significant group differences across all colour conditions associated with Gender.

Table 4. Experiment 3: Kruskal-Wallis test with grouping variable Stimulus Type.

Test Statistics[b,c]

			Mean Target Latency	Mean Error Latency	D Prime
Chi-Square			2.921	4.274	10.634
df			2	2	2
Asymp. Sig.			.232	.118	.005
Monte Carlo Sig.	Sig.		.231[a]	.119[a]	.005[a]
	99% Confidence Interval	Lower Bound	.220	.110	.003
		Upper Bound	.241	.127	.006

a. Based on 10000 sampled tables with starting seed 2000000.

b. Kruskal Wallis Test

c. Grouping Variable: Stimulus Type

A Kruskal-Wallis test was done using the Monte Carlo re-sampling method, with a confidence interval of 99%, using 10000 samples. Monte Carlo resampling was used as it gives an unbiased estimate of the exact significance level. *Stimulus Type* refers to the colour condition in which stimuli were presented.

Although the comparisons show significant group differences, the analysis is unrevealing as there is a trend which suggests group differences in target and error latencies associated with stimulus type (see Figures 45, 46). The data for each participant consists of summaries like the example in Table 5. The Mean Target Latency, and Mean Error Latency scores are calculated for each individual, and this tends to blur more subtle patterns, although it may mask other problems, such as cases with skewed data. To get a more fine-grained picture and increase statistical power, all individual responses were pooled and reanalyzed.

Table 5. Experiment 3: individual result summary.

BEGIN TEST 2010/04/28 04:36:05 PM
ID: 103142 Gender: M Age: 21
Object Discrimination Task
Score: 37 Targets: 40 Non Targets: 31 Errors: 16
Mean Target Latency: 520 Std Targets: 192.536 Std Errors: 196.812 Mean Error Latency: 495
END TEST 2010/04/28 04:38:10 PM
Actual Practice Time: 00:02:01
Timer 1: 10 Timer 2: 200 Timer 3: 500
Hits: 37 Targets: 40 Hit Rate Z: 1.4395 False Alarms: 0 Non-Targets: 31 False Alarms Z: 0.0404
Hit Rate: 0.925 False Alarm Rate: 0.516 D Prime for Practice: 1.3991

BEGIN TEST Phase 1 2010/04/28 04:38:20 PM
ID: 103142 Gender: M Age: 21
Object Discrimination Task
Score: 40 Targets: 40 Non Targets: 31 Errors: 17
Mean Target Latency: 645 Std Targets: 207.605 Std Errors: 169.034 Mean Error Latency: 552
END TEST PHASE 1 2010/04/28 04:40:25 PM
Actual Testing Time (Phase 1): 00:02:04
Colour: ParvoCG Mask used: enabled- Timer 1: 10 Timer 2: 200 Timer 3: 500
Hits: 40 Targets: 40 Hit Rate Z: 4.7534 False Alarms: 17 Non-Targets: 31 False Alarms Z: 0.1216
Hit Rate: 1.000 False Alarm Rate 0.548 D Prime for Phase 1: 4.6

BEGIN TEST Phase 2 2010/04/28 04:40:39 PM
ID: 103142 Gender: M Age: 21
Object Discrimination Task
Score: 38 Targets: 40 Non Targets: 31 Errors: 12
Mean Target Latency: 658 Std Targets: 159.255 Std Errors: 250.533 Mean Error Latency: 671
END TEST PHASE 2 2010/04/28 04:42:55 PM
Actual Testing Time (Phase 2): 00:02:11
Colour 318: Magno Mask used: enabled- Timer 1: 10 Timer 2: 200 Timer 3: 500
Hits: 38 Targets: 40 Hit Rate Z: 1.6449 False Alarms: 12 Non-Targets: 31 False Alarms Z: -0.2869
Hit Rate: 0.950 False Alarm Rate 0.387 D Prime for Phase 2: 1.9317

Merging individual output files resulted in a data file with 7359 records containing all participants' responses, with several other parameters. Figure 54 is a plot of one individual's response latencies. The Y-axis shows response latencies in ms, and the X-axis shows the trial series times.

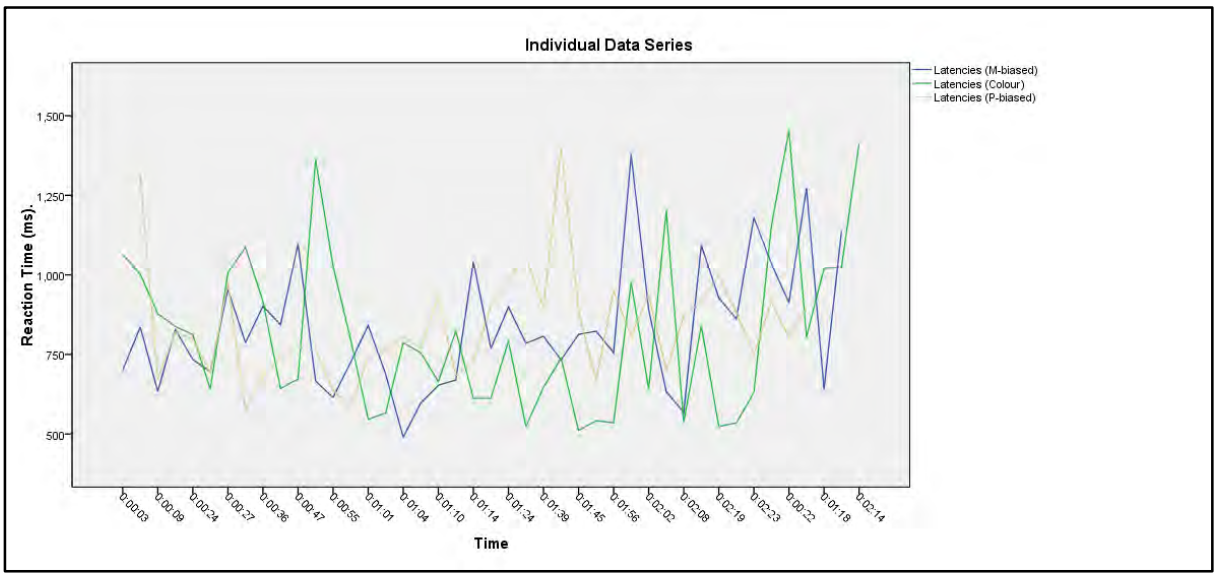


Figure 54. Experiment 3: Plot of response latencies for 3 conditions for one participant – see legend for details.

The correlation between two methods of measuring response latency was good[2] (Pearson Correlation r =0.97, p (two-tailed) <0.001.

Figure 55 shows the 95% confidence interval of the mean latency for stimulus type (targets/non-targets) grouped by the different colour conditions (phase). It shows that non-target response latencies were at similar levels, and tended to be faster than for targets (± 700 ms). The Practice phase (coloured images) means look similar to those for the M-biased presentations. Latency means for targets in the P-biased condition appear longer than for M/Colour means by about 45 – 50 ms. This is illustrated slightly differently in Figure 56, where the means are grouped by Stimulus Type on the X-axis.

A General Linear Model using the aggregated data set with gender as the IV and mean target latency, mean error latency as DVs shows that there is a significant difference for mean target latency, but not mean error latency $F(1) = 10.84$, $p = 0.001$; Wilks' Lambda = 0.94, $F(2, 165) = 5.44$, p <0.001.

[2] This was a concern, since different hardware platforms were used, and back-ground services controlled by the operating system can interfere with user interactions

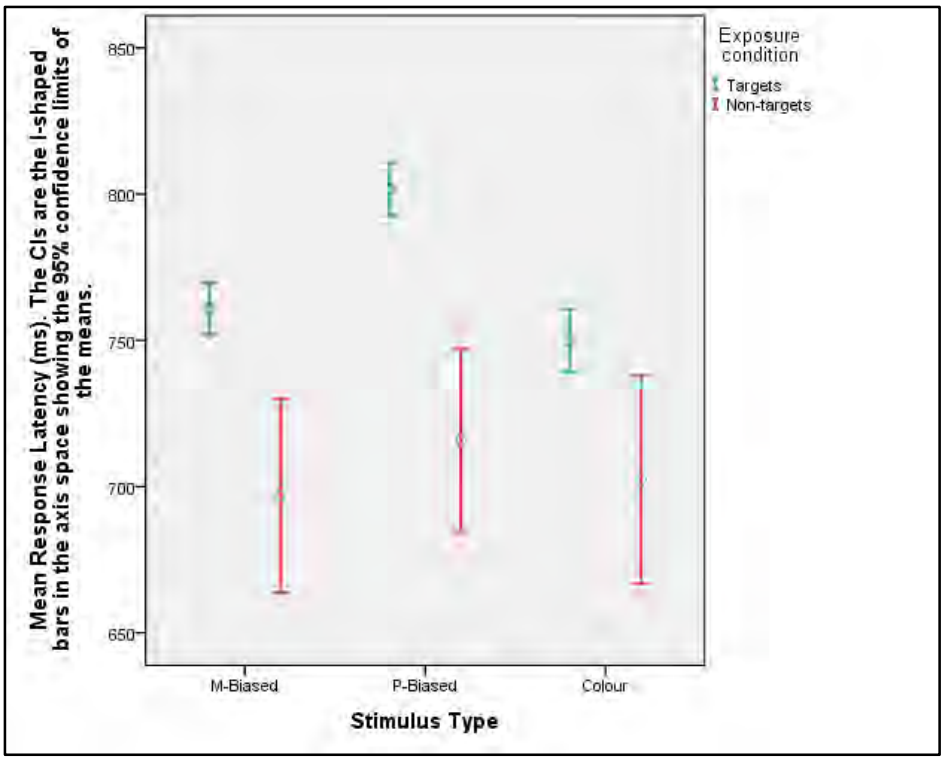


Figure 55. Experiment 3: 95% CI of mean response latency by Stimulus Type by Exposure condition. Response latency is shown on the Y-axis. The error bars indicate the 95% confidence limits of the mean for each group.

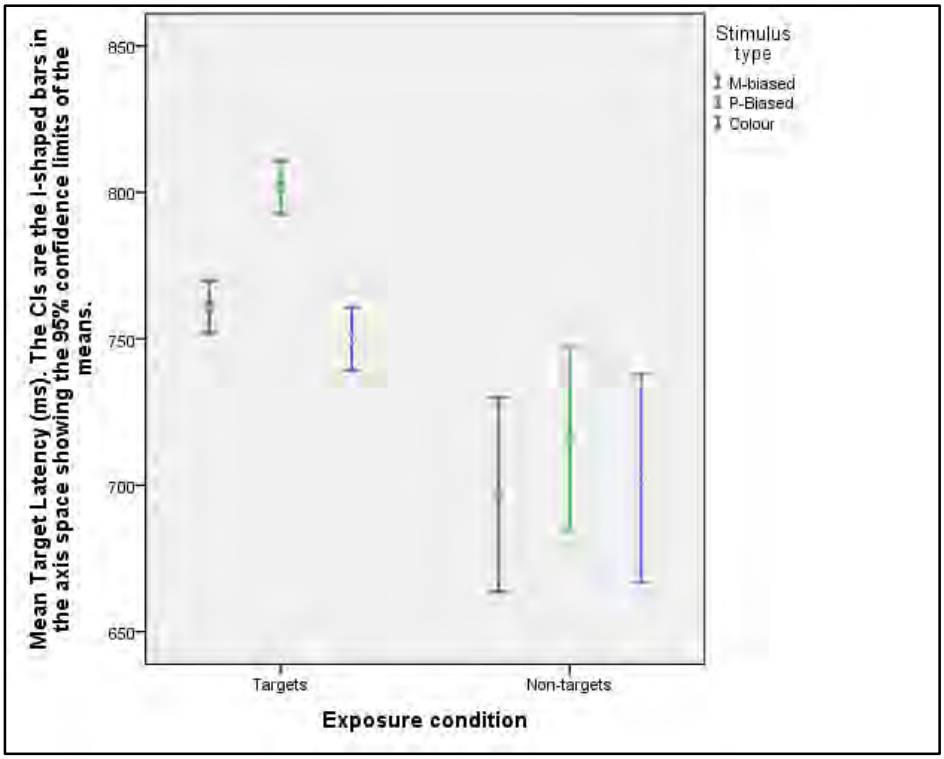


Figure 56. Experiment 3: 95% CI of mean latency by Stimulus type by Exposure condition. Response latency is shown on the Y-axis. The error bars indicate the 95% confidence limits of the mean for each group.

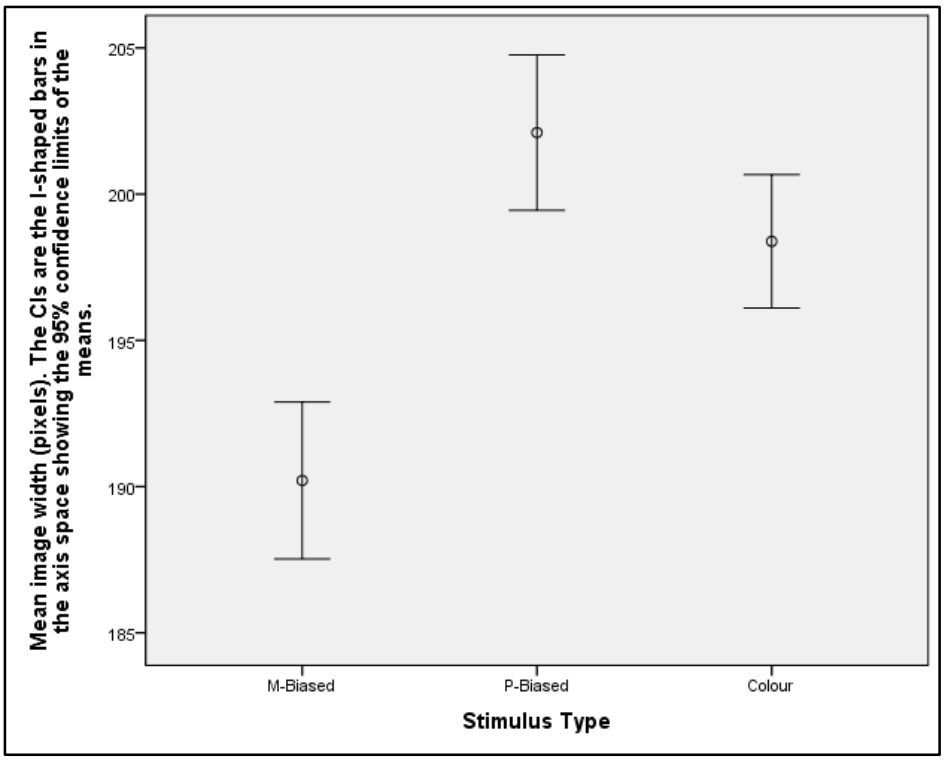


Figure 57. Experiment 3: 95% CI of mean image width responses by Stimulus type. The error bars indicate the 95% confidence limits of the mean (width in pixels) for each group.

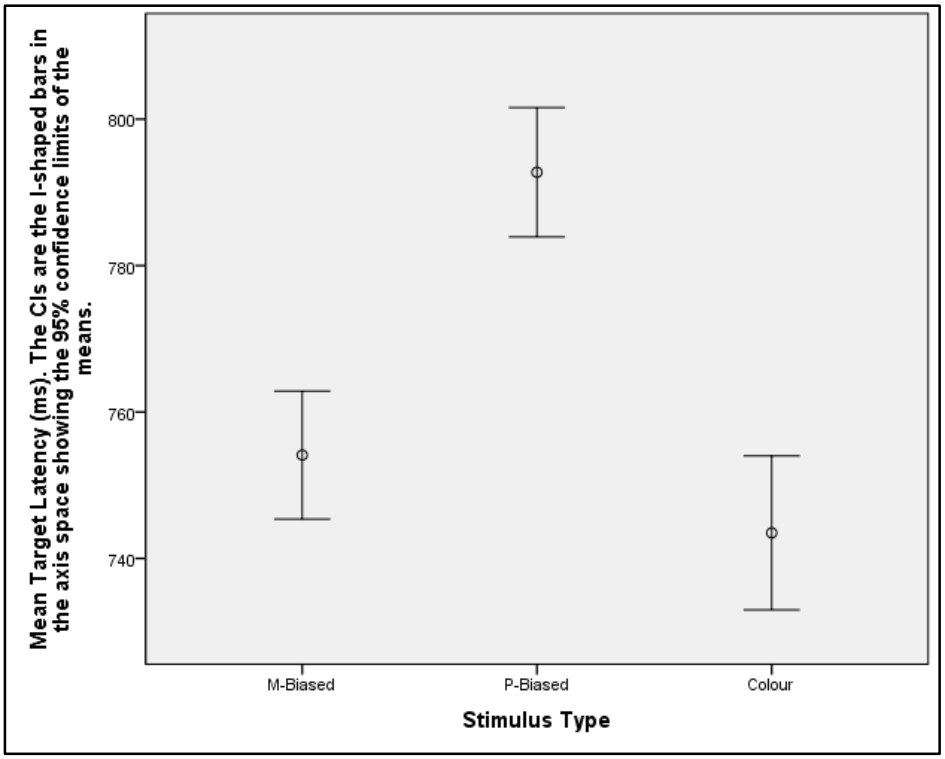


Figure 58. Experiment 3: mean response latency by Stimulus type. Response latency is shown on the Y-axis. The error bars indicate the 95% confidence limits of the mean for each group.

Figures 57 and 58 show a reasonably close correspondence between response latency (Figure 58), and the mean image width (Figure 56) for all presentation conditions. Participants had longer response latencies to P-biased images. Participants responded at larger image sizes ($\bar{x}$

= 202 vs. $\bar{x}$ = 190 pixels in width, SD = 67.5/66.3 for P/M -biased images respectively). Similarly, their response latencies were longer for P-biased, compared with M-biased images ($\bar{x}$ = 792.8 ms, $\bar{x}$ = 754.1 ms, SD = 220/220.7 respectively).

It is interesting that response latencies to non-targets (smaller than a shoe box) were shorter and more similar to each other between the different conditions, than response latencies to targets (larger than a shoe box), where P-biased presentations were associated with significantly longer latencies than full-coloured (Practice) items and M-biased items. See Figure 56. The difference between M and P-biased latencies is approximately 40 ms.

Although the distribution of the data seemed reasonable from a cursory visual inspection, all groupings of response latency scores had a positive skew, and a non-normal distribution. According to the Kolmogorov-Smirnov tests of normality all tests were significant ($p < 0.05$). These histograms are available for inspection in 'Appendix 1.pdf' (Experiment 3).

A log transformation was done to the response latencies (using the base 10) to try and improve the distribution. The kurtosis for the log transformed variable is more leptokurtic than for the untransformed variable and the skew became negative for all groupings. The Kolmogorov-Smirnov tests of normality are all significant (p <0.05).

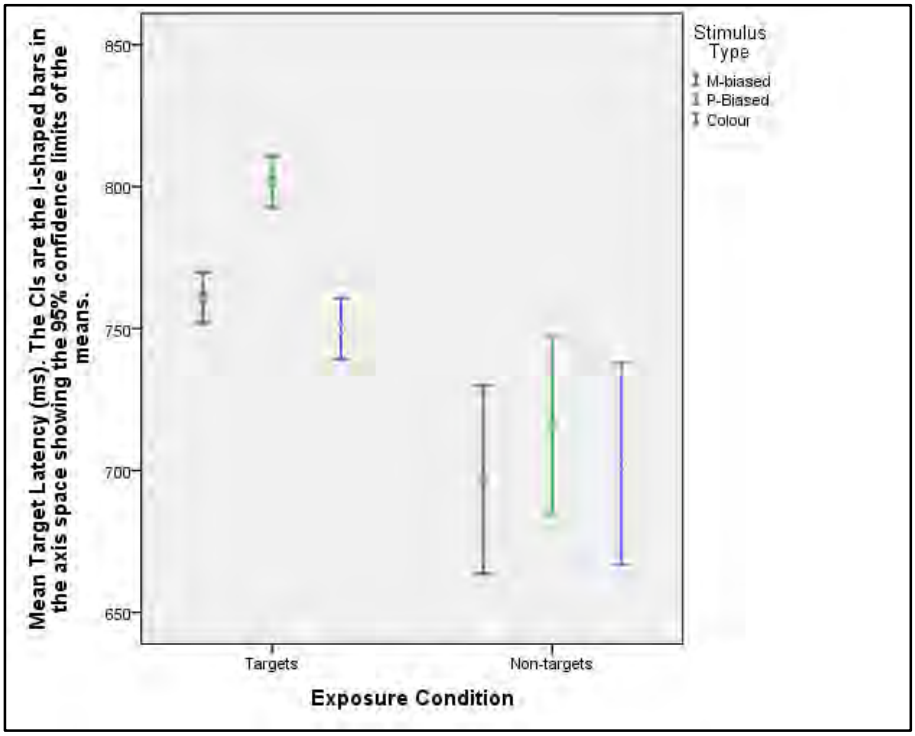


Figure 59. Experiment 3: response latency means by Stimulus and Exposure type. Response latency is shown on the Y-axis. The error bars indicate the 95% confidence limits of the mean for each group.

The log transformed data seemed worse in terms of the distribution so the untransformed data was used for an ANOVA. The ANOVA is regarded as being robust against deviations in its assumptions (normality, equality of error variances – see Howell, 1989), so a Univariate ANOVA model was used. The independent variables were 'Phase' (Colour, M / P-biased), and Response Type (Hit / FA). There were significant main effects for both independent variables: Phase $F(2) = 5.57$, $p = 0.004$; Response Type $F(1) = 57.02$, $p < 0.001$. *Post hoc* comparisons were done, controlling for multiple comparisons (Bonferroni adjustment).

Pairwise comparisons, using the Bonferroni correction show a significant difference in response latency between M-biased, and P-Biased presentations (mean difference = -38.64 ms, $p < 0.05$). There is also a significant difference in response latency between the Practice presentations and the P-biased presentations (mean difference = -49.26 ms, $p < 0.001$). These comparisons show that response latency was longer for P-biased than M-biased and Practice presentations There is also a significant difference in mean response latencies between 'Bigger' and 'Smaller' stimuli presentations (mean difference = -65.9 ms, $p < 0.001$) with longer latencies for Bigger presentations.

Because the distribution was non-normal, a non-parametric test was run to confirm the ANOVA. This test was run with the independent variables *Stimulus Type* (*Picture Size*) and the dependent variable *Latency*. The Kruskal-Wallis test with Monte Carlo resampling (10000) is significant ($p < 0.001$) for Colour Exposure Type (see Table 6). Monte Carlo resampling was used as it gives an unbiased estimate of the exact significance level. These tables give significance levels such as '0.000' which I have rounded to 0.0005.

Table 6. Experiment 3: response latency by picture size.

Test Statistics[a,b]

			Latency
Chi-Square			108.459
df			1
Asymp. Sig.			<.0005
Monte Carlo Sig.	Sig.		<.0005[c]
	99% Confidence Interval	Lower Bound	.000
		Upper Bound	.000

a. Kruskal Wallis Test

b. Grouping Variable: Stimulus type

c. Based on 10000 sampled tables with starting seed 926214481.

Table 7. Experiment 3: response latency by colour exposure type.

Test Statistics[a,b]

			Latency
Chi-Square			102.946
df			2
Asymp. Sig.			<.0005
Monte Carlo Sig.	Sig.		<.0005[c]
	99% Confidence Interval	Lower Bound	.000
		Upper Bound	.000

a. Kruskal Wallis Test

b. Grouping Variable: Exposure Type

c. Based on 10000 sampled tables with starting seed 1314643744.

The Kruskal-Wallis test is significant (p <0.0005) for the grouping variable *Exposure type* (M- / P-biased / Coloured picture types) - see Table 7.

2.5.6 Discussion

The mean d' scores were >3.0; >4.0, >0.4 for the Colour, M- and P-biased phases respectively. These are robust scores, which show reliable detection and one can be confident that detection accuracy was at a reasonable level. These scores were better than scores seen in Experiment 2.

In summary:

1. An ANOVA test showed a significant difference between the 3 stimulus colour conditions, $F(2, 165) = 6.01$, $p = 0.003$, but the difference between the P and M conditions in terms of d' seems negligible.
2. Females tended to outperform males – most noticeably in the M-biased condition – see Figures 47, 48 and this is difficult to explain. The fact that the groups were unbalanced may have created an artefact.
3. Non-target response latencies (approximately 750 ms) were at similar levels, and tended to be faster than for targets (between 800 - 850 ms). The Colour means appear similar to those of the M-biased presentation phase. Mean latency for targets in the P-biased phase appear longer than for M/Colour means by about 45 – 50 ms.
4. Participants had longer response latencies to P-biased vs. M-biased images and responses were delayed by about an average of 12 pixels in width until they had

zoomed to larger sizes ($\bar{x}$ = 202 vs. $\bar{x}$ = 190 pixels in width, SD = 67.5/66.3 respectively). Similarly, their response latencies were longer (by approximately 39 ms) for P-biased, compared with M-biased images ($\bar{x}$ = 792.8 ms, $\bar{x}$ = 754.1 ms, SD = 220/220.7 respectively).

5. Response latencies to smaller objects (smaller than a shoe box, i.e. non-targets) were shorter in M-biased, P-biased and normally coloured object presentations and more similar to each other than response latencies to larger objects (larger than a shoe box, i.e. targets) where P-biased presentations were associated with significantly longer latencies than normally-coloured (Practice/Colour) items and M-biased items. The difference between the P- and M-biased conditions was approximately 40 ms.
6. Pairwise comparisons, using the Bonferroni correction, show significant difference in response latency between M-biased, and P-Biased presentations (mean difference = -30 ms, $p < 0.05$). There is also a significant difference in response latency between the Colour presentations and the P-biased presentations (mean difference = -32.6 ms, $p < 0.05$). The P-biased and M-biased response latencies were also significantly different (mean difference = -38.64, $p = <0.05$). There was also a significant difference in response latencies between 'Bigger' and 'Smaller' exposure conditions (mean difference = -65.1 ms, $p < 0.05$).
7. Since the distribution was skewed and the error variances were significantly different, non-parametric tests were also run with the independent variables *Stimulus Type Type* and *Picture Size* and the dependent variable *Latency*. A Kruskal-Wallis Test with Monte Carlo resampling (10000) was significant ($p < 0.0005$) for Colour Exposure Type (normally coloured Practice Phase items, vs. M-biased and P-Biased presentations. The Kruskall-Wallis test with Monte Carlo resampling (10000) was also significant ($p < 0.0005$) for the the grouping variable 'Picture Size', showing longer response latencies across all presentation colour conditions for the large picture category.
8. The mean picture width for responses to targets was greater for all conditions, see Figures 60 - 61.
9. The strategy of presenting images and zooming them rapidly may have introduced a confound in that image size and response latency are necessarily correlated. It is therefore not possible to determine whether differences in response latency are due to image size or presentation colour (M/P). However, other inferences may be drawn

from the means plots (Figures 55, 56). It seems clear that false alarm latencies are similar for the 3 exposure conditions, but that hit latencies are quite close (overlapping confidence intervals) for the Colour and M-biased conditions. The *difference* in mean non-target, and target response latency in the Colour and M-biased conditions is about 50 - 60 ms, whereas it is about 85 ms in the P-biased condition. It seems that errors are made at similar latencies in all conditions and at shorter latencies, so the difference between non-target means and target means may signify the amount of time required for cerebral processing associated with accurate recognition of the object, as opposed to merely reacting to the appearance of a stimulus on the screen. The M-biased and Colour latency means are within 10 ms of each other, and the non-target latency means are within 3 - 4 ms of each other. The additional time, required by the P system could be estimated by subtracting the M and Colour mean target latencies from the P-biased target latency. This yields an estimate of 50 - 60 ms. Estimates of conduction speed differences between the M and P pathways vary considerably. Kveraga *et al.* (2007b) found a speed advantage of approximately 100 ms for the M-biased stimuli. However, Maunsell, Ghose, Assad, Mcadams, Boudreau & Noerager (1999) found conduction differences between the two pathways were approximately 10 ms when measured at the LGN of a macaque.

10. The general conclusion for this experiment is that while some differences were seen between the M- and P-biased conditions in terms of both latency and the size at which responses were made, these two measures are highly correlated. The reason that responses to P-biased stimuli are longer may be that there were delays associated with processing by the slower P system because the faster M system contribution to recognition was not available due to the blue/green colour that these images were presented in. However, the fact that the images were zoomed from an initially small size may have slowed the M system response, as this system is considered to have poorer spatial frequency resolution. This may have impacted on the predicted difference in d' between the M- and P-biased conditions.

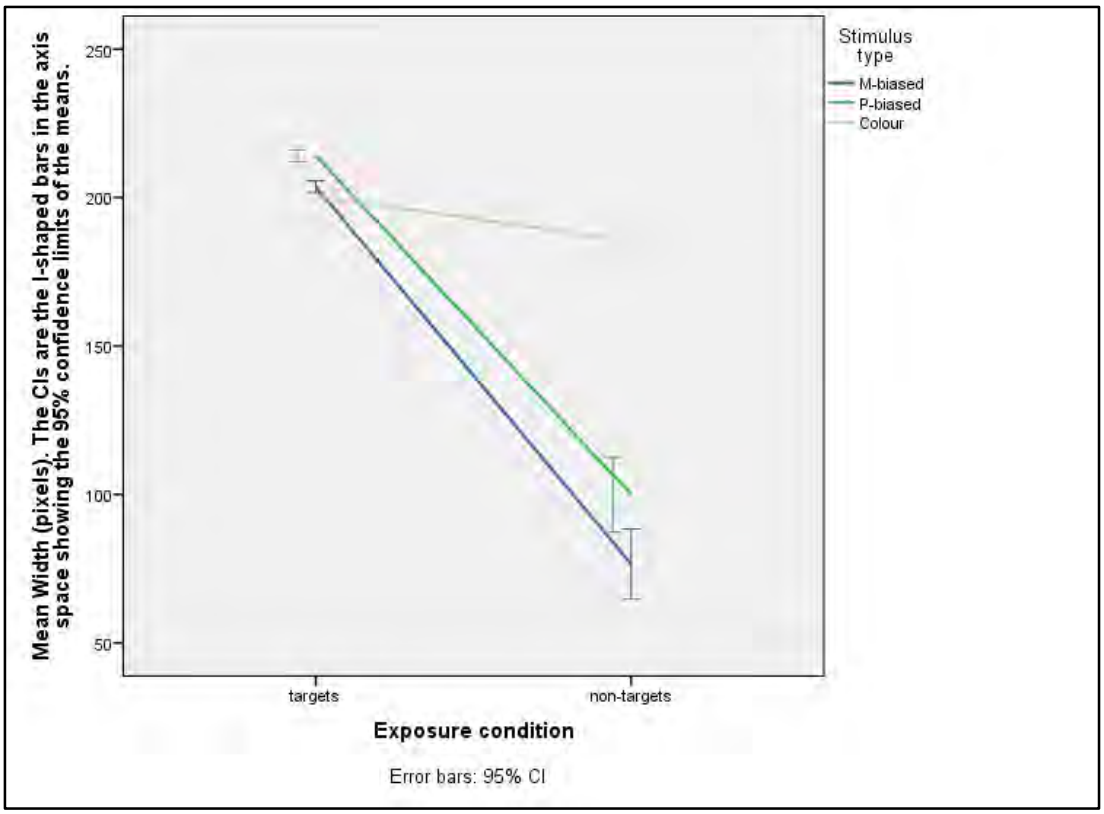


Figure 60. Experiment 3: mean stimulus width for different presentation conditions. The error bars show the 95% confidence intervals of the means for each category. Each line shows a category – the blue line represents M-biased images, the green line represents P-biased images and the light brown line represents normally coloured images. The target type (targets / non-targets) categories are shown on the x-axis.

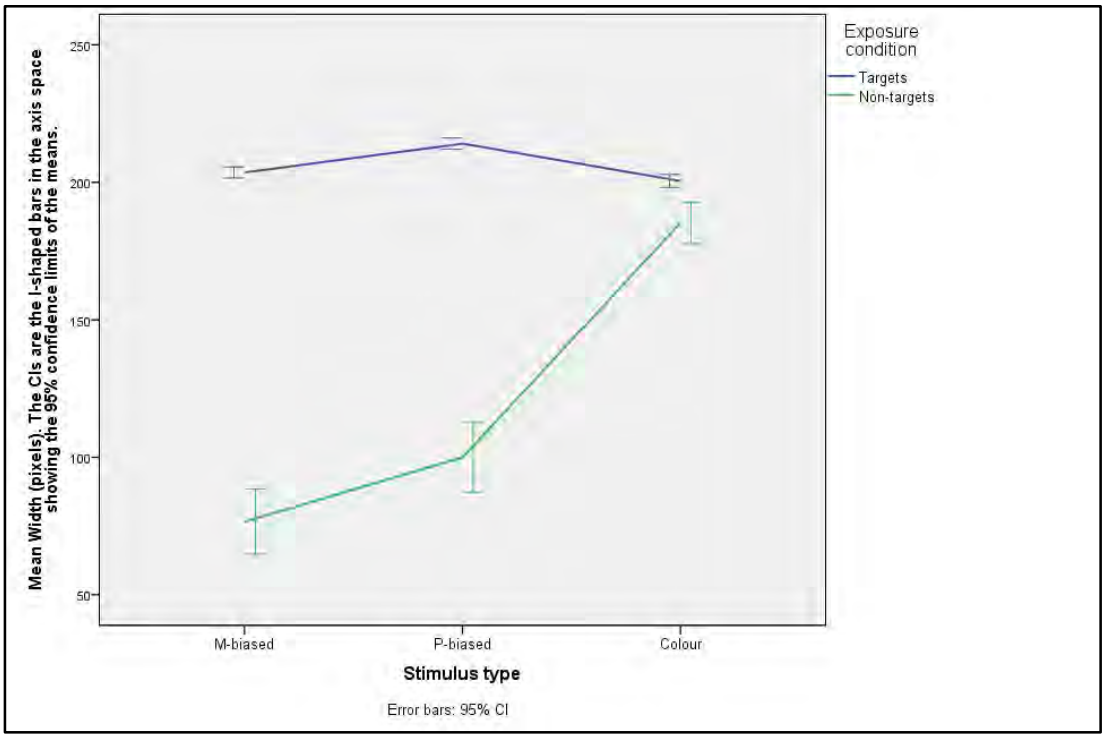


Figure 61. Experiment 3: mean stimulus widths by Stimulus typ. The error bars show the 95% confidence intervals of the means for each category. Each line shows a category – the blue line represents target presentations and the green line represents non-target presentations.

2.6 Experiment 4: The role of the M system in reading – second follow up study

2.6.1 Introduction

Experiment 1 (black type against a red background) seemed to yield reasonable results and Experiment 2 demonstrated that the cyan/green colour solution for the P-biased presentations was effective in significantly reducing detection accuracy, compared with the M-biased and normal (black type on white background). Experiment 4 was designed to test the colour combinations suggested by Kveraga (2010, Personal Communication), namely RGB(0,70,0/77,0,0) for the P (isoluminant colour condition) and RGB(30,30,30/50,50,50) for the M (luminance, or achromatic condition) and compare them with the results of Experiment 2. This experiment was a preparation for the proposed replication of the Kveraga, *et al.,* 2007 experiment.

The following hypotheses were tested with the colour combinations stated above:
H_1 Word detection accuracy will be poorer when words are presented with high colour contrast, and low luminance contrast (P-biased), and response latency will be longer than when presented with low colour contrast, but higher luminance contrast (M-biased).
H_2 Word detection accuracy will be better and response latency will be shorter in the M-biased higher luminance contrast, low colour contrast than the P-biased condition.

Instead of presenting the words in the colour combinations of Experiment 2 (green type on a cyan background for the P-biased condition), an approximation of Kveraga *et al.*'s M and P-biased conditions was used, with red/green for the P-biased condition, taking the suggestion that the screen being darkened by about 50%. The M-biased condition was made more stringent: the type back-ground/font colour: RGB(30,30,30) / RGB(50,50,50) vs. RGB(128,128,128) / RGB(160,160,160) for Experiment 4 and 2 respectively. The screen background was darkened considerably in order to optimise conditions for this solution – see Figure 62.

2.6.2 Participants

Participants were recruited from an undergraduate class. They received course credit for attendance at the experiment, but were given the option of refusing participation, or withdrawing their data. They were required to sign a register by logging into the program,

and then they could elect to do the experiment, or stop at any point. This experiment took approximately 20 minutes. The same arrangement was followed for entering an anonymous ID, and recording it so that they could identify their own results. The protocol was similar to that of Experiment 2. There were a total of 36 participants (9 males, 27 females). Mean age was 21.3, SD = 1.7.

2.6.3 Materials

The program was ported from Embarcadero RAD Studio (2009) to Embarcadero RAD Studio (2010), using the Delphi language which compiles to native Win32 executable code. The program was run on 13 workstations and the data was written to local ASCII files. The stimulus word list consisted of the same 100 correct words and 100 pseudo-words as was used in Experiment 2. Each time they were presented, they were shuffled into a random order.

Table 8 shows the colour RGB values for each condition. Words in the Practice Phase were in black type against a white background. Figure 62 illustrates the 'M' and 'P' conditions visually.

Table 8. Experiment 4 colour values.

Colours for P-biased condition:
Back-ground: RGB(0,70,0);
Font colour: RGB(77,0,0);

Colours for M-biased condition:
Back-ground: RGB(30,30,30);
Font colour: RGB(50,50,50);

Screen back-ground colour:
RGB(25,25,25).

Font: Arial bold, 22 point.

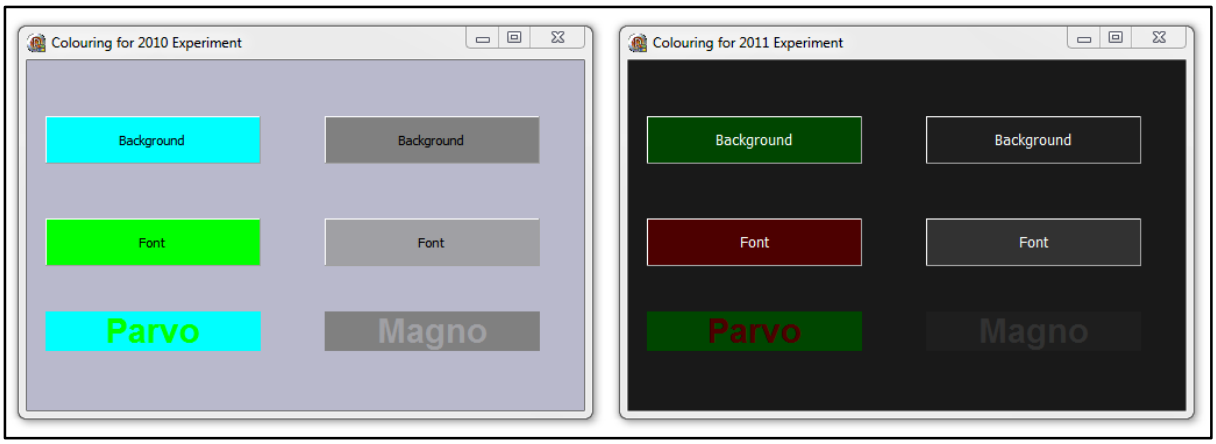


Figure 62. Comparison of colour combinations and backgrounds for Experiment 2 (LHS) and Experiment 4 (RHS).

2.6.4 Procedure

The procedure was the same as that of Experiment 2. The screen was reformatted slightly in terms of colour, layout, and minor wording changes. The calibration procedure was the same as Experiment 2 and the experimental protocol was identical, except for these changes in the colours of the word presentations, and background.

2.6.5 Results

The randomisation of trials was satisfactory: the Binomial test showed a reasonable distribution in terms of the trial ordering (p = 0.41) – see Figure 63. Generally, there were no significant patterns in the results.

A One-way ANOVA test for d' across the 3 conditions was non-significant, $F(2) = 1.2$, p = 0.31. It is unlikely there were any differences between the mean target and error latencies (Figure 64 – 67), although this was not tested statistically because of the considerable overlap in confidence intervals. In general, the means for the Practice (full contrast) and M-biased phases were similar to those seen in Experiment 2: $\bar{x}$ = 0.42/0.42; SD = 0.49 /0.45 ; ($\bar{x}$ = 0.49/0.63, SD = 0.50/0.58). However, d' scores were better in Experiment 4 for the P-biased phase: ($\bar{x}$ = 0.21/0.53, SD = 0.81/0.66): Experiment 2/4 respectively.

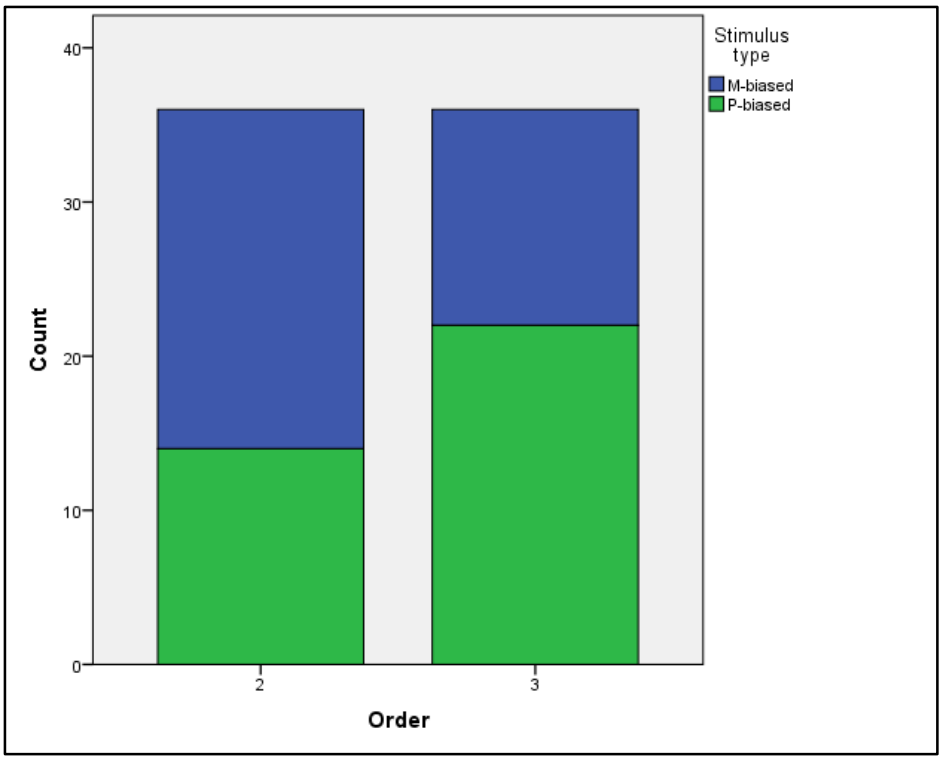


Figure 63. Experiment 4: Frequency of M- / P-biased presentation by trial block order.

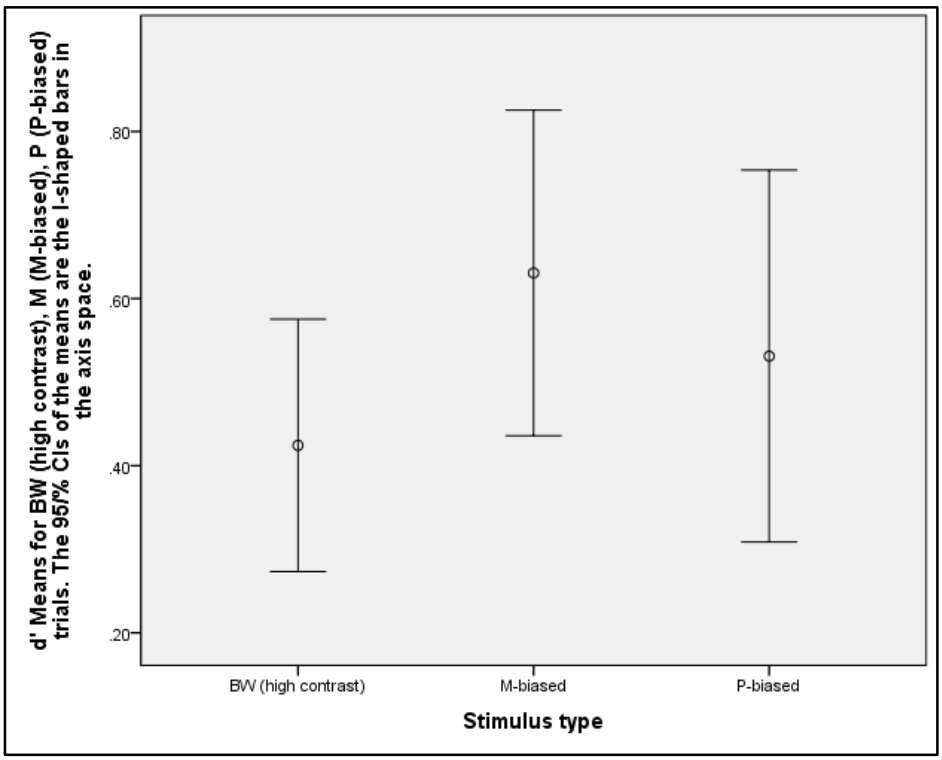


Figure 64. Experiment 4: 95% CI of d' for BW (high contrast), M-biased P-biased conditions. Error bars show the 95% confidence limits of the means for each condition.

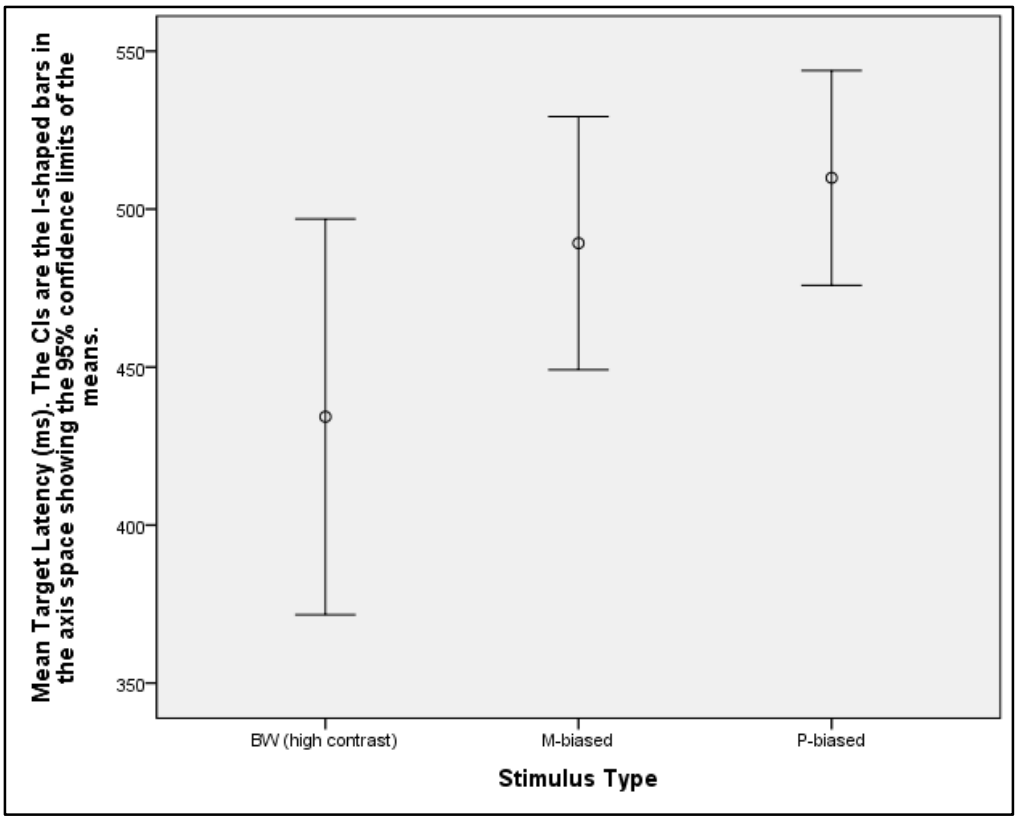


Figure 65. Experiment 4: Error bars show 95% CI of mean target latencies for BW/M/P-biased conditions.

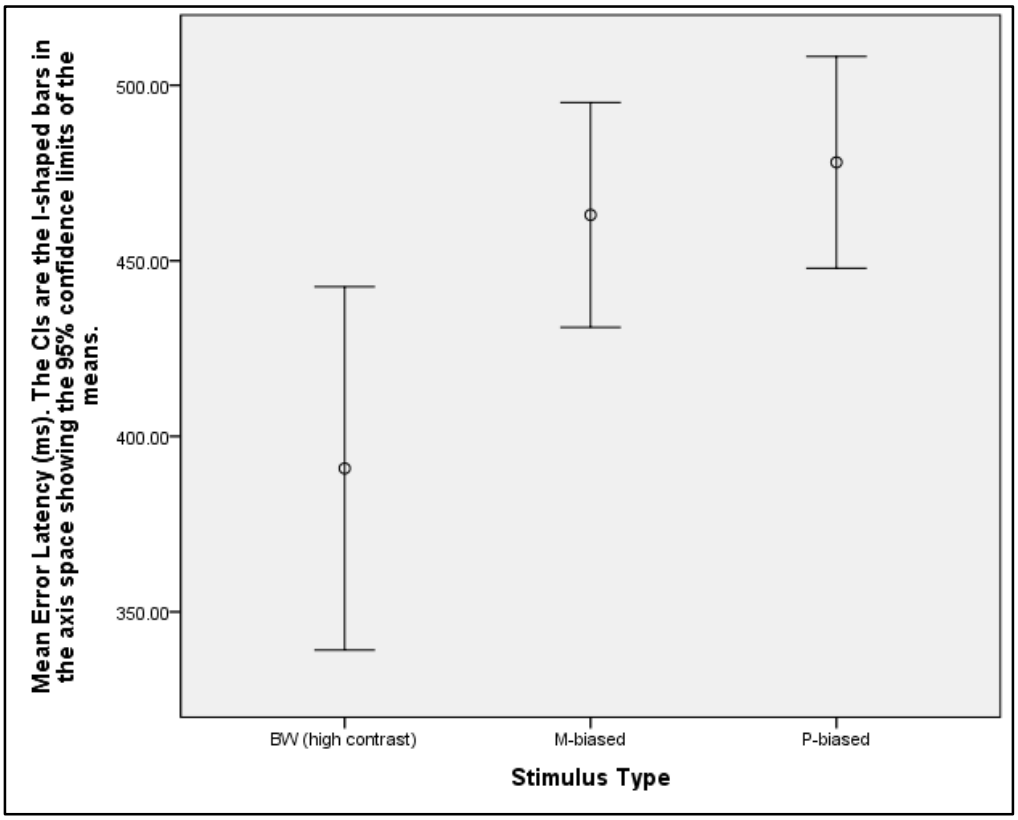


Figure 66. Experiment 4: Error bars show 95% CI of error latency means for BW/M/P-biased stimulus types.

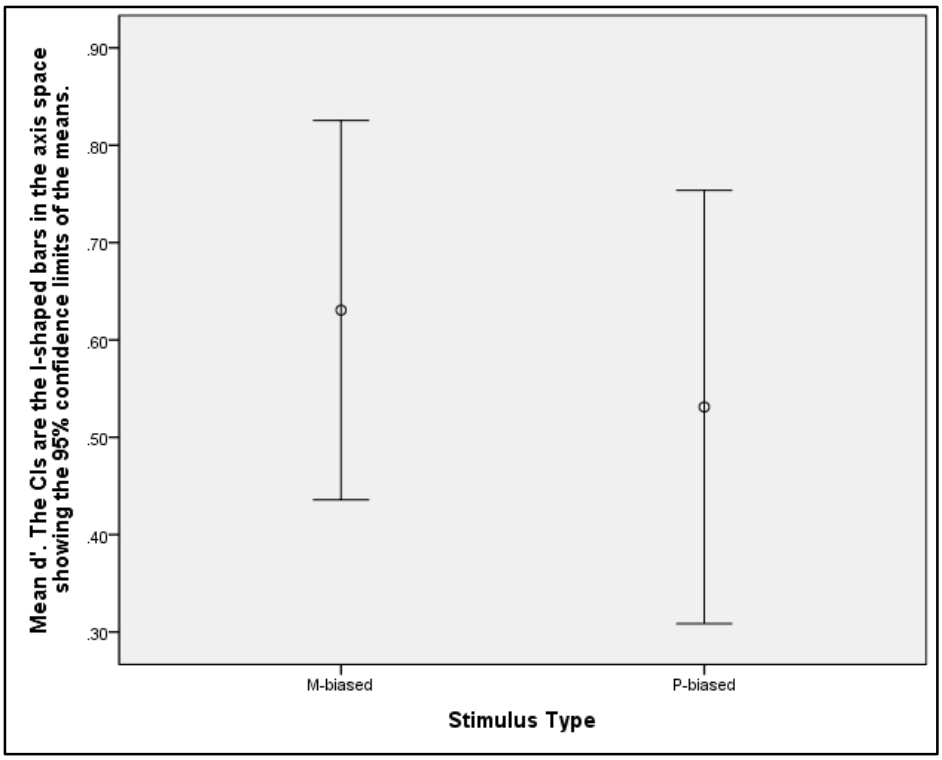


Figure 67. Experiment 4: Error bars show 95% CI of d' for M-biased and P-biased conditions.

It seems self-evident that the improvement in scores for the P-biased condition is due to the fact that the red/green colour solution makes the type relatively easy to see. Whether these colours are isoluminant is difficult to determine and whether they were isoluminant on the various work stations is impossible to say.

A comparison of the distribution of scores for the 'P-biased' condition for Experiment 2 (Figure 68) and Experiment 4 (Figure 69) shows that there is a clear negative skew in Experiment 2, but not in Experiment 4. The distributions for the other phases in Experiment 2 and 4 are not as skewed and unusual (Figure 70 - 73).

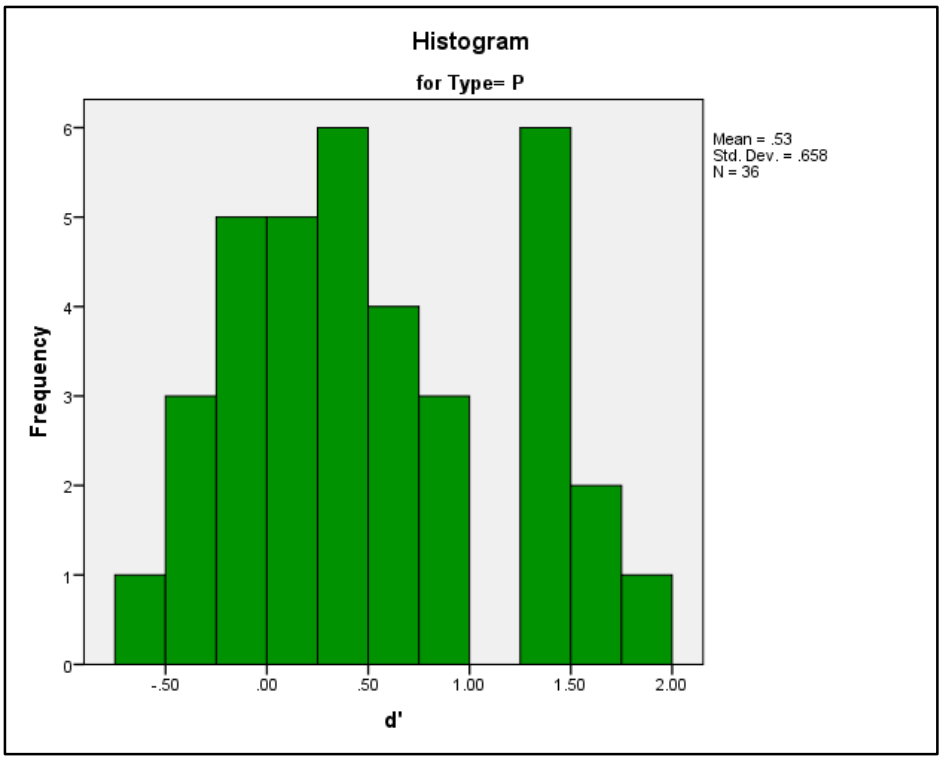


Figure 68. Experiment 4: Distribution of d' scores for P-biased stimuli.

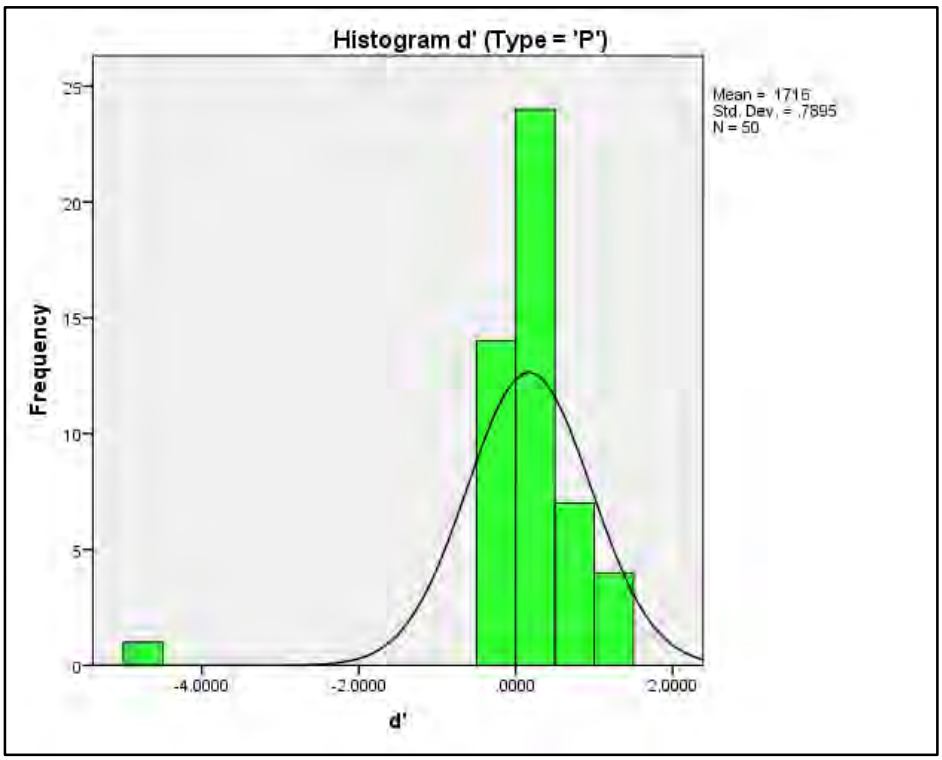


Figure 69. Experiment 2: Distribution of d' scores for P-biased stimuli.

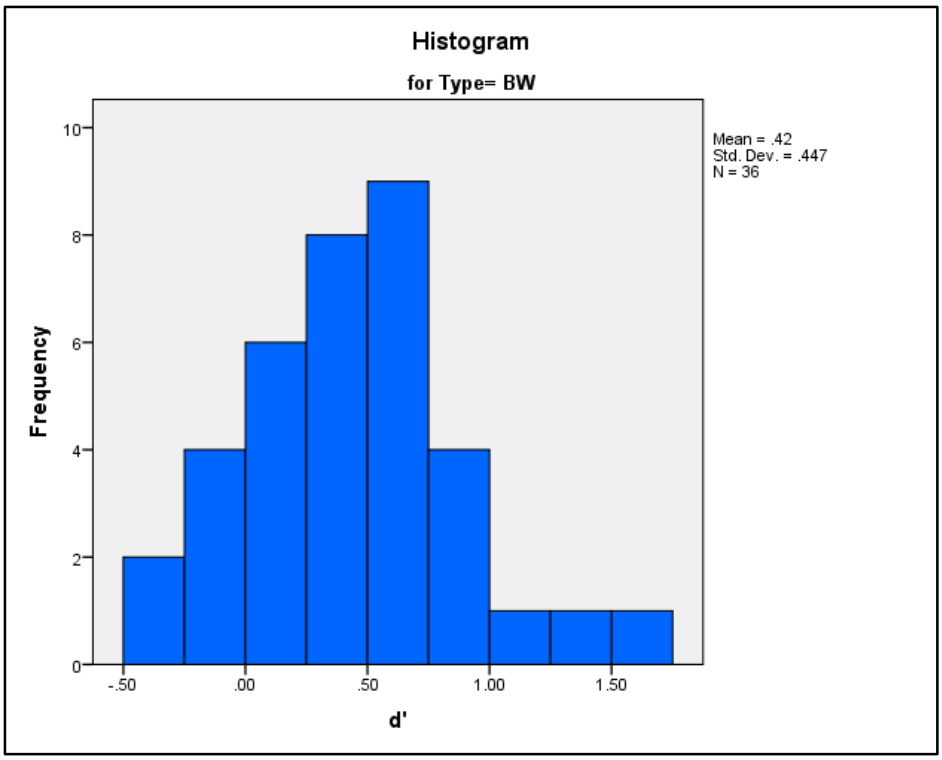


Figure 70. Experiment 4: Distribution of d' scores for BW/high contrast stimuli.

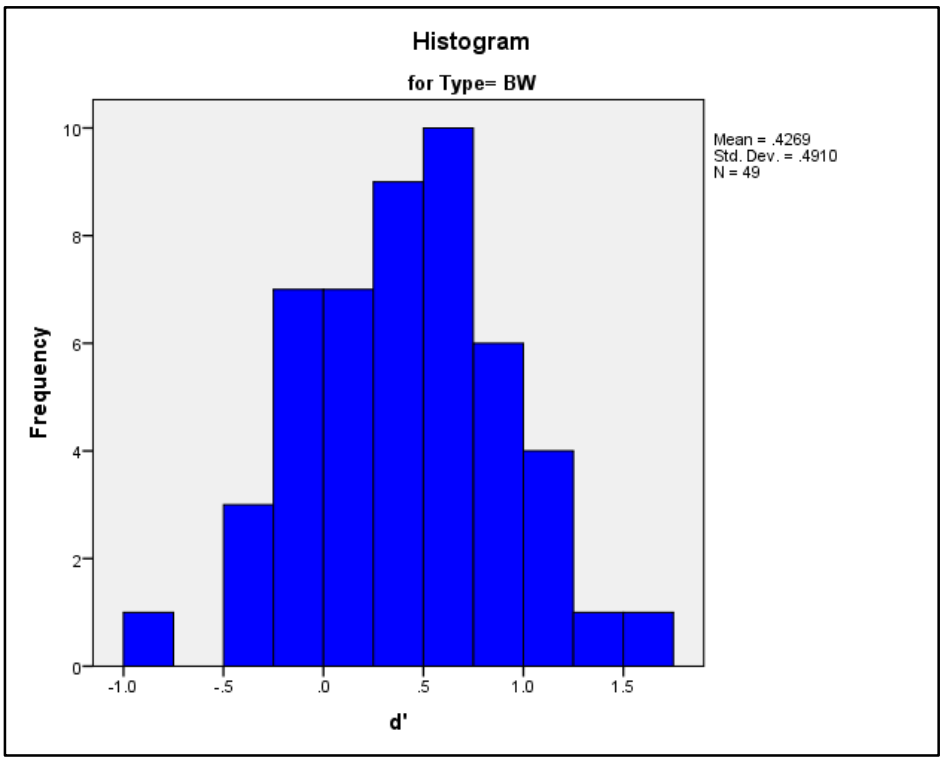


Figure 71. Experiment 2: Distribution of d' scores for BW/high contrast stimuli.

The distributions are very similar between Experiment 4 and Experiment 2 (Figure 70 – 71). The means are similar (0.42 /0.43) and the standard deviations are comparable (0.45 / 0.49) respectively. This suggests a good degree of uniformity for comparable phases of this experiment.

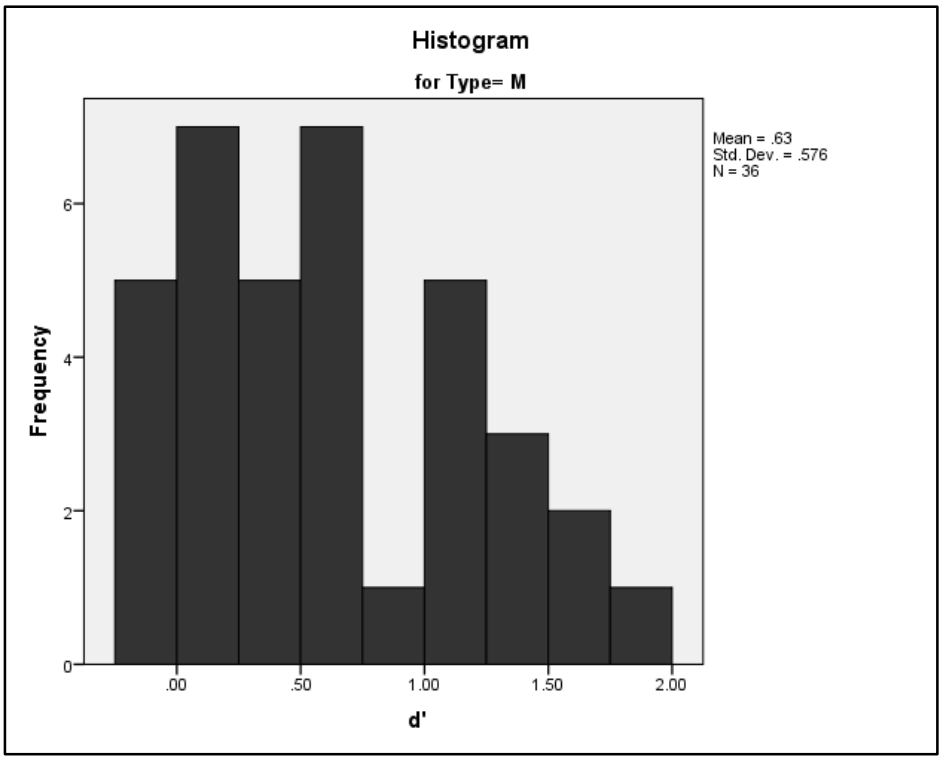


Figure 72. Experiment 4: Distribution of d' scores for M-biasd stimuli.

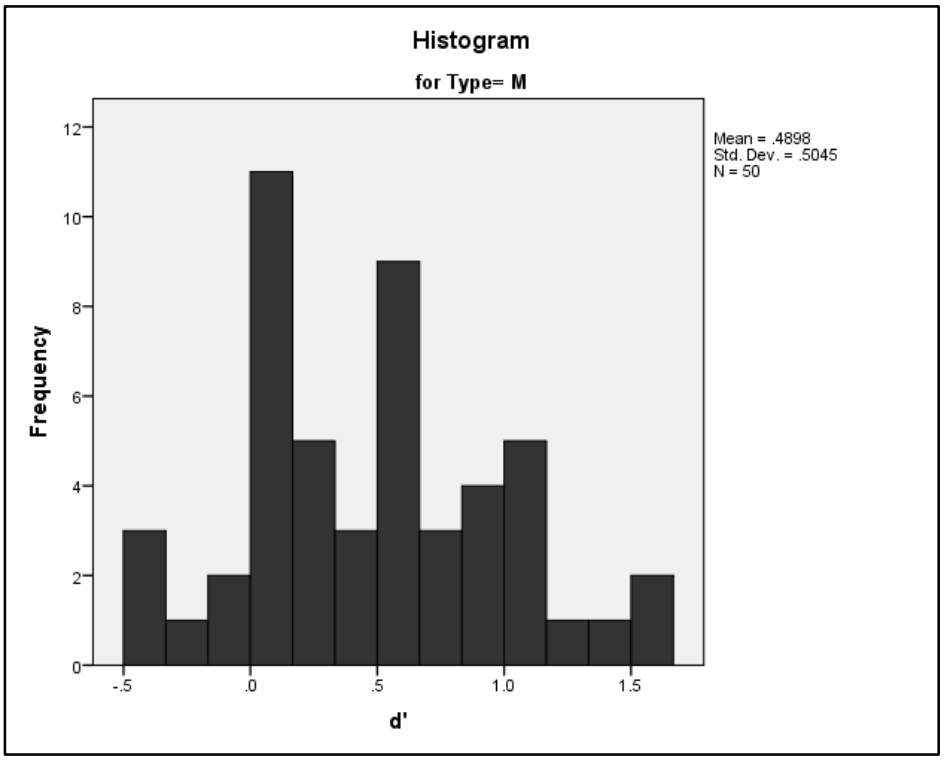


Figure 73. Experiment 2: Distribution of d' scores for M-biased stimuli.

The distributions for the M-biased phase in Experiment 4 and 2 are less similar (Figure 72, 73), although the central tendency and standard deviation are comparable ($\bar{x}$ = 0.63/0.49, SD = 0.58/0.50). Impressionistically, it does seem that as participants' difficulty of detecting the stimuli increases, the distribution shifts away from normality. In Figure 69 for instance, most scores are clustered around the level of zero, which is equivalent to the level of chance responding, although there is a significant cluster near the level of 1.0 where discrimination is better than chance. The fact that these two groups performance differed from each other most clearly in the P-biased phase seems to be due to the different colours used for the P-biased condition), since there was no comparable performance disjunction in the *Practice*/*high contrast*, or *M-biased* condition.

It is interesting to note that although contrast in the M-biased condition was different between Experiment 2 and 4, there was not much difference in the general level of performance. In effect, the contrast difference was RGB(32,32,32) / RGB(20,20,20) for these experiments respectively – see Table 9 for a comparison (relevant values in bold for each condition):

Table 9. Comparison of P/M condition values between Experiments 2/4.

Experiment 2
Colours for P-biased condition:
Back-ground: RGB(0,255,255);
Font colour: RGB(0,255,0).
Colours for 'M' condition:
Back-ground: RGB(128,128,128);
Font colour: RGB(160,160,160).
Screen back-ground colour:
RGB(240,240,240).
Experiment 4
Colours for P-biased condition:
Back-ground: RGB(0,70,0);
Font colour: RGB(77,0,0)
Colours for 'M' condition:
Back-ground: RGB(30,30,30);
Font colour: RGB(50,50,50);
Screen back-ground colour:
RGB(25,25,25).
Font: Arial bold, 22 point.

This is also illustrated in Figure 62.

2.6.6 Discussion

There is no evidence that the colour combinations used for the P-biased presentations in Experiment 4 were isoluminant, or that they were any more difficult to detect than the M-biased stimuli. The experiment showed similar results to that of Experiment 2 for the Practice phase where words were presented in black type against a white back-ground, and the d' means and target latency means were similar for this condition. The following are d' means for Experiment 2/4 respectively: $\bar{x}$ = 0.42/0.42; SD = 0.49 / 0.45. The mean target latencies were: $\bar{x}$ = 450.88 / 434.28; SD = 115.01 / 185.127 respectively.

The d' means for the M-biased condition were similar between Experiments 2/4 ($\bar{x}$ = 0.49 / 0.63, SD = 0.50 / 0.58), and the mean target latencies were: $\bar{x}$ = 497.63 / 489.22; SD = 100.26 / 118.48 respectively.

The results between Experiment 2 and 4 seem consistent for these conditions, and the only inconsistency is associated with the changed P-biased condition when detection was more difficult in Experiment 2 where the cyan/green colouring was used than Experiment 4 where the red/green colouring was used (d': $\bar{x}$ = 0.21 / 0.53, SD = 0.81 / 0.66 respectively).

To reiterate: the colours chosen for the P-biased condition in Experiment 2 were based on 3 sources of ideas. The first was the article by Livingstone (1988) which offers a visual example of isoluminance which seems clear (Figure 21). The second source was Gazzaniga (2005), who reports that the maximum colour sensitivity of the rod cells is in the region of 495 nm. As this is the region which falls between the S (short wavelength/blue sensitive cones) and M (medium wavelength/green sensitive cones) and given the likelihood that this is the region of the spectrum which the M system, which receives significant rod input, is least able to discriminate blue from green, it is likely that the M system signal carried very little information. It also seems to fit subjectively with Livingstone's (1988) example of an isoluminant image which is printed in green against a bluish back-ground. If the frequency window in which the rod cells are relatively insensitive is approximately 495 +/- 30 nm, this would be more or less in the cyan/green range.

A number of other modelling tools were used to explore the phenomenon of isouminance:

1. The Barber's pole illusion (from Gazzaniga, 2005) was used to explore the motion illusion with different settings, colour combinations, flicker frequencies, etc. The program is available on a cd 'Pole.exe' and a Dropbox folder with instructions on how to use it. This will be shared on request to Prof. Colin Tredoux (colin.tredoux@uct.ac.za).
2. Mach bands were used for experimentation with colour combinations (see http://www.nku.edu/~issues/illusions/MachBands.htm, https://courses.cit.cornell.edu/bionb2220/UnderstandingLateralInhibition.html for examples)
3. The following illusions by Kitaoka (downloaded from http://www.ritsumei.ac.jp/~akitaoka/index-e.html 20 July, 2009): 'Rotating snakes' (Kitaoka, 2003), 'Rotating rays' (Kitaoka, 2004), 'Rollers' (Kitaoka, 2004), 'A Bulge' (Kitaoka, 1998). Various manipulations with colour were done to test their effects on the illusion of motion.
4. Hermann's Grid Illusion (downloaded from https://courses.cit.cornell.edu/bionb2220/UnderstandingLateralInhibition.html on 7 September 2011).

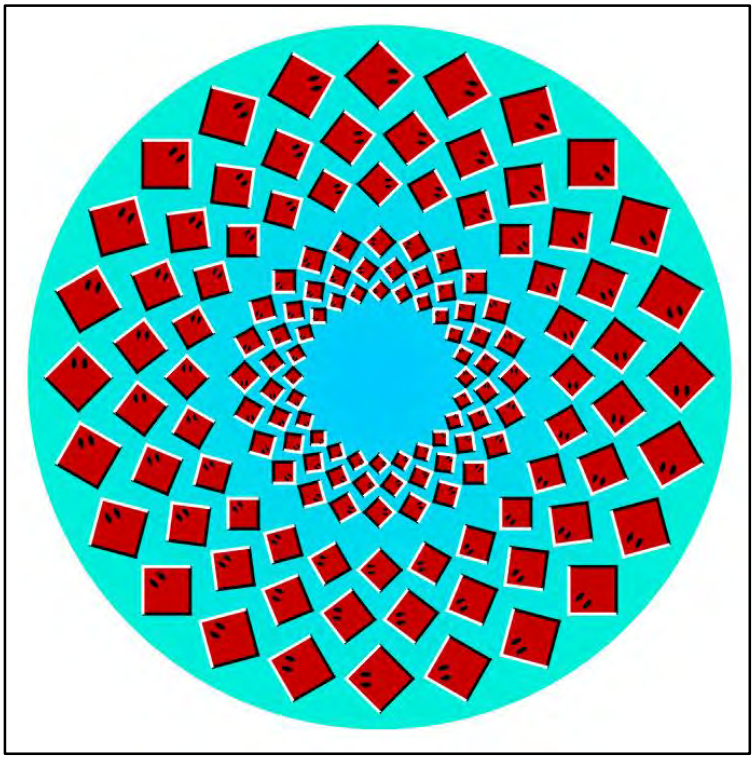


Figure 74. Rotating rays (Kitaoka, 2004) - original colours.

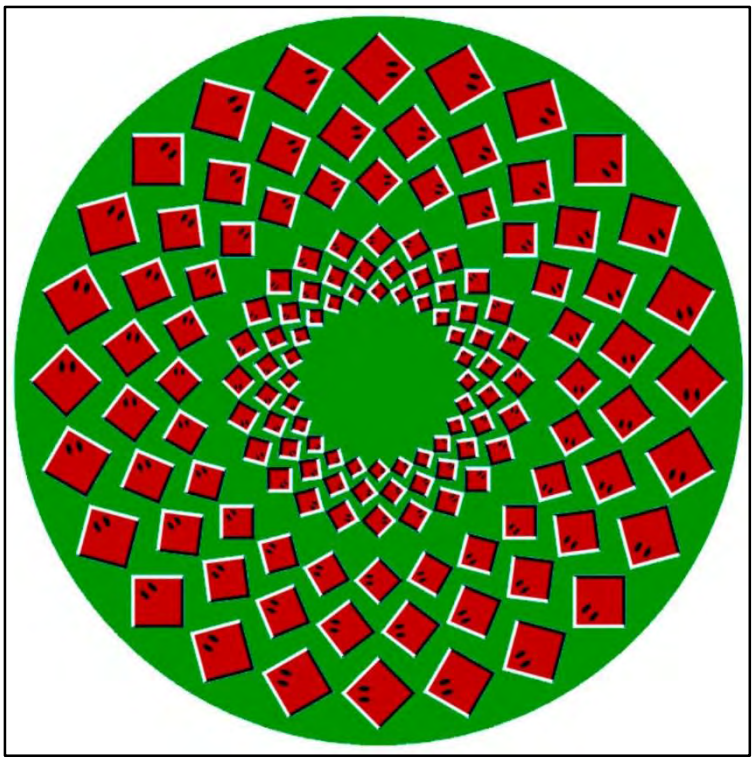


Figure 75 Rotating rays adjusted to red/green (clockwise rotation evident).

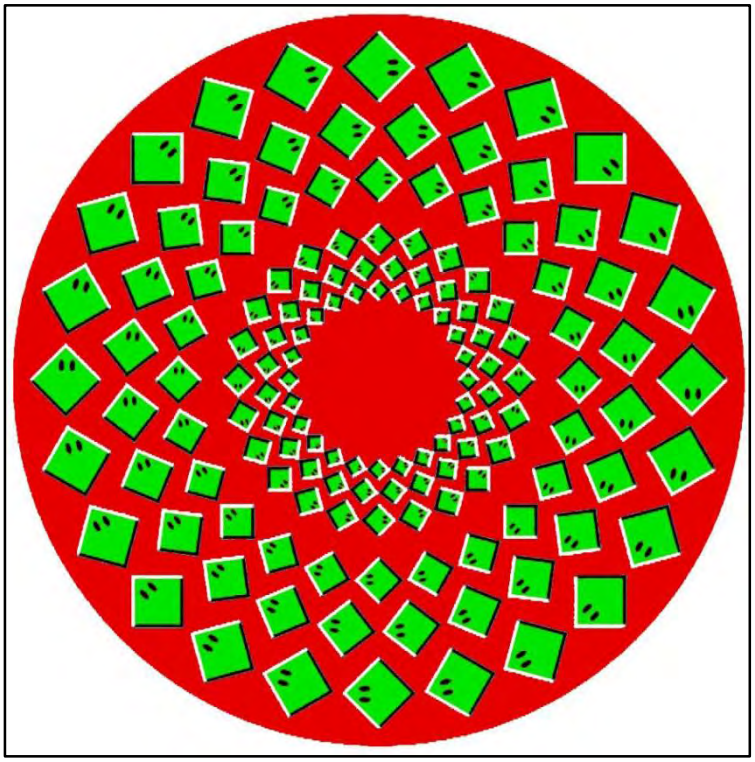


Figure 76. Rotating rays adjusted to reverse colour scheme (anti-clockwise rotation evident).

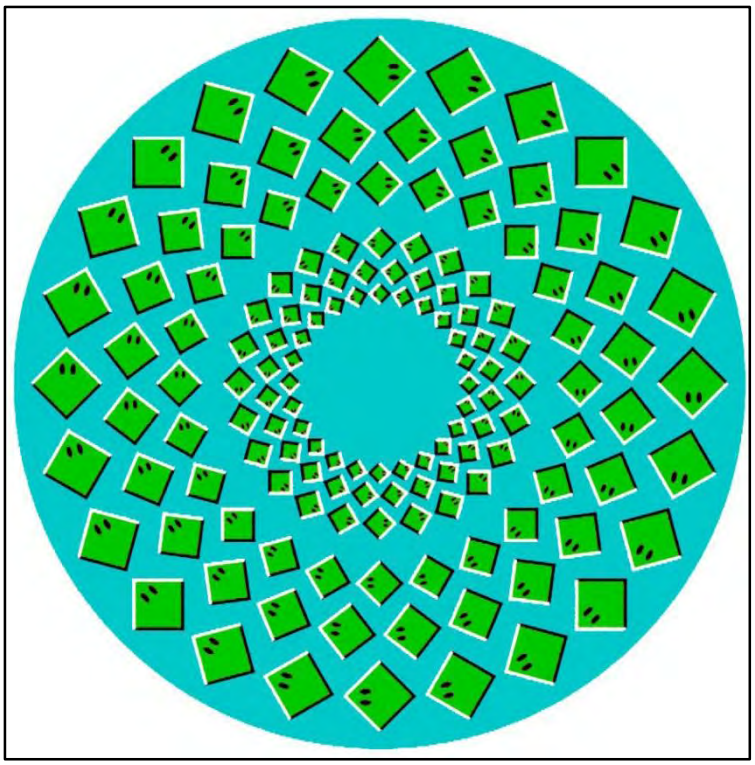


Figure 77. Rotating rays illusion adjusted to green/cyan colour scheme (rotation halted).

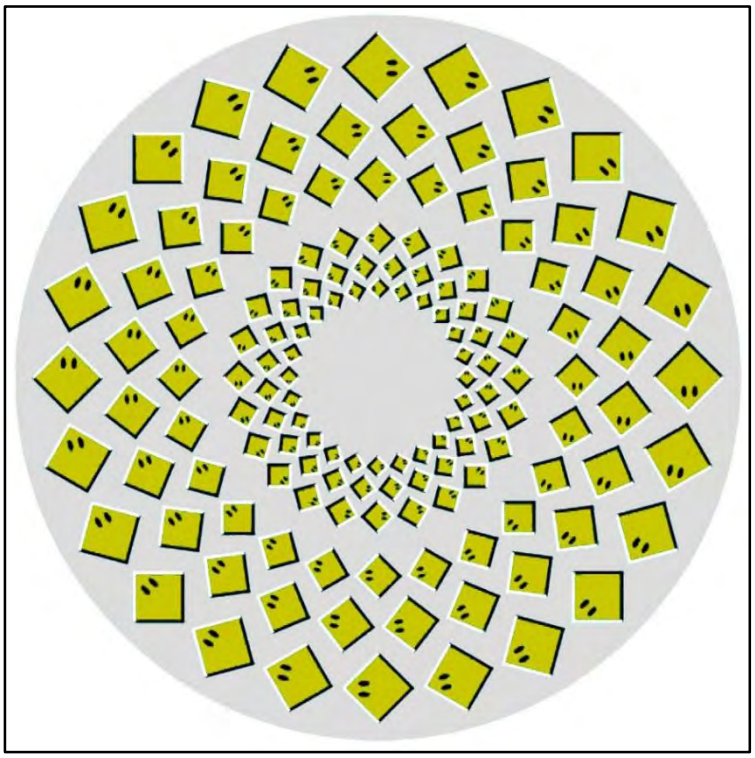


Figure 78. Rotating rays illusion adjusted to yellow/grey colour scheme (rotation halted).

The above figures (74 – 78) show the effects of different colour manipulations. These, and other examples of colour manipulations with motion illusions are available on a media disk and Dropbox. The applications ‘FM Image Processor’ or ‘Image Processor’ may be used and the folder ‘Transformed Images’ and ‘Test Files contain suitable images for demonstration.

2.7 General discussion and conclusion

This chapter has explored word and object recognition in 4 experiments:

1. Experiment 1 used ideas from Chase *et al.* (2003). There were indications of a significant effect associated with an intervention aimed at inhibiting contributions of the M system. The sample size was small and there were problems with trial ordering. However, the effect was statistically significant and in the predicted direction.

2. Experiment 2 was devised with a host of methodological and practical challenges in view, and an approach was taken on the basis of hypotheses which surfaced from the literature (Livingstone, 1998) and from experimenting with various computer models. One of the aims of this experimentation was to derive a number of plausible colour isoluminance models that could be built into the calibration procedure of the experiment which aimed to replicate that of Kveraga, Boshyan & Bar (2007b). The other aim was to find a colour isoluminance candidate that could be tested in a word recognition experiment similar to the first.

The second experiment was thus designed with a particular approach to isoluminance, and the effect of the isoluminance (*P-biasing*) solution seemed to be promising. It also introduced a calibration procedure to adjust the task difficult to participants' performance levels so that it would be adequately challenging.

3. Experiment 3 explored object recognition using *M*- and *P*- biased presentations, in order to develop and test software and procedures to use for the main replication. It was moderately successful, but did not result in clear recognition differences in the main dependent measure (d'). However, the colour manipulation was associated with longer response latencies which may involve a difference in the size at which smaller objects were recognised between the M-biased and P-biased conditions. These are actually error responses, as they are key-presses for non-targets. It is difficult to explain the differential performance between males and females where females out-performed males significantly in the M-biased condition. No explanation comes to hand, and a literature search did not result in any compatible finding or explanation except for a recent finding by Stoesz & Jakobson (2013) which concerned identity and recognition of facial expression and showed an advantage for women. However, this disagrees with research by other authors, such as Rizk-Jackson, Acevedo, Inman, Howieson, Benice & Raber (2006) who concluded from their study of object recognition, that there were no sex-related differences for facial or object recognition. It was curious that this was only seen in the M-biased condition, but not the P-biased, or BW (*high contrast*) condition.

4. Experiment 4 was a replication, with differences in the *P-biasing* solution which tried to approximate the advice of Kveraga (2010). It showed a null result but was otherwise successful in that it demonstrated that the experiment had stability in terms of results in the other, more comparable experimental phases.

Chapter 3. Replication

3.1 Introduction

The outline of the experiment was explained in Section 1.5.3. In brief, a theory of top-down processing was tested by Kveraga, Boshyan & Bar (2007b) in an experiment where subjects were asked to judge a series of line drawings by the criterion 'is the object bigger or smaller than a shoe box?' There were two conditions in the task: the stimuli were either line drawings presented as M-biased (achromatic, with low luminance contrast) or P-biased (red line drawings against a green background, i.e. isoluminant, but with colour contrast). For the chromatically defined stimuli, a procedure was developed to determine the isoluminance point (between the brightness of the red line drawing against the green background) for each participant. This involved presenting the line drawings in rapidly alternating (red and green) colours in the range of 12 - 20 Hz. They found that the flicker frequency of 14 Hz gave the best results. After this isoluminance setting procedure for each subject, all P-biased stimuli were displayed at the determined point of isoluminance.

Their rationale was to test whether the M system when exposed to suitably treated stimuli, enhances perceptual identification by projecting coarse resolution information to the orbitofrontal region of the brain, which according to their model, back-projects information to the inferior temporal cortex aimed at speeding up object recognition (by utilising this back-projected information to constrain the number of possible object representations). The essence of their idea is that the M system conducts rapid, low spatial frequency (LSF) information to the OFC (orbitofrontal cortex) and activates a rough partial guess about what the object might be. There is a back-projection to ITC (inferior temporal cortex), to which the slower P-pathway routes a large amount of fine-grain visual detail and the back-projected information acts as a kind of template to speed up recognition - see Figure 79 for an illustration of their idea.

Both the behavioural and BOLD (Blood Oxygen Level Dependent) analysis supports this idea by showing a response latency advantage for 'Magno-biased' presentations of about 100 ms and improved accuracy (Figure 80).

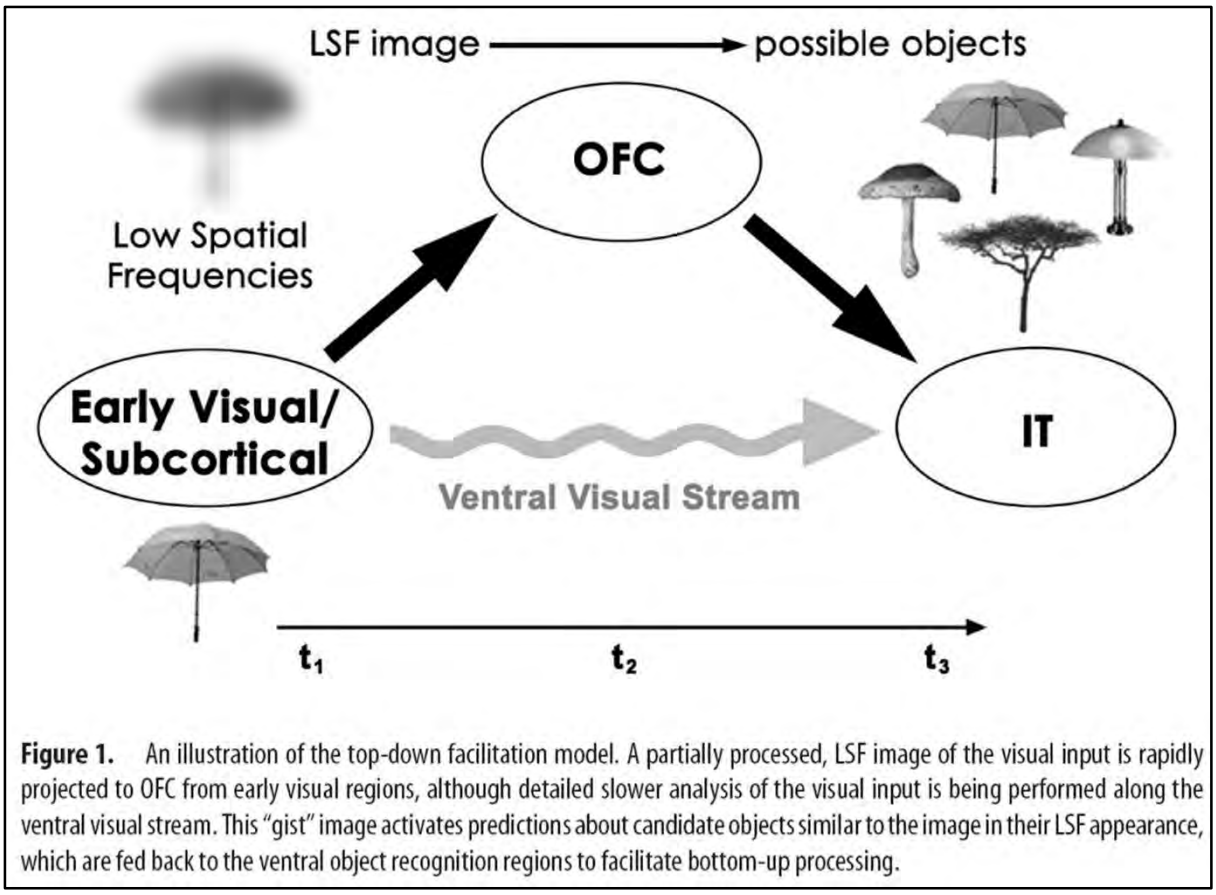


Figure 1. An illustration of the top-down facilitation model. A partially processed, LSF image of the visual input is rapidly projected to OFC from early visual regions, although detailed slower analysis of the visual input is being performed along the ventral visual stream. This "gist" image activates predictions about candidate objects similar to the image in their LSF appearance, which are fed back to the ventral object recognition regions to facilitate bottom-up processing.

Figure 79. LSF objects can activate a limited number of objects - which function as a 'gist' to constrain the number of possible object representations in ITC (Inferior Temporal Cortex). (From Kveraga, et al., (2007b), p. 13233).

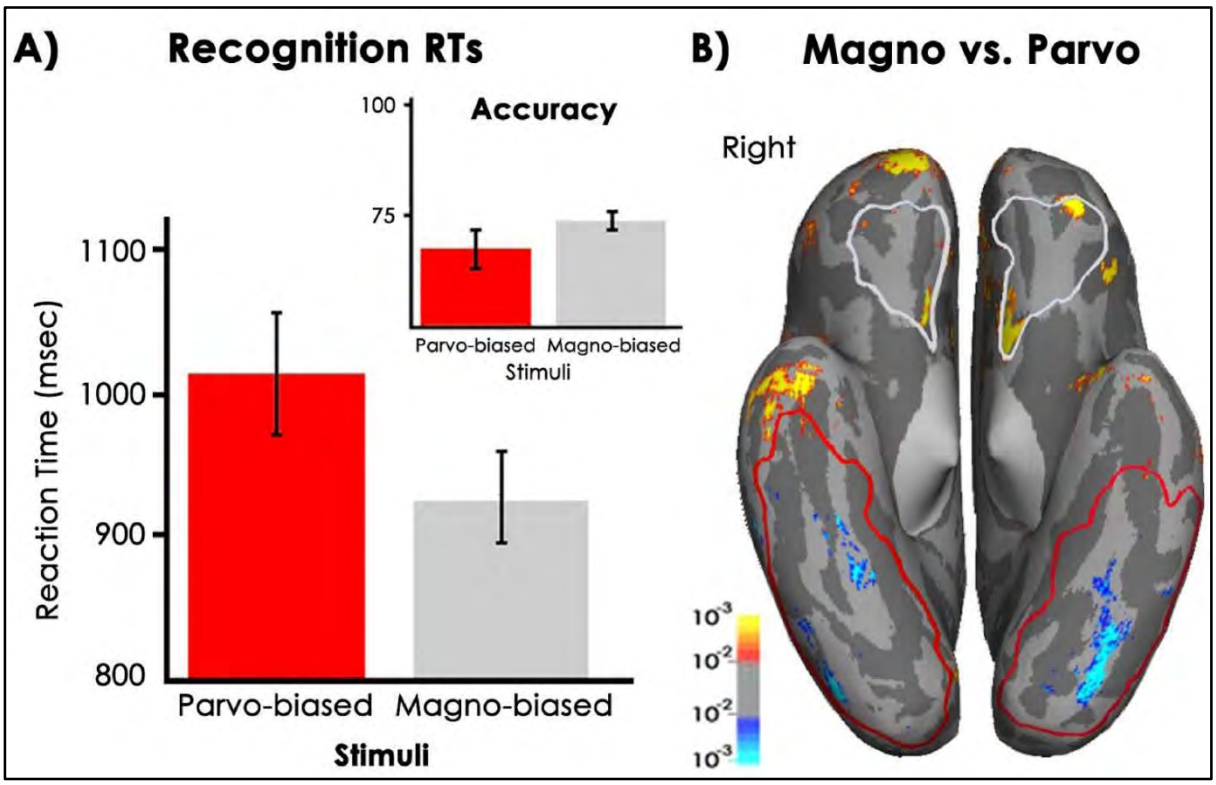


Figure 80. Behavioural and ROI analyses of results support the predicted areas of activation in the ventral temporal areas, and OFC. (from Kveraga, et al., (2007b), p. 13234).

Their main conclusion was that their study provides support for the idea that fast M connections project LSF information to the OFC and the process of object recognition is thereby facilitated by this *top-down* mechanism. They draw potential links between the possibility that there is a subcortical projection of M fibers to the mediodorsal nucleus of the thalamus, the pulvinar nucleus, and from thence to the frontal cortext, and other sub-cortical nuclei (p. 13238).

3.2 Participants

Participants were recruited from an undergraduate class. They did not receive financial remuneration for their participation, but they received course credit for doing the experiment. This experiment took aproximately 35 minutes. After the experiment was completed, they were given a full explanation. All data was coded with an anonymous ID which the participants chose, and recorded for themselves, so that they could identify their own data. Participants were informed that they could stop the experiment at any stage, by pressing a designated key. They were given a general briefing by means of a handout, and their informed consent was obtained. Their anonymity was maintained by logging into a secure *MySQL*® database which was only accessible to the administrator through a user name and password. Participants were given a randomly generated number which was only known to them for identification of their data. There were a total of 65 participants (45 female, 20 male). The mean age for participants was 21.85 years (SD = 3.6).

3.3 Materials

The computer software was written using Embarcadero RAD Studio (2010) in Delphi. This compiles to native Win32 code. Data was written to a MySQL® (Version 5.5.24) database over a network.

The program was installed in the Psychology laboratory at The University of Cape Town (UCT) on 5 workstations and in the Psychology laboratory at the University of KwaZulu-Natal (UKZN) on about 20 workstations. The source code is available in a folder named ‘Appendix 4’ on the media disk and a folder named ‘Appendix 4’ in the Dropbox.

3.3.1 Luminance calibration materials and procedure

The stimuli consisted of 70 line drawings or objects for the luminance calibration phase. These were gleaned from a Word Perfect Clip Art Library, and some were drawn by hand by a post-graduate student. A total of 20 blanks were inserted randomly, as 'catch trials', and these were actual pictures selected at random and then transformed to the appropriate shade (where fore-ground and back-ground were equal and the background at the level expected for the next presentation – see original image, transformed image, and blank transformed image below (Figure 81).

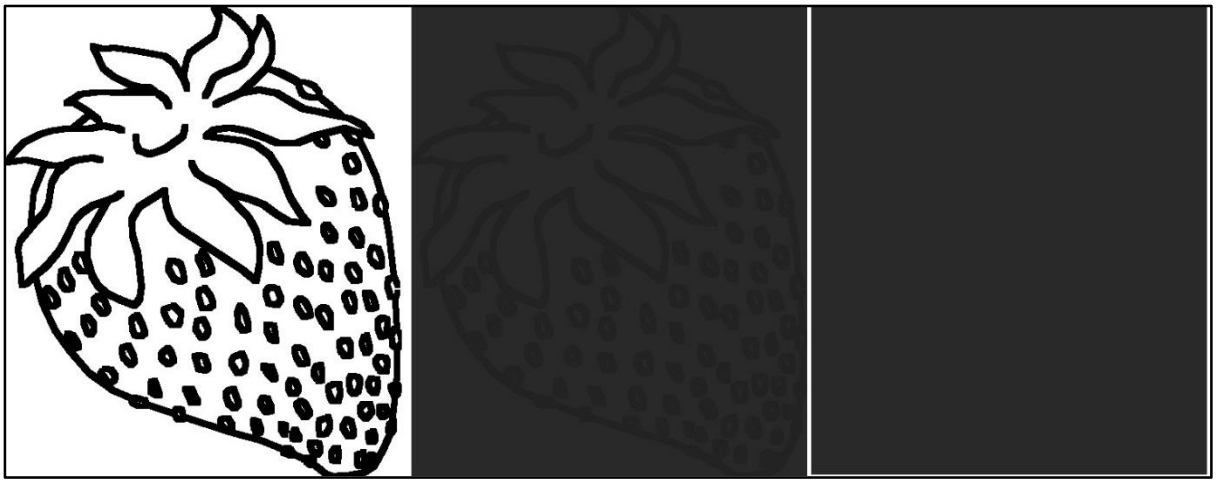


Figure 81. LHS: Original line bitmap drawing, middle panel: transformed bitmap to RGB(41,41,41)/(35,35,35), RHS: blank bitmap image, transformed to RGB(41,41,41).

There were effectively a large number of variables to be controlled and adjusted and it was conducted and administered remotely (1800 km away at the UCT laboratory). The 'Calibration Panel Application' (see Appendix 1, ‘Appendix 1.pdf’ and the accompanying explanation under the heading ‘Replication’) was designed to log into a remote database server (UCT or UKZN) and alter run time values as needed. Every time the experiment ran in any location, it would download and run with these settings.

Figure 82. Bitmap calibrated to upper value RGB(42,42,42) (LHS); lower value RGB(20,20,20) (middle), source line drawing (RHS).

Figure 82 (RHS) shows an example source line drawing in the original contrast configuration. The LH panel shows the same drawing with the detail transformed to RGB(42,42,42) - the upper limit for the brightness detail and the middle panel shows the drawing at the defined lower limit of RGB(20,20,20). In all cases, the surround 'background' was constant at RGB(31,31,31).[3]

3.3.2 Colour isoluminance calibration materials and procedure

The visual interface for this procedure was designed in the Embarcadero RAD Studio IDE using third party components (*TMS® SmoothControls* components). An algorithm was designed to implement the flicker photometry methodology and an icon was presented on the screen in alternating colours. Because of the uncertainty about what colour combinations would work, 5 were chosen. In addition to this the frequency at which they might 'merge', also needed to be determined, so 5 different frequencies were used: 20, 18, 16, 14, 12 Hz. There were thus 5 colour solutions, with 5 frequencies for each, and each presentation consisted of 100 iterations (2500 trials). This was simplified visually, by arranging them as a matrix of colours, and frequencies – see Figure 83 below:

The calibration cycle began at the top LHS at 20 Hz, and the 'smiley face' would flicker between the two colours displayed on the active button in the 'Response Matrix'. Every time a key was pressed, the button would grow and the count would be incremented, to help the user track their responses. The procedure would cycle down through the column in sequence (flickering the icon between the different colour pairs 100 times) and then move to the top of the next column (18 Hz), and continue, until the last (25^{th}) trial block was complete (on the bottom, RHS) and stop. The total number of iterations was 2500 (5 frequency ranges X 5 colour solutions X 100 presentations for each). The duration of the trial blocks depended on the frequency.

[3] Please refer to Appendix 6 for details. A line drawing has values which range from 0 – 255 ($FF in hex, depending on the notation). The back-ground (BG) settings specify that all values <= 5 are transformed to 33 (the 'destination' colour). The fore-ground (FG) setting specify that all values >=250 are transformed to 31 (the 'destination' fore-ground colour). Specifying '31' as the isoluminance value for blank M pictures' means that the back-ground values are transformed to 31, and since this is the same value as the fore-ground, all pixels are the same, and the picture is blank. Unfortunately, using the terms 'fore-ground' and 'back-ground' are slightly confusing in this context. The 'upper' and 'lower' values for the 'M calibration' are the most extreme values above and below the isoluminance point that the picture details will be displayed at.

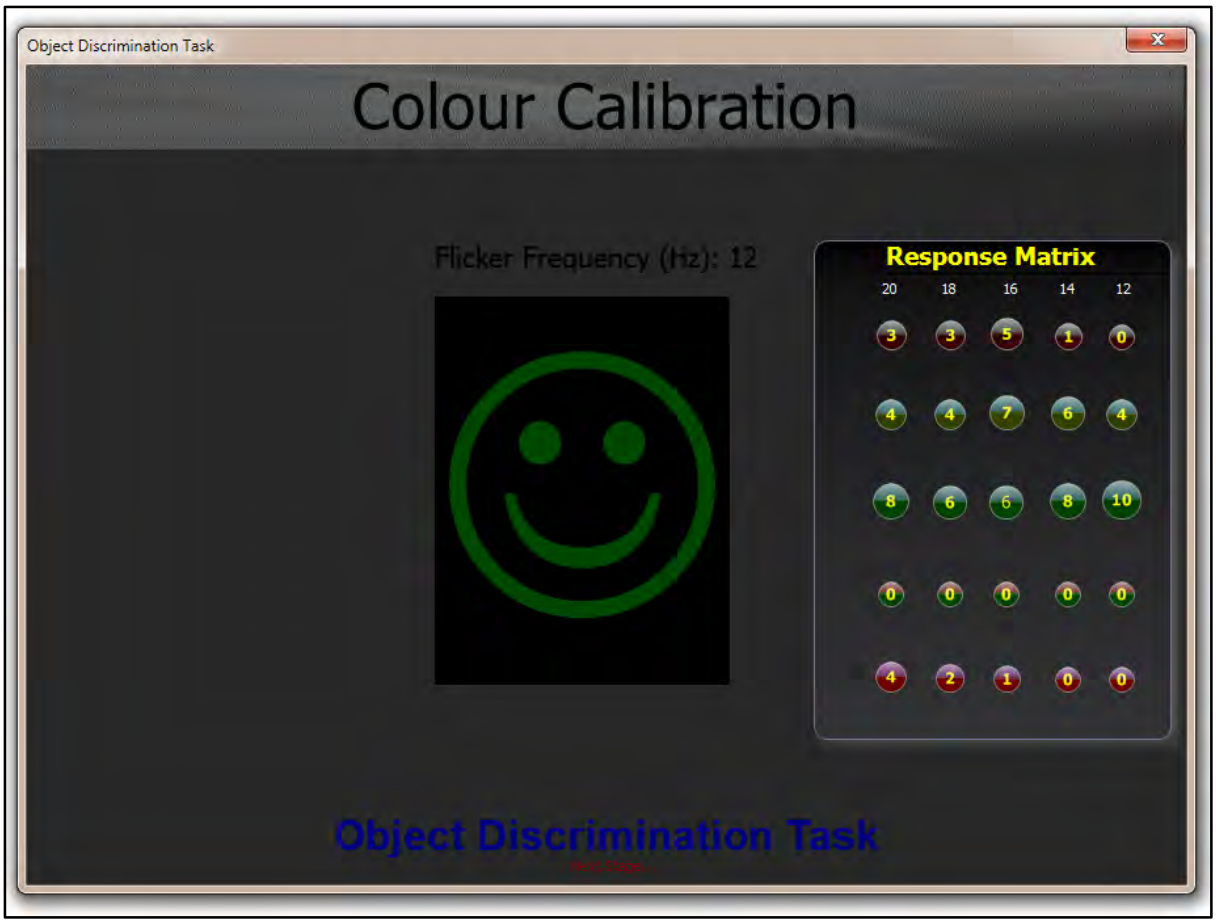


Figure 83. Colour isoluminance calibration display.

A setting controlled whether the result of the colour calibration procedure was used, or default values were enabled. After initial experimentation, I decided to enable the colour calibration procedure (and record the calibration data), but prescribe a default colour solution. Figure 84 illustrates the colours that were prescribed, based on extensive experimentation, and data from Experiment 3.

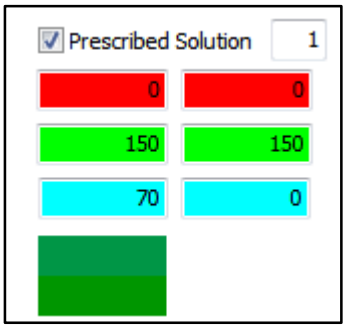


Figure 84. Prescribed Colour Solution values. This two columns show the values between which the image flickered: RGB(0,150,70) / RGB(0,150,0). The colour blocks show these values in the two rows, respectively.

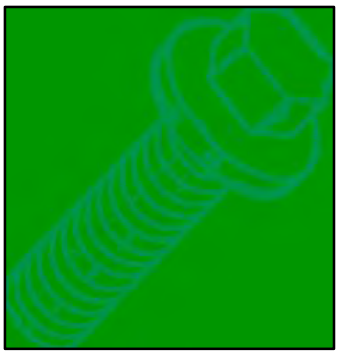


Figure 85. Example of picture transformed to prescribed values.

Figure 85 is an example of a picture transformed to these prescribed values.

In effect, the experiment collected data from the colour isoluminance calibration procedure, but it used the prescribed solution as I had major concerns that applying the colour solution data would result in far too many and perhaps arbitrary solutions resulting from this complex procedure.

3.4 Procedure

Participants first signed an electronic register for course credit, and were then seated at a workstation in the laboratory. They were given a brief verbal explanation of the task, and shown the materials which gave an outline of the procedure. The first computer task they did was to view a simulation of the experiment, using screen captures, and videos of the various procedures. This was prepared as a PowerPoint presentation with embedded video clips. Following this they began the task by logging in with a 9 digit random number that they could self-generate. After successful login, a display screen, Introduction and then demographic and self-rating information was entered (see Appendix 1, 'Appendix 1.pdf').

A short colour vision screening test was presented, which participants could decline if they wished. This involved 9 samples from the Ishihara Colour Vision Test, from a website (http://colorvisiontesting.com/ishihara.html). After the test, feedback was given, and the suggestion was made that if they made more than 1 or 2 errors, they should consider professional testing. See Appendix 1 for colour vision test presentation format.

3.4.1 Luminance contrast calibration procedure

Instructions were given for the Luminance Calibration (see Appendix 1, 'Appendix 1.pdf'). A series of 70 pictures, with 20 randomly spaced blanks, was presented at varying levels of background/foreground contrast (making a total of 90 trials). The first presentation was at the default level of RGB(33,33,33)/(31,31,31) for the fore-ground and back-ground, respectively. This begins at a difficult level, with only a 2 point contrast difference. The sequence begins with a fixation cross appearing for 500 ms, then the stimulus for 1500 ms, followed by the next fixation point for 500 ms. The response latency was measured between presentation and a key-press and recorded for hits and errors. Depending on the participant's performance, the degree of difficulty would change. If s/he did poorly, the contrast level was increased to a maximum of 42 (42-31 = 11 point positive contrast level), or a minimum of 20 (31-20 = 11 point negative contrast level). With poorer performance, the contrast would cycle to more extreme values, above and below the isoluminance point. With better performance, the contrast levels would range within smaller extremes. For each presentation, a line drawing was read from a local file into the program and calibrated to the desired value for the next presentation. These transformations had to be done at runtime and were done in a separate processing thread. See Appendix 1 for an example of a stimulus presentation and examples of feedback given for correct and incorrect responses. Video samples are also provided on the media disk or Dropbox folder ('Replication Video demos and screen captures').

At the end of the calibration procedure, the average (absolute) contrast value at which the images were correctly recognised was calculated and a constant of 2 was added to this. The constant was added after experimentation and this could be varied by changing the calibration table in the database if necessary. All the images were then loaded and transformed to this contrast value, and saved to a separate folder. This took from 10 to 30 seconds, depending on the hardware platform.

3.4.2 Colour isoluminance calibration procedure

The instructions for the colour calibration procedure can be viewed in 'Appendix 1.pdf' in the media disk or Dropbox folder. The program cycled through 5 colour solutions and 5 frequency blocks - see *Colour Calibration Videos* on the media disk or Dropbox folder. Whenever a keystroke was recorded, the counter in the current display was incremented and the size of the button increased, so that participants could keep track of their performance.

When this entire procedure was complete, the program processed all of the stimuli to either the defined defaults, or to the selected colour solution calculated by averaging the RGB values of the solutions with the highest number of key-presses recorded across all frequency blocks. Essentially, this means calculating the average of a matrix of pairs of (RGB) values, after determining the solution with the highest frequency. If there was a tie, or there were no responses, the algorithm could select default settings. The details are available in Appendix 3 (‘**Delphi code to select isoluminance colour pairs and compute values**’**)**. Some examples of the obtained data will be displayed presently.

3.4.3 Main trial stages 1 – 4

The main trial stage was divided into 4 blocks, since one long block is excessively long, and would risk losing the participants’ interest, or focus. A total of 160 images were used that were not shown for the luminance calibration procedure. They were objects that (in real life) were either bigger, or smaller than a shoebox. There were 80 line drawings classified as smaller, and 80 classified as bigger. They were coded by adding an 'S', or 'L' to their names (Smaller/Larger). Following the procedure of Kveraga, *et al.* a total of 33% blank, or null trials were added at random intervals. There were a total of 80 blanks, so the total number of presentations was 160+80 = 240. These were divided equally between 4 blocks, so that each block contained 20 *S* (smaller) images, 20 *L* (larger) images, and 20 blanks, or *N* (null) presentations.

This is a significant task in terms of distributing the items between the blocks in the correct proportions, ordering them randomly, and determining the stimulus Type (*P*- or *M-biased*). The algorithm is available in Appendix 3 (‘**Delphi procedures to extract and randomly distribute picture files between different conditions’)**.[4] Illustrations are given in Appendix 1 of the instructions given for one of the main stage blocks and the concluding information

[4] In brief, it involves creating a file list ('TStringList' in Delphi), and populating it with all the names of the items in the folders concerned. They are then sorted by type, (S/L), and separated into two sub-lists. These lists are shuffled into random order, and divided into 8 smaller lists (4 lists of S, 4 lists of L items with 20 in each). These are assigned symmetrically to 4 arrays, which keep track of the stimulus type (S/L), and shuffle them into a new random order, adding the null trials as well. During this shuffle, a new variable in each record determines randomly whether this will be a *P*- or *M-biased* presentation. Since the process involves randomness, there is usually not a perfect balance of *P/M* trials within each block. However, the blocks tend to balance themselves out evenly. To verify the integrity of this process, the lists are written to a local file. A table (Appendix 1) shows an example item list. The coding for the items is: column 1 is the item name, column 2 is the target code (0 = 'bigger', 1 = 'smaller'), column 3 is the stimulus type (0 = '*M-biased*', 1= '*P-biased*').

given to participants to inform them about the rationale of the experiment. There is also an illustration in Appendix 1 of the result graph which displays the participants' performance information. The results are analysed by means of a *SQL* query, which calls the relevant tables, from which the calculations are done.

3.5 Results

3.5.1 Self-rating of visual reflexes

This rating task was included in the protocol as a curiosity, to see if there were any gender, or other patterns in this self-rating. The response to the question 'How fast do you think your visual reflexes are?' is depicted in Figure 86.

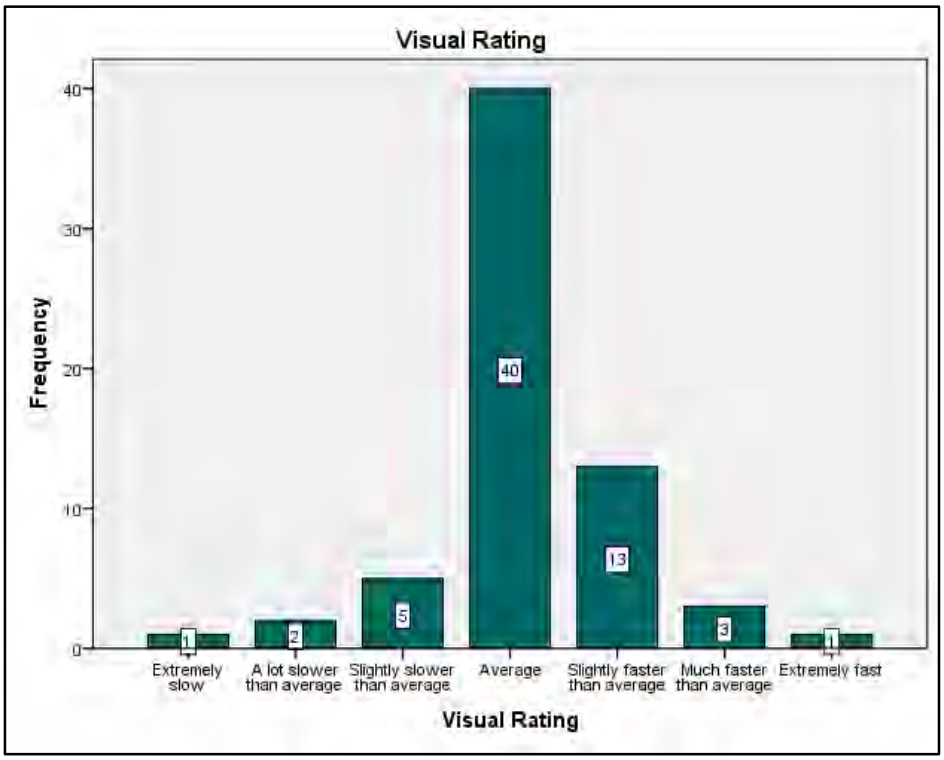


Figure 86. Frequency plot of self-rated speed of visual reflexes.

There was no statistically significant group difference associated with Gender, $F(1) = 0.31$, $p = 0.58$.

3.5.2 Colour vision test

There were no gender differences associated with this measure (i.e. the total score for all 9 items), Gender, $F(1) = 0.009$, $p = 0.924$. One participant scored 1/9 for this test, and the majority (63%) had perfect scores.

Figure 88 shows the frequency of items for which participants made errors (worst for Card 3, and least, for Cards 1, 2, 8). The Y-axis shows the percentage who scored <1, since 1 was correct, and 0 was incorrect. The items are shown in Figure 87 (n.b. not to scale).

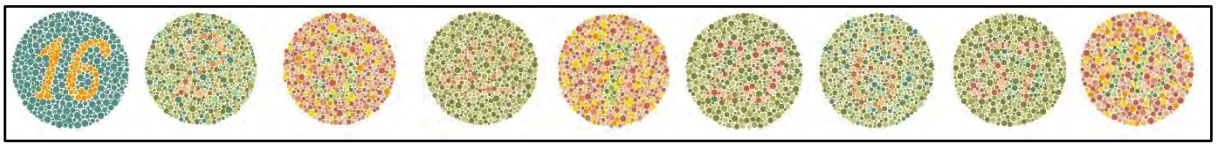


Figure 87. Cards 1 - 9 of the Colour Vision Test (http://colorvisiontesting.com/ishihara.html).

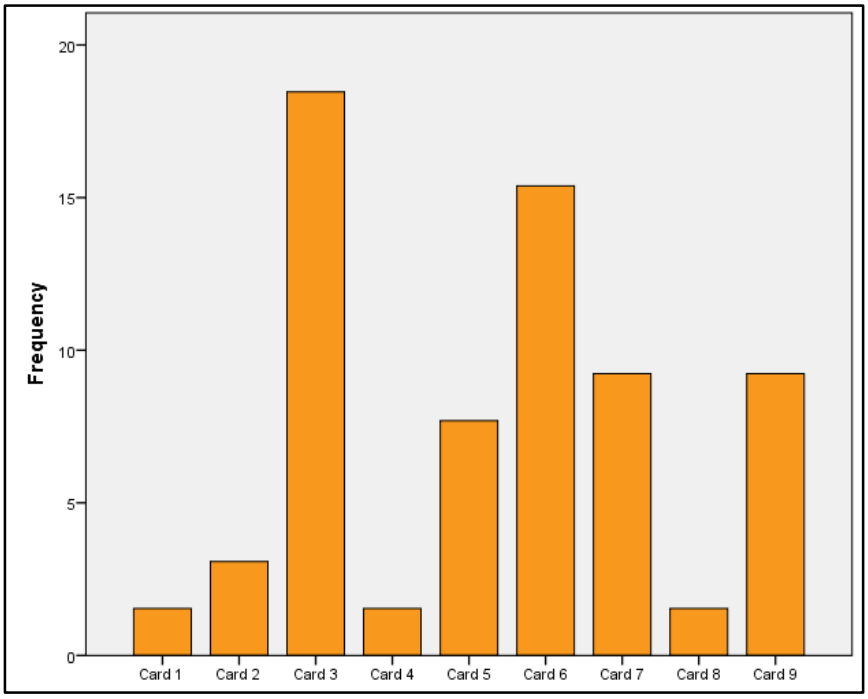


Figure 88. Frequency of items that were answered incorrectly.

3.5.3 Luminance calibration results

The luminance calibration procedure involves 90 presentations, including 20 randomly spaced non-target (blanks) presented at varying contrast levels. When performance is accurate, and detection is at a good level, the range in which the contrast varies is low, and a regular cycling is seen above and below the zero (isoluminance) point where there is no contrast between the fore-ground and back-ground of the picture. When there is poor performance, the values cycle to greater extremes, so that the participant's performance is facilitated, in order to get fair estimates of the contrast level at which they are able to recognise the stimuli. Figure 89 represents a luminance trial for one participant, with time on the X-axis, and contrast on the Y-axis. The red line is a fitted moving average plot.

There was considerable variability in performance. Figure 90 is slightly confusing in that it is a cumulative plot which sums the responses, but it illustrates both the variability in performance and also the fact that there was a trend towards improvement for most participants, as the cycling pattern became more similar towards trial 46 (about half-way through the procedure).

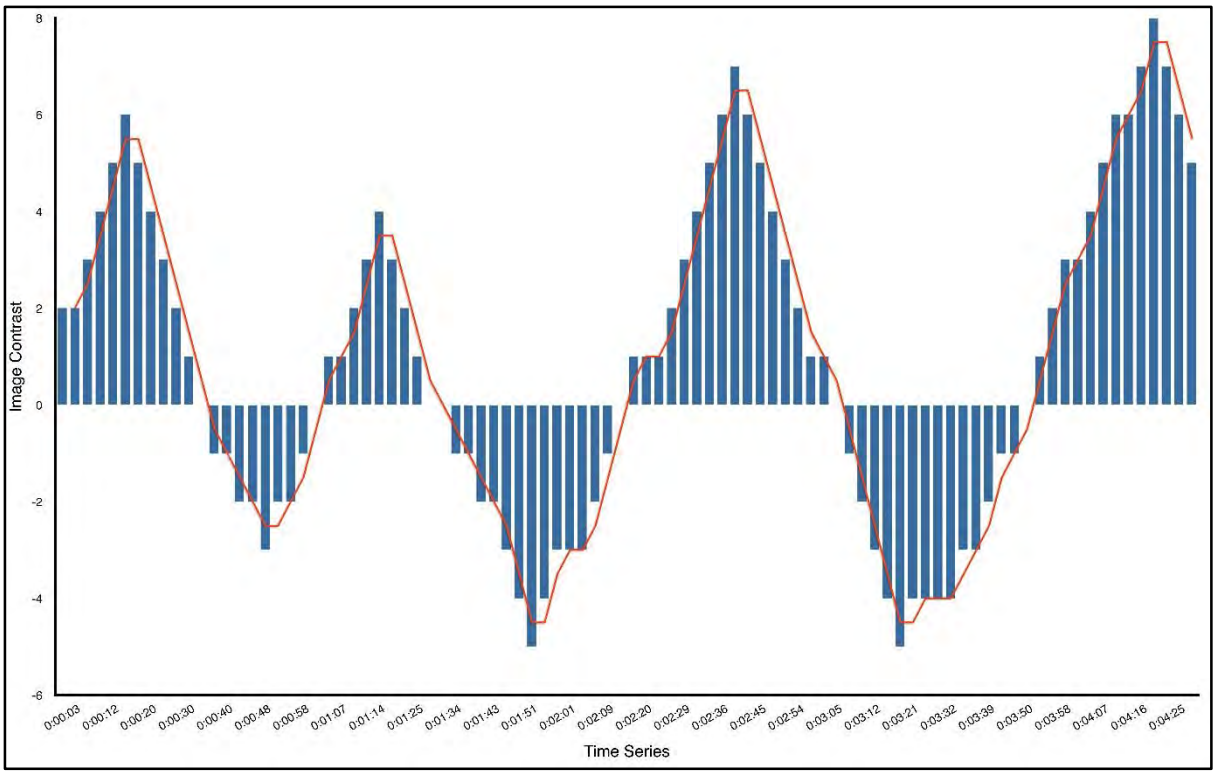


Figure 89. Luminance calibration data for one participant.

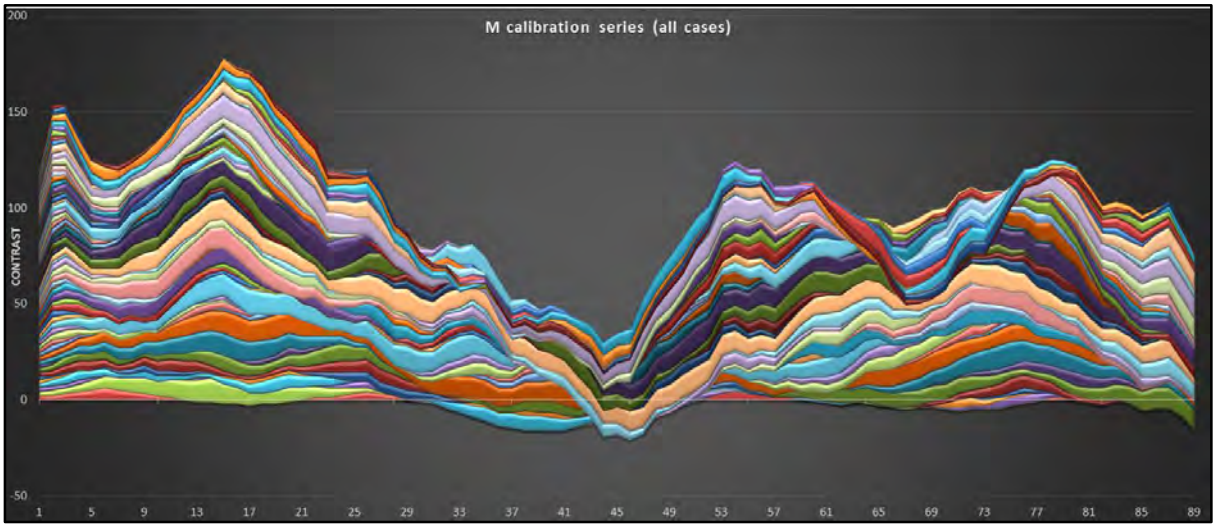


Figure 90. Plot for entire sample represented as cumulative series - note the convergence near trial 45, and some of the symmetries in the pattern of contrast cycling.

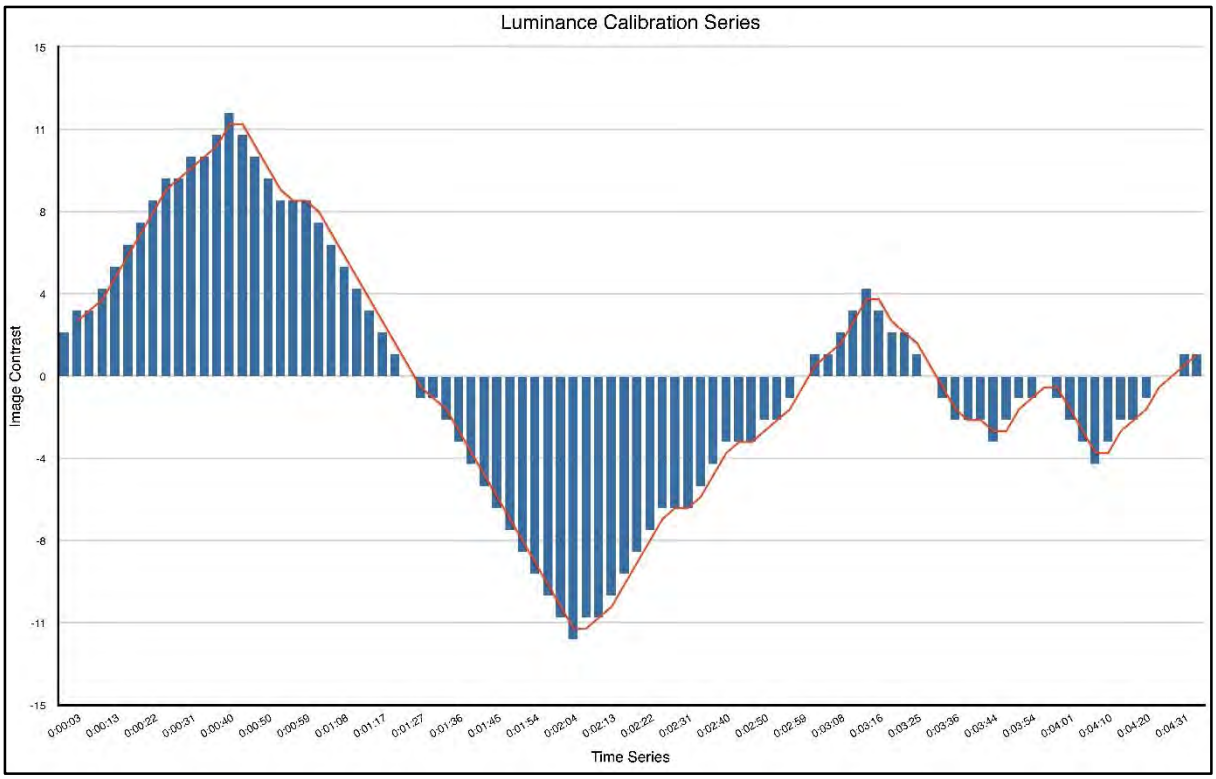


Figure 91. Example plot of luminance calibration showing attenuated cycling associated with consistently improved performance.

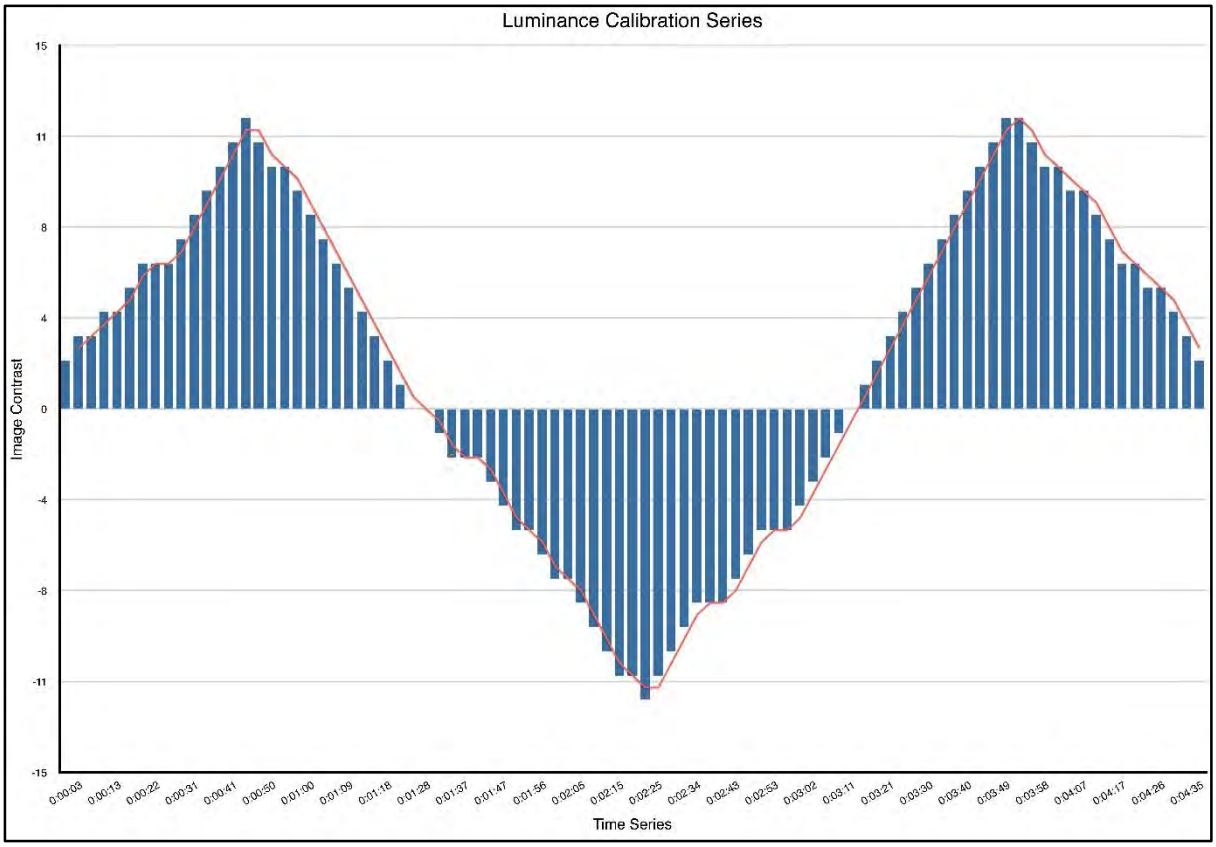


Figure 92. Poor performance showing no improvement over the series.

A variety of different response styles are illustrated above. The plot in Figure 89 reflects a reasonable amount of user input in that there are 3 full positive/negative exposure cycles. There is a tendency for the exposures to drift towards higher extremes in both the positive and negative direction. The series in Figure 91 shows good user interaction and attenuation of the exposure extremes due to good detection performance. Little user interaction (or few correct responses) is evident in the series in Figure 92 and the calibration values cycle to the pre-set boundary values. Another way of examining performance patterns is displayed below:

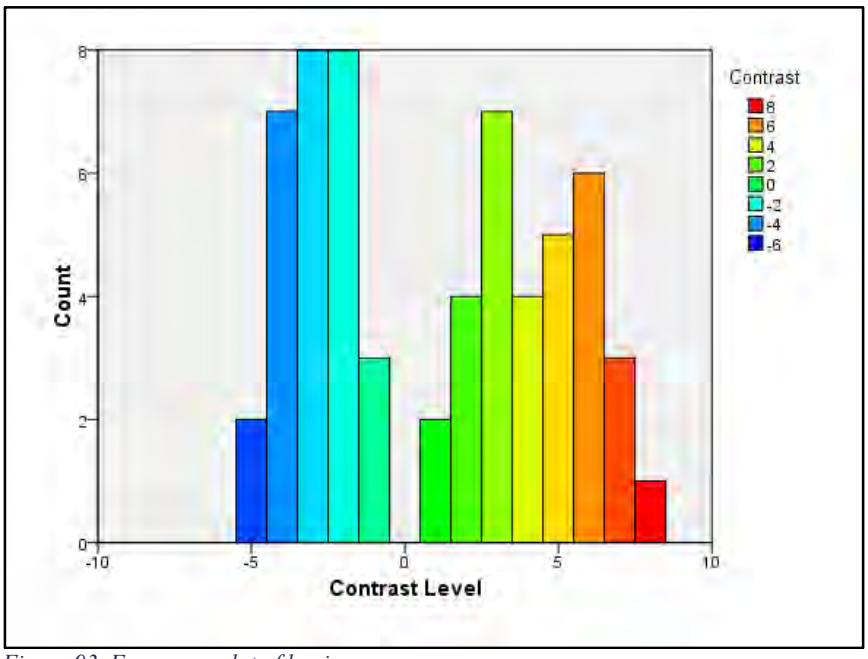


Figure 93. Frequency plot of luminance exposures.

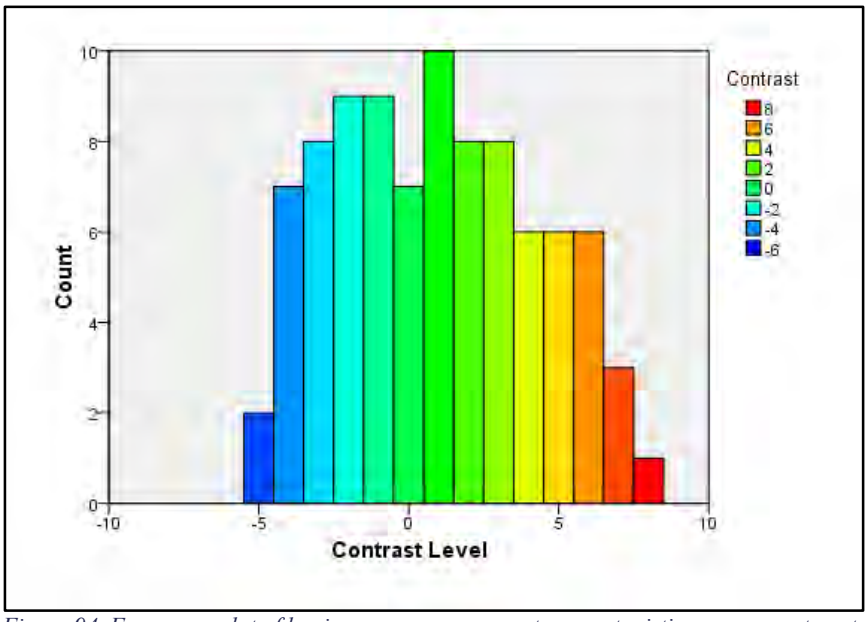


Figure 94. Frequency plot of luminance exposures - note opportunistic responses at contrast level 0.

Figures 93 – 94 show the relative frequency of responses (hits) at various luminance contrast levels. Figure 94 shows an overall higher rate of responding and a tendency to respond indiscriminately (7 responses were made for null (blank/no contrast) presentations at the contrast level of 0). This is a more careless response style, compared with the more accurate and careful style shown in Figure 93, where no opportunistic responses are seen at the contrast level of 0.

3.5.3.1 Summary statistics for the luminance calibration procedure

Although 65 participants began the experiment, only 57 valid data sets were obtained. This is because some participants stopped prematurely, or yielded data that was unusable because of low rates of responses.

The average adjustment applied (calculated from the average absolute contrast values at which participants scored hits was 1.6). The final mean adjustment was 3.6, SD =0.96 (1.6 + a constant of 2 = 3.6). The overall d' mean was 4.13 (SD = 1.7).

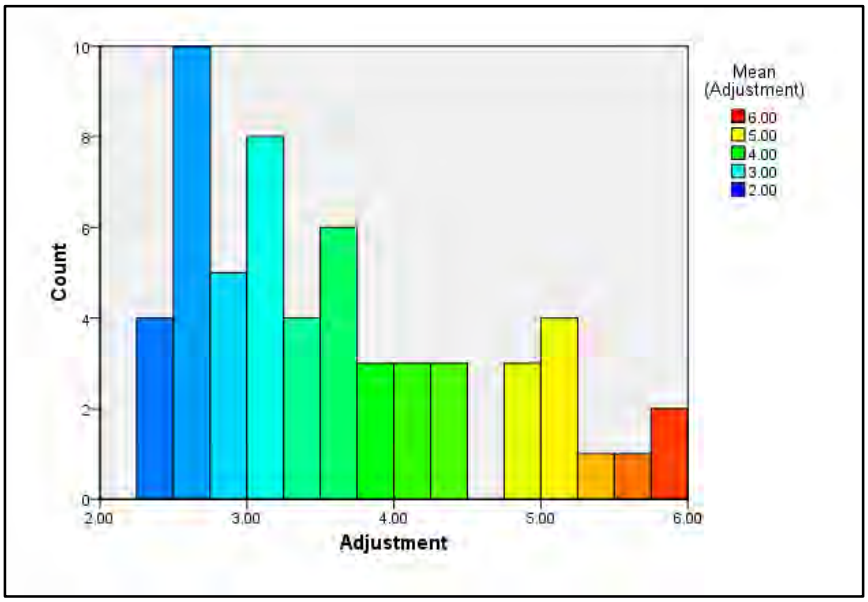


Figure 95. Distribution of adjustment scores.

Figure 95 shows the distribution of adjustment scores. The majority of participants received adjustments ≤ 4, and 11 received adjustments ≥ 5.

The d' mean was at a respectable level (4.13), which shows good detection levels (see Figure 96). The mean target latency was 694.3 ms, and the mean error latency was 370.1 ms (see Figure 97). This suggests that participants' error responses tended to be impulsive and

perhaps that they were motivated to respond quickly. This tendency for error latencies to be lower than target latencies was seen in Experiment 2 and 3 (see Figures 36 and 55).

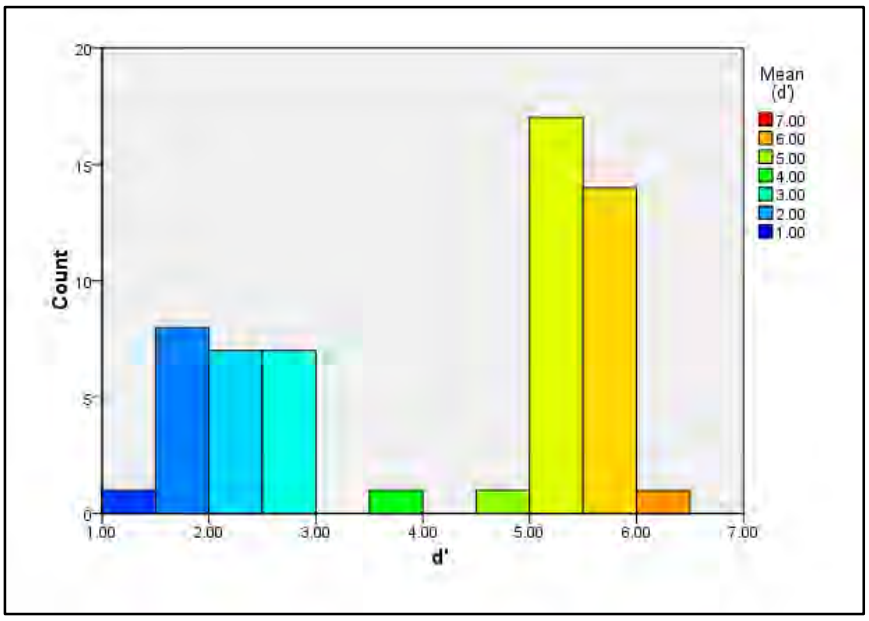


Figure 96. Distribution of d' for luminance calibration procedure.

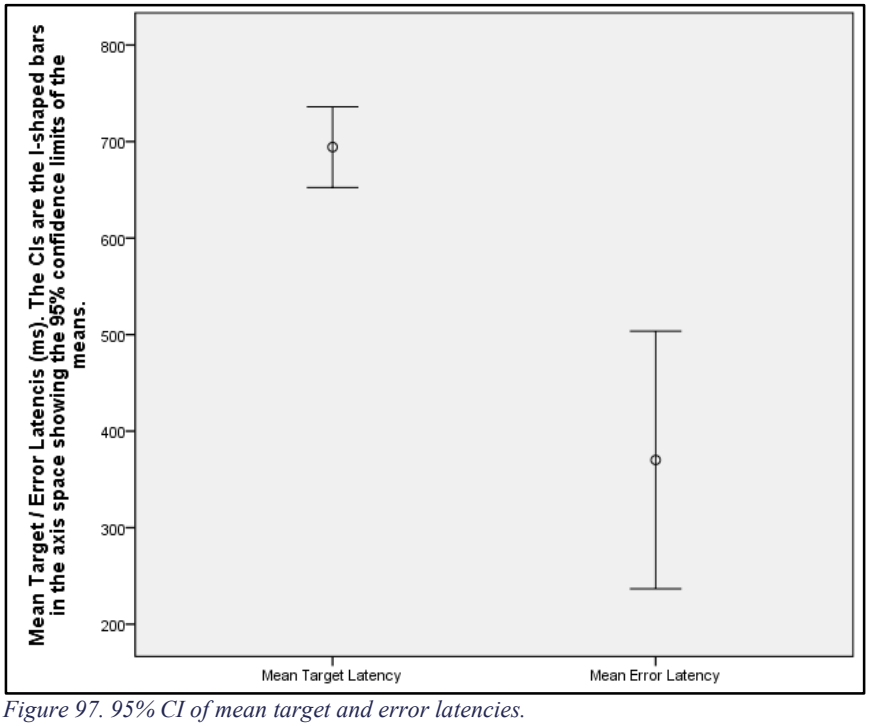


Figure 97. 95% CI of mean target and error latencies.

Lastly, the correlation between the adjustment values applied, and the Hit Rate$_Z$ scores was negative (r = -0.53, p < 0.01 (two-tailed)). This was expected, since poorer performances were expected to be associated with higher adjustment values and vice versa. Interestingly, the Hit Rate$_Z$ scores were also negatively correlated with mean target latency (r = -0.4, p < 0.01 (two-tailed)), and the mean error latency had a very strong negative correlation with d'

scores (r = -88, $p < 0.01$, (two-tailed)) – i.e. the lower the error latency, the higher d' scores tended to be, which tends to discount the explanation that low error latencies were associated with a careless, impulsive response style. There was virtually no correlation between d' and mean response latency (r = 007. $p > 0.05$).

3.5.4 Colour isoluminance calibration results

The most outstanding aspect of the data from the colour calibration tests was the sheer variety. Table 10 shows the colour solutions selected by the algorithm, on the basis of the averaged total number of key-presses for each colour solution, at each frequency. Note that n = 56, as one participant did not record any data, and this case was eliminated from the analysis.

Table 10. Frequency of solutions selected in colour calibration test.

Colour Solution

		Frequency	Percent	Valid Percent	Cumulative Percent
Valid (Solution Number)	1	1	1.8	1.8	1.8
	2	8	14.3	14.3	16.1
	3	21	37.5	37.5	53.6
	4	10	17.9	17.9	71.4
	5	16	28.6	28.6	100.0
	Total	56	100.0	100.0	

Figure 98 illustrates the colour solutions, with the corresponding colour values (RGB), with the colour pairs in the third cell column, each row is labeled 1-5 by the round buttons on the left. The values of the first value of the colour pair for each presentation is shown in the first column and the second value of the colour pair for the initial presentation is shown in column 2. In column 3, the highlighted block shows the colour value which was varied and the range of the variation. For example, in solution 1 (top row) the first value of the colour pair is RGB(30,30,30) and the second initial value of the colour pair is RGB(57,0,0). The third column shows the amount by which the red value of the latter colour is varied, i.e. 20. The block below this is grey (RGB(30,30,30)) and the third block is brown. The stimulus flickered between grey and brown and the red value of the brown colour began at the initial value of 57 and ranged from 57+20, to 57-20.

Figure 98. Defined colour solutions and RGB values.

Figure 99 illustrates a calibration series recorded by a participant where one can see, for instance, that for colour solution 1, colour 1 was held constant at RGB(30,30,30), and the participant selected values between RGB(51,0,0) and RGB(65,0,0). This illustration is a screen capture from a program written by D. Mansfield that was designed to sort and display the actual colour values for each cell. It is ordered by 'Colour Solution', 'Timing' and 'Number'.

Number	Timing	Colour Solution	C1 Red	C1 Green	C1 Blue	C2 Red	C2 Green	C2 Blue	Key Pressed
15	62	1	30	30	30	51	0	0	1
16	62	1	30	30	30	65	0	0	1
17	62	1	30	30	30	58	0	0	1
28	71	1	30	30	30	55	0	0	1
43	83	1	30	30	30	52	0	0	1
44	83	1	30	30	30	58	0	0	1
45	83	1	30	30	30	53	0	0	1
46	83	1	30	30	30	56	0	0	1
1	50	2	0	65	65	45	65	0	1
2	50	2	0	65	65	46	65	0	1
3	50	2	0	65	65	52	65	0	1
4	55	2	0	65	65	44	65	0	1
5	55	2	0	65	65	44	65	0	1
18	62	2	0	65	65	49	65	0	1
29	71	2	0	65	65	47	65	0	1
30	71	2	0	65	65	44	65	0	1
31	71	2	0	65	65	45	65	0	1
32	71	2	0	65	65	60	65	0	1
33	71	2	0	65	65	52	65	0	1
34	71	2	0	65	65	51	65	0	1
35	71	2	0	65	65	52	65	0	1
36	71	2	0	65	65	43	65	0	1
37	71	2	0	65	65	50	65	0	1
47	83	2	0	65	65	44	65	0	1
48	83	2	0	65	65	52	65	0	1
49	83	2	0	65	65	43	65	0	1
6	55	3	0	60	60	0	68	0	1
7	55	3	0	60	60	0	62	0	1
8	55	3	0	60	60	0	69	0	1
19	62	3	0	60	60	0	62	0	1
20	62	3	0	60	60	0	69	0	1
21	62	3	0	60	60	0	62	0	1
22	62	3	0	60	60	0	69	0	1
38	71	3	0	60	60	0	63	0	1
39	71	3	0	60	60	0	67	0	1
50	83	3	0	60	60	0	65	0	1
51	83	3	0	60	60	0	71	0	1
9	55	4	74	0	0	0	70	0	1
23	62	4	69	0	0	0	70	0	1
40	71	4	87	0	0	0	70	0	1
41	71	4	73	0	0	0	70	0	1
52	83	4	74	0	0	0	70	0	1
53	83	4	82	0	0	0	70	0	1
54	83	4	86	0	0	0	70	0	1
55	83	4	69	0	0	0	70	0	1
56	83	4	75	0	0	0	70	0	1
10	55	5	75	0	75	107	0	0	1
11	55	5	75	0	75	103	0	0	1
12	55	5	75	0	75	109	0	0	1
13	55	5	75	0	75	117	0	0	1
14	55	5	75	0	75	116	0	0	1
24	62	5	75	0	75	104	0	0	1
25	62	5	75	0	75	101	0	0	1
26	62	5	75	0	75	120	0	0	1
27	62	5	75	0	75	119	0	0	1
42	71	5	75	0	75	109	0	0	1
57	83	5	75	0	75	111	0	0	1
58	83	5	75	0	75	105	0	0	1

Figure 99. Colour calibration data series illustrating colour values recorded by 1 participant.

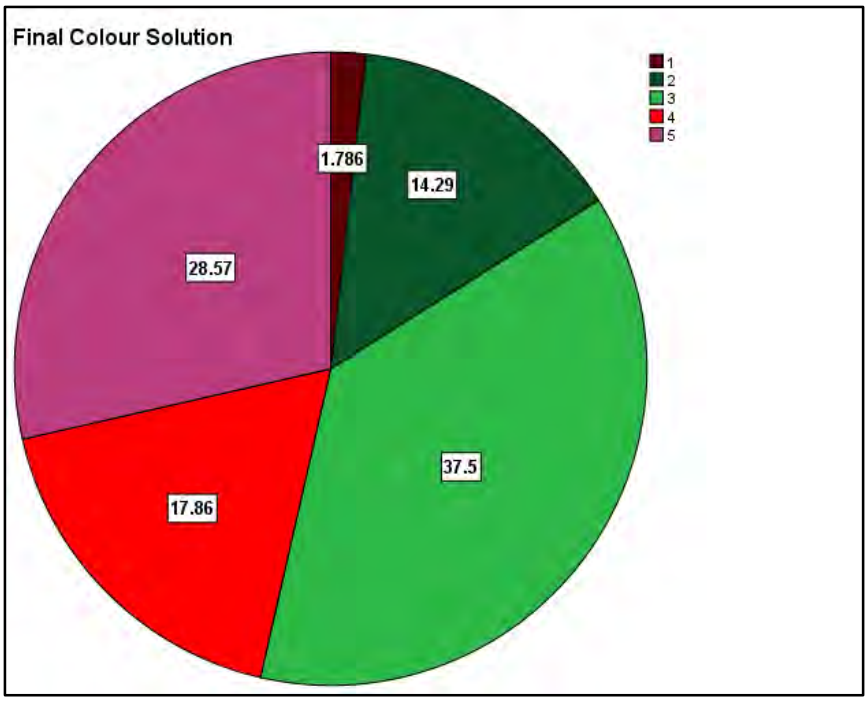


Figure 100. Colour Calibration Solution frequencies pie chart.

Figure 100 shows that colour solution 3 was the most frequently selected one (37.5%). Colour solution 5 average frequency was less than 10% lower than solution 3.

Figure 101 shows the averaged colour for all participants for each solution. The values in each block in Figure 101 are the mean colour solution values for all participants, showing the actual colours these means represent. The most frequently selected solution was colour solution 3 and the recorded average values were exactly the same as the prescribed value: RGB(0,66,0).

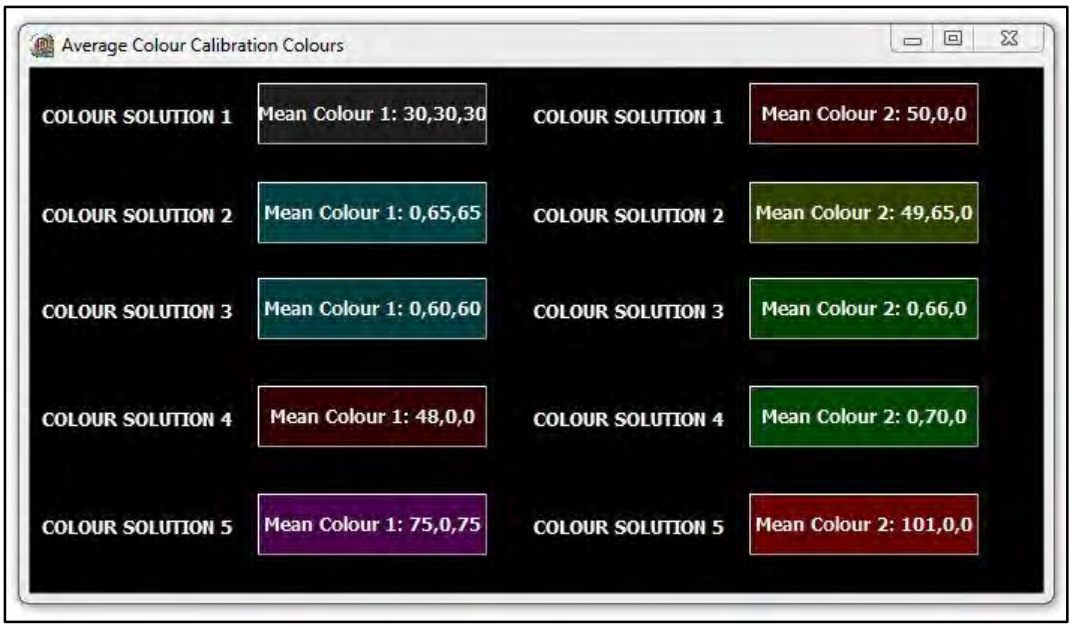


Figure 101. Summary of colour summaries (all cases), displayed in actual colours.

3.5.5 Main Trial Blocks

Following the calibration blocks, there was a main testing stage which was split into 4 separate trials consisting of 20 small (non-target pictures), 20 large (target pictures) and 20 blanks. The presentation colour of both targets and non-targets was randomly varied between P- and M-biased, which was also the case for the blanks. Figure 102 shows the composition of each block, excluding blanks (cases on X-axis are truncated IDs).

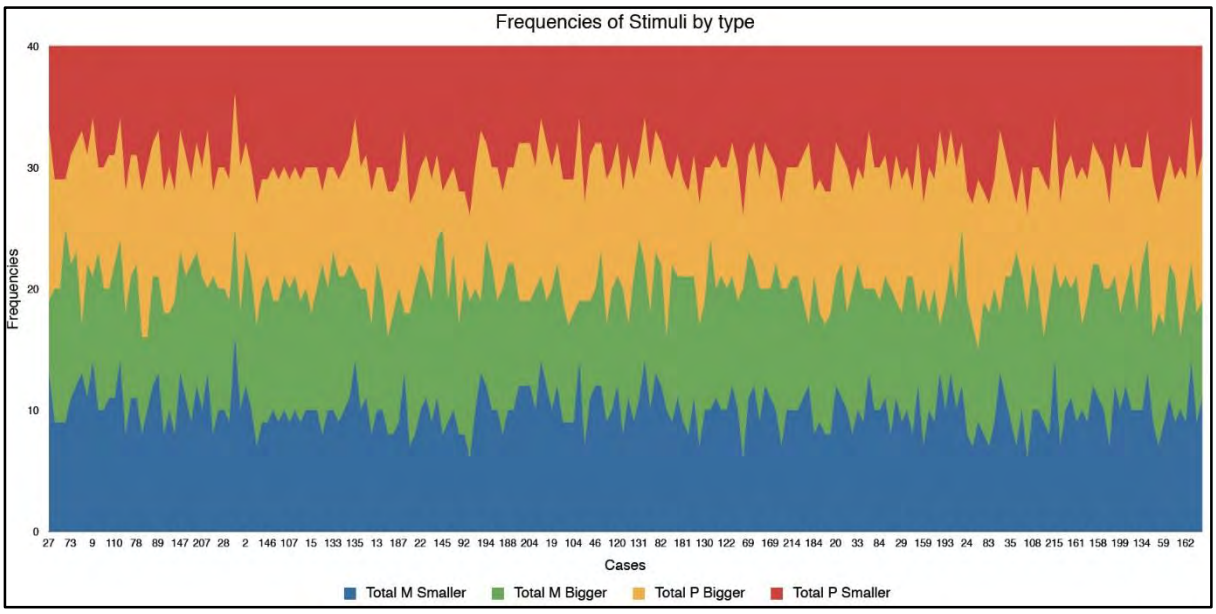


Figure 102. Stimulus composition of trial block, excluding nulls. Y-axis max. = 40.

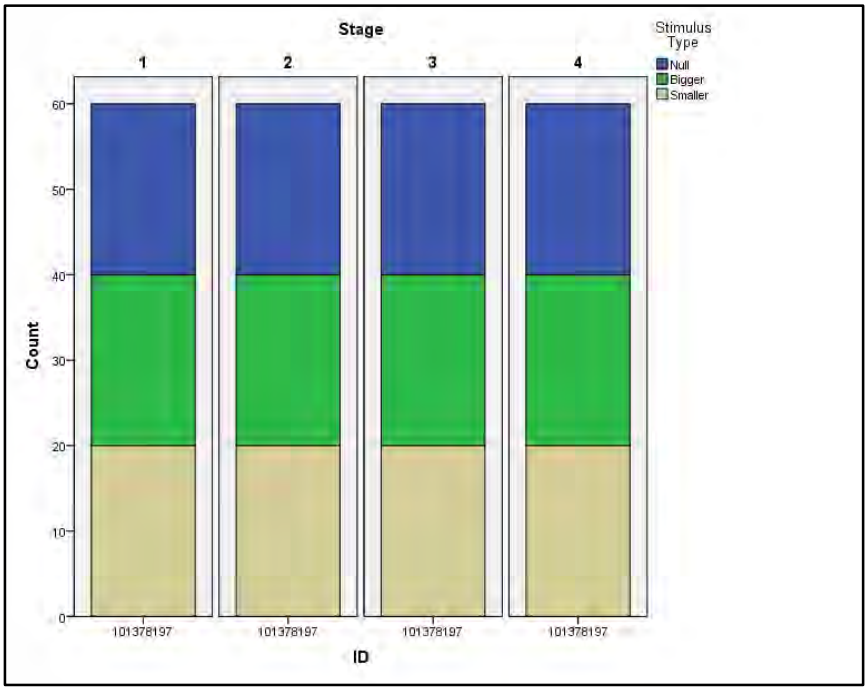


Figure 103. Main trial blocks 1-4, composition for one participant.

The trial blocks were separated into 4 blocks which were identical in terms of the number of items, and item types, but the items were randomly distributed between these 4 blocks, so they did not repeat. Each block contained 20 targets ('Bigger'), 20 non-targets ('Smaller'), and 20 null presentations – see Fig. 103.

Because the stimuli were randomly distributed between the M- / P-biased and Target / Non-target condition, the distribution of these different presentation conditions varied between subjects (Figure 104).

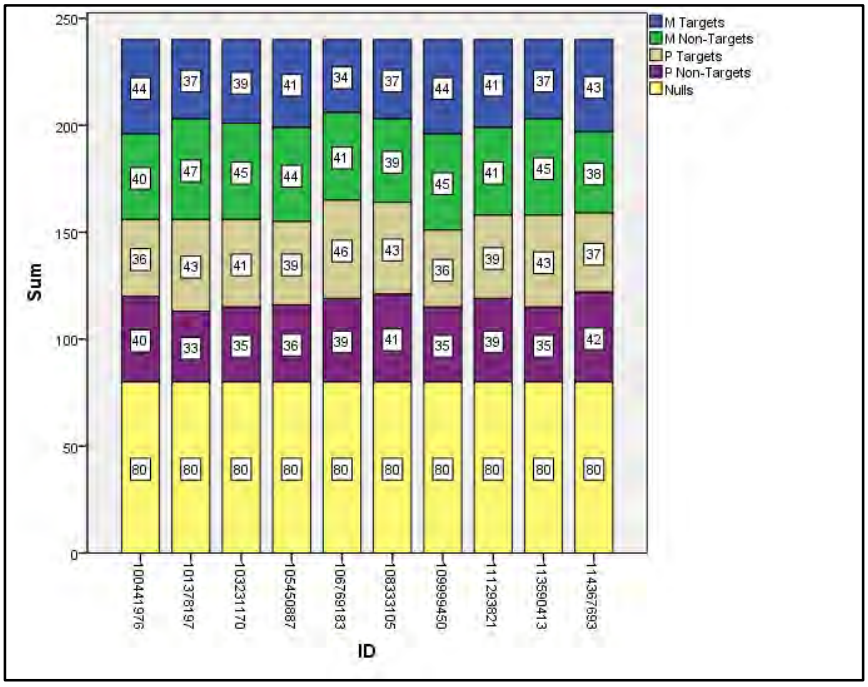


Figure 104. Distribution of stimulus type for sum of 4 blocks (10 participants).

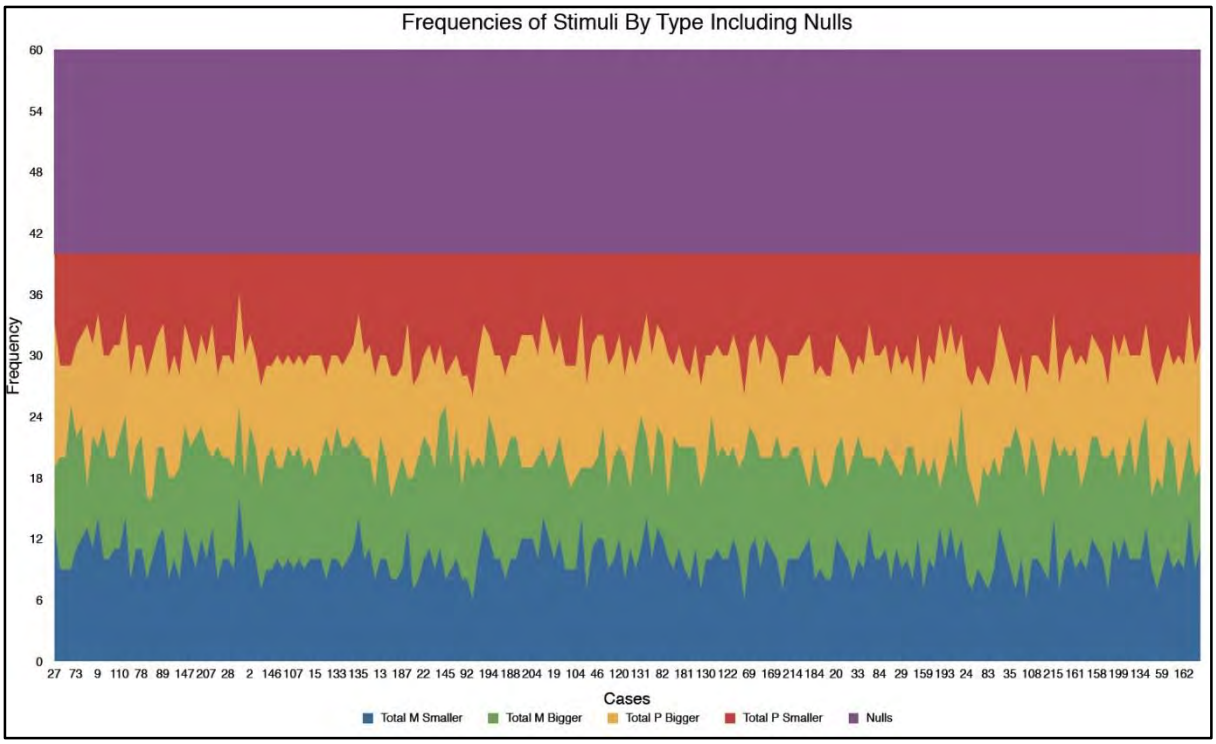


Figure 105. Stimulus composition of trial blocks for entire sample, including nulls. The x-asis represents cases (represented by numbers).

Figure 105 shows the distribution of stimulus item types for each block, including nulls for the entire sample.

3.5.5.1 Data considerations

Each 'case' in the final data set summarises 4 block of trials and the distribution is odd. For the sake of brevity, this will not be displayed. The solution was to aggregate the cases, i.e. to average d' and various other parameters for each of the 4 blocks of each participant. The resulting distribution was satisfactory and most variables did not deviate from normality. The results are unchanged in terms of all of the patterns of differences. A total of 54 cases were selected, as some participants stopped prematurely, or recorded little or no data.

Importantly, the d' scores were normally distributed, although some of the response latencies were not – notably the mean target latency for the P-biased condition, and the mean error latencies.

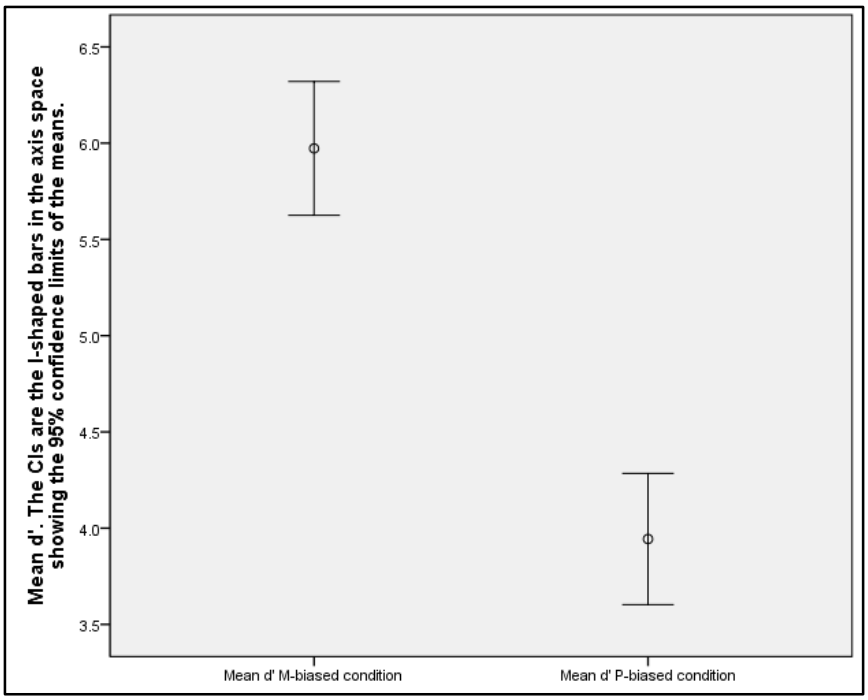


Figure 106. 95% CI of d' means for M and P conditions.

The results show a clear difference between the M- and P-biased conditions, with d' at a significantly higher level in the M condition (Figure 106). A paired samples T-Test is significant, $t(53) = 8.9$, $p = <0.001$ (two-tailed). This is a robust, stable difference, confirmed by significant non-parametric tests. Wilcoxon Signed Ranks Test, shows a significant difference $Z(53) = 5.96$, $p <0.001$ (exact significance, two tailed).

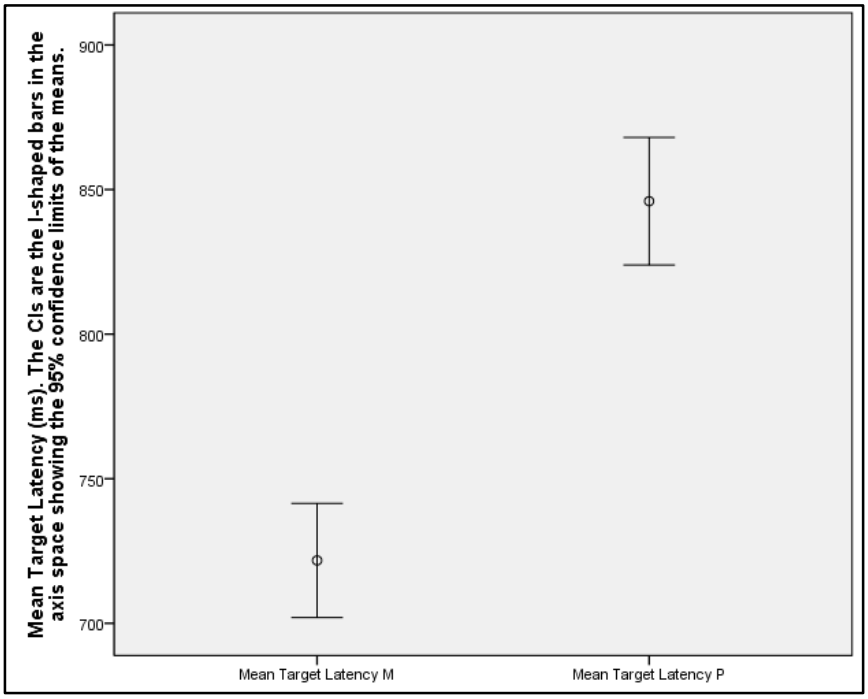


Figure 107. 95% CI plot of mean target latency M and P.

Figure 107 shows the 95% CI of the means for M- and P-biased target latencies. The Wilcoxon Signed Ranks Test is significant for a comparison of the M and P response latencies, Z (53) = -6.67, p <0.001, and also the paired T-test, t(53) = -10.7, p <0.001 (two-tailed). The difference is about 123 ms (846/723 ms) for M- and P-biased stimuli respectively. Importantly, this is compatible with the reaction time difference associated with the exposure conditions found by Kveraga *et al.* (1997b) which is approximately 100 ms. There is a mean difference of about 200 ms between the mean latencies from this experiment, and that of Kveraga, *et al.* This may be explained by the different software (Matlab), and hardware systems (Mac OS 9), but despite the general difference, there is still consistency in terms of the response time differentials associated with the different presentation types.

Comparison with the results of Kveraga, *et al.*, is complicated by fact that different measures were used. This study used signal detection theory, which takes into account the 4 possibilities of Hits, Misses, False Alarms, and Correct Negative responses, and transforms these into standardised values from which the detection measure d' is calculated. Kveraga, *et al.* simply reported detection accuracy percentage. However, although the measures are different, they do correspond with each other in the correct direction (superior detection accuracy for M- vs. P-biased presentations – although this was not statistically significant in the original experiment).

3.6 Discussion

Designing and implementing calibration procedures for luminance contrast, and colour isoluminance was challenging, as there are many complexities and variables to control for. It was challenging for participants to learn how to do the two calibration tasks and then the main experiment with minimal intervention from the researcher. The initial *Powerpoint* presentation contained a lot of information which participants mostly seemed to gloss over. Participants relied more on the instructions and prompts in the program and from feedback and there were some participants who needed support and some extra tuition. In general, the rapid learning and competence generally displayed by participants contributed to the success of the experiment.

It was difficult to predict the 'correct' colour solution and to know whether participants would manage the complex colour calibration procedure. It is difficult to say whether all the solutions generated were valid, but the fact that the solution with the highest frequency was

chosen means that this was fortuitous. If a range of different colour solutions were used, the results may have been diverse and uninterpretable.

The results of this experiment are consistent with those of Kveraga *et al.* (2007b), in terms of response reaction time and accuracy and it appears to validly replicate their experiment. There was some deviation in the heterochromatic flicker photometry methodology.

The practical question about what colours to use for heterochromatic flicker photometry seems to depend on theoretical developments. This was discussed in Section 1.4.6 where it was noted that Lee & Sun (2009) suggested the possibility of chromatic input to magnocellular cells (especially with red/green contrasts). The general conclusion of this discussion was that red/green isoluminance solutions probably occur in a narrow range and are dependent on ambient light conditions (probably towards the mesopic range where the magnocellular system receives more rod-based input).

Another aspect which seems consistent with this is the finding by Crook *et al.* (2008) of what appeared to be colour opponent input to the superior colliculus in rhesus monkeys which were carried by a magnocellular projection. The finding by White, *et al.* (2009) of the availability of chromatic information to brain systems which influence saccadic eye movements (and which probably facilitate visual searching) also seems to fit with this finding.

Although the P-biased stimuli were more difficult to recognize accurately and response latencies were longer, they were certainly not invisible. The mean d' for this condition was in the region of 4, which is relatively good and discrimination is well above the level of chance.

The fact that fewer trials were administered in this experiment, than that of Kveraga, *et al.* (240, including nulls, vs. 720, including nulls, respectively) is offset by the greater number of participants (54 vs. 12) respectively. The experiment was also done in a more everyday setting (participants were seated at a workstation in a computer laboratory, vs. lying inside an fMRI scanner) and this potentially eliminates a range of artefacts. There is also an implication in terms of the ambient light levels: the conditions for this experiment were close to photopic, whereas the conditions inside an fMRI scanner were possibly darker (perhaps closer to mesopic).

In the current experiment there was possibly less 'coaching' involved, as many participants received minimal verbal briefing, whereas in the original experiment 'subjects' had to be '...given instructions about all parts of the experiment verbally before entering the scanner...' (Kveraga *et al,* 2007b, p. 13234). This is certainly understandable since it would not be possible to respond to queries during scanning.

Interestingly, almost all of the participants in the original study reported that the P-type stimuli were easier to see. An omission in this replication is that in the post-experimental debriefing, this question was not posed to participants, so this is unknown.

The major technical challenge was in the fact that it was difficult to know how to precisely replicate the chromatic (P) calibration procedure, since the images were projected by an LCD projector onto a translucent LCD screen inside the scanner. The theoretical challenge lies in the possibility that there are 'M' (green) and/or 'L' (red) cone inputs to the parasol cells at some signal processing stage in the retina, or beyond.

Chapter 4. Model Application in the Implicit Association Test

4.1 General introduction

Having worked quite extensively on aspects of M- vs. P-system attributes, this phase of the research focussed on building on the findings, especially the successful replication of the research of Kveraga *et al*. (2007b). The approach focussed on testing hypotheses using the IAT. Three experiments will be described:

1. IAT 'species – bias' experiment.
2. IAT race experiment.
3. IAT race experiment 2 which uses an approach similar to that used in the replication of the Kveraga, *et al.* (2007b) experiment to test hypotheses about whether the M system is involved in recognising race-related facial features in the same way that it is thought to be involved in the object recognition.

4.2 IAT Experiment 1: Species bias test

4.2.1 Introduction

The Species IAT experiment, although published, and now firmly established in the history of the IAT, was '...intended more as a demonstration showing that the method could work than for actual interest in studying individual differences...' (Greenwald, A., 27 August, 2012, personal communication). According to Greenwald, it was used to demonstrate differences between entomology PhD students and Psychology PhD students, although that study was not published [1]. There are robust effects associated with this IAT, with the almost ubiquitous finding of an automatic preference for flowers, over insects. The first publication of data from an experiment using these targets was by Greenwald, McGee, & Schwartz (1998), and this was part of a series of implicit experiments which examined psychometric aspects, reliability, and compared similar kinds of tests (musical instruments vs. weapons; Japanese Americans vs. Korean Americans). The authors note that the experiment was almost unaffected by variations of inter-trial intervals, the set size of categories used in the discrimination tasks, or the assignment of response keys (left or right) to the pleasant category, or the position of the

[1] Interestingly, Greenwald stated that this 'automatic preference' was not seen in entomology students (*ibid.*).

trials which gave rise to the IAT measure in either replication. The classic, original IAT experiment consists of 5 blocks of trials – see Table 6. The IAT procedure consists of (historically) 5 blocks of trials, in which the participant is required to rapidly sort lists of exemplars consisting of two concept categories (e.g. Black and White), and two attribute categories (pleasant and unpleasant). In the case of the former (concept category), this might be done in the first block. In this block, the words (or picture exemplars) are presented and the participant has to press a certain key ('q') whenever a Black name or picture is presented, and another ('p') key whenever a White name or picture is presented. This is indicated by the '∘/•' in Table 11. The same procedure is followed in the second block of trials, except that exemplars of two attribute dimensions are presented. The participant responds with similar key-presses shown in Table 6. In the third block 'Initial combined task' the participant has to respond with the same key for both Black and 'pleasant' words or pictures (e.g. 'q' and the opposite key e.g. ('p') for both White and 'unpleasant' words or pictures. Block 4 reverses the key assignment for Block 1, and Block 5 reverses the key assignment of Block 3. Essentially, the concept and attribute categories are reversed between the critical blocks (3 & 5). All response latencies (the amount of time taken to respond to each stimulus presentation) is recorded, and the difference in mean response latency between these critical blocks is used to infer the relative ease of sorting the concept and attribute categories, and is understood as an implicit measure of association strength, or more crudely, relative preference or bias.

Table 11. Schematic Illustration of IAT 'Race' test procedure (from Greenwald, McGee & Schwartz, (1998).

Sequence	1	2	3	4	5
Task description	Initial target-concept discrimination	Associated attribute discrimination	Initial combined task	Reversed target-concept discrimination	Reversed combined task
Task instructions	• Black White •	• pleasant unpleasant •	• Black • pleasant White • unpleasant •	Black • • White	Black • • pleasant • White unpleasant •
Sample Stimuli	Meredith ∘ ∘ Latonya ∘Shavonn Heather ∘ ∘ Tashika Katie ∘ Betsy ∘ ∘ Ebony	∘ lucky ∘ honor poison ∘ grief ∘ ∘ gift disaster ∘ ∘ happy hatred ∘	∘ Jasmine ∘ pleasure Peggy ∘ evil ∘ Colleen ∘ ∘ miracle ∘ Temeka bomb ∘	∘ Courtney ∘ Stephany Shereen ∘ ∘ Sue-Ellen Tia ∘ Sharise ∘ ∘ Megan Nichelle ∘	∘ peace Latisha ∘ filth ∘ ∘ Lauren ∘ rainbow Shanise ∘ accident ∘ ∘ Nancy

Computation of the 'IAT Effect' is done by subtracting the mean item response latency of block 3 ('Initial combined task') from the mean item response latency of block 5 ('Reversed combined task'). Initially, the untransformed latencies were used and at a later stage, a variety of transformations were used, until in 2003 (Greenwald, Nosek & Banaji, 2003), the reciprocal, log transformations and other parameter computations were dropped in favour of the *d* measure[5]. Error response treatments were also systematically compared, and various options considered. The major consideration was the correlation between the implicit effect measure and self-report measures. Amongst the considerations was the question of what to do with boundary latencies (<300 ms, >3000 ms). The various strategies involved excluding very short latencies, and errors, or transforming them and adding some systematic factor. The basis for using the d score seem defensible, but in general the correlations between the IAT d score and self-report measures appear highly variable. For example, the mean effect size calculated by Hofmann, Gawronski, Gschwendner, Le & Schmitt, 2005 was $r = 0.24$; another estimate is provided by Maison, Greenwald & Bruin, 2001 which varies between behavior (r =0.20; $p < 0.047$), behavioral intention ($r = 0.29$; $p < 0.008$), liking ($r = 0.38$; $p < 0.001$), and subjective beliefs ($r = 0.40$; $p < 0.0005$) for consumer attitudes). The effects of defining

[5] This is not d' used in signal detection theory, but is a measure of effect size, similar to Cohen's d.

different boundaries and testing the effects of imposing boundary penalties are complex, and the authors note that the re-coded scores where error penalties were not added, produced the best internal consistency. The fact that when errors are made, participants are given the opportunity to correct their response is in effect an error penalty, since response latency timing continues until the response is corrected.

While this reflection is inherently interesting, it raises many questions, and the authors concede that '...researchers who use the more sophisticated (and painstaking) methods are rarely rewarded for their extra work - conclusions based on the more effortful methods often diverge little from those based on simpler methods.' (Greenwald, Nosek & Banaji, 2003, p. 198).

It seems that there is some consensus that the '...predictive validity of explicit measures, but not IAT measures...' are '...weakened in socially sensitive outcome domains and for responses that are difficult to consciously control...' (Gawronski & Galdi, in press). Poehlman, Uhlmann, Greenwald & Banaji (2008) state that 'When IAT and explicit measures were strongly correlated, both predicted criterion measures more effectively than when implicit-explicit correspondence was low.' In summary, it seems that IAT measures predict behaviours reasonably well (Maison, Greenwald & Bruin, 2001). The utility of the IAT as a predictive measure seems to be good when sensitive social issues are involved, and Poehlman, Uhlmann, Greenwald, Banaji (2008) claim that this is where it is most dissociated from (or poorly correlated with) explicit measures (also see Greenwald, & Nosek, 2006).

There have been various innovations in the IAT, which are argued to improve its performance, such as the 7 stage IAT which introduces practice blocks for the 2 combined critical trials (3,4 / 6,7) – see Figure 108.

Sequence of Trial Blocks in the Standard Election 2000 (Bush vs. Gore) IAT

Block	No. of trials	Function	Items assigned to left-key response	Items assigned to right-key response
1	20	Practice	George Bush images	Al Gore images
2	20	Practice	Pleasant words	Unpleasant words
3	20	Practice	Pleasant words + Bush items	Unpleasant words + Gore items
4	40	Test	Pleasant words + Bush items	Unpleasant words + Gore items
5	20	Practice	Al Gore images	George Bush images
6	20	Practice	Pleasant words + Gore images	Unpleasant words + Bush images
7	40	Test	Pleasant words + Gore images	Unpleasant words + Bush images

Figure 108. 7 Block IAT from Greenwald, et al., 2003, p. 198.

One does wonder what the 'definitive' IAT would consist of, as since then, a Brief IAT (BIAT) has been designed, and published (Sriram, & Greenwald, 2009). Over the years, a staggering variety of IAT tests has been devised to measure self-esteem (Greenwald & Banaji, 1995), dermatological conditions (Grandfield, Thompson, & Turpin, 2005) and work-related behaviours (Haines & Summer, 2006) amongst other things.

The IAT Species study does nevertheless surface frequently as showing robust and replicable results (Dasgupta & Greenwald, 2001; Dasgupta, McGhee, Greenwald, & Banaji, 2000; Greenwald & Nosek, 2001; Coates & Campbell, 2010). It was for this reason that I chose it to test the technology, prepare to build a series of experiments and draw links between the perceptual framework elaborated earlier, and the implicit cognitive framework of the IAT.

4.2.2 Participants

Participants were recruited from undergraduate classes in Psychology at the University of Cape Town (UCT). The experiment was advertised on the local web page, and participants signed up for a session that lasted approximately 30 minutes. They were given course credit in the form of research participation points (1 point per 30 minute session). The experiment was run in the UCT Psychology computing laboratory. A total of 46 participants did the Species IAT: 32 (70%) were females and 14 (30%) were males. Their mean age was 20.24 (SD=1.65).

4.2.3 Materials

Computer software was designed so that the experiments could be run in different locations – at the UCT laboratory, and at the UKZN (University of KwaZulu-Natal) laboratory. This required a local installation of the software at each workstation at each venue. Provision was made for different database server connections, but the desired default connection string was compiled into the program. The software was written in Delphi, using *Embarcadero RAD Studio* (2010), although a later version was ported to the *Embarcadero RAD Studio XE2* version of Delphi to take advantage of various optimisations, and an improvement in the components of the later VCL (Visual Component Library). The executable file size was reduced from 9.8 MB, to 5.18 MB after being ported and re-compiled in *XE2*.

Both these development systems compile to native Win32 code. The IAT program used a remote database server (*MySQL*® version 5.5.24). Response timing used the clock on the local machine and a dynamic link library ('.dll') encapsulated the timing functions. This was loaded into memory at runtime[6].

An application was developed by D. Mansfield, similar to that for the replication in Chapter 3, for uploading .bmp (bitmap) images, .rtf (rich text format) instruction screens, and run-time parameters (see Figure 109). This application was developed so that the experiment could be configured and administered remotely. Figure 109 shows the stimuli used for the 'species' IAT with the 'in-group' pictures (flowers) in the left panel and the 'out-group' pictures in the right panel.

[6] The accuracy of the timing was checked on 10 different computers, with a variety of operating systems, including Windows XP, and Windows 7. This was done to test different timing methods and set the program thread priority and CPU sharing allocations to optimum values. It was decided to set the thread priority of the IAT program at these levels: 'SetPriorityClass(GetCurrentProcess, HIGH_PRIORITY_CLASS); SetThreadPriority(GetCurrentThread, THREAD_PRIORITY_TIME_CRITICAL);'

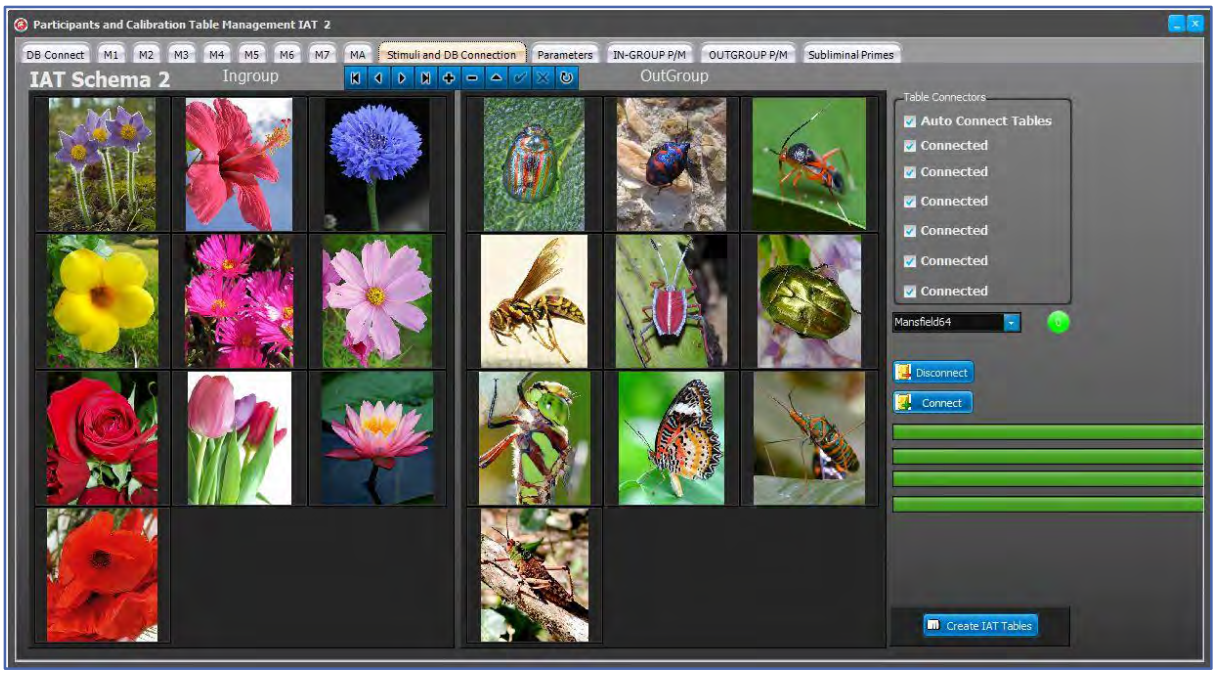


Figure 109. IAT remote database management utility showing stimuli for species experiment.

The other tabs in the application are used to upload instruction screens, questions and other parameters which define the experiment and configure various run-time settings. The stimuli were mined from various sources on the internet and converted to bitmaps, cropped and colour corrected. There were 10 exemplars of each category, and 10 words in each list. The bitmap format was used as there is no processing overhead and delay for decompressing and rendering images as there is with the .jpeg[7] (.jpg) format.

4.2.4 Procedure

Participants were required to sign an informed consent form, which also recorded details of their participation for the course credit system. They were then given a brief verbal explanation and seated at a workstation. The protocol was similar to that used before, in that an anonymous number is used to generate a login and this identified the participant's data. It was also used as a name for the database table which recorded their responses. As soon as the program starts, it downloads all the required files from the database, and stores them to local folders on the computer. This was done, in order to avoid delays, and heavy traffic on the network. Each time the stimuli were presented, they were shuffled into random order and in the case of blocks 3 and 5, the additional lists were also combined in random order.

[7] Joint Photographic Experts Group.

The experiment involved 5 stages:

1. Sorting **pleasant and unpleasant** words: when an unpleasant word was presented (the category names were on the upper part of the screen), the participants were required to press the 'Q' key. When a pleasant word was presented, they were required to press the 'P' key. If they made an error, a large red X was displayed until they corrected their response with the appropriate key-press. The words were gleaned from previous experiments which showed them to be effective in terms of clarity and representative of the category.
2. Sorting **insects / flowers**. This block was randomly swapped (in terms of the key position), with block 4. Participants would either press a Q for insects, and a P for flowers, or vice versa, depending on the order randomisation.
3. Sorting **insects / flowers** and **pleasant / unpleasant**. The pairing was randomly swapped between block 3 and 5. If block 3 required participants to respond to flower pictures and unpleasant words with a 'Q' key-press, and insect pictures and pleasant words with a 'P' key-press, block 5 would reverse this pairing. If block 3 required participants to respond to flower pictures and pleasant words with a 'P' key-press, and insect pictures and unpleasant words with a 'Q' key-press, block 5 would reverse this pairing.
4. This was the same as block 2, but the opposite key-presses were made to sort the pictures of flowers, and insects. If block 2 involved pressing Q for insects and a P for flowers then block 4 would require pressing P for insects and Q for flowers or vice versa.
5. Sorting **insects / flowers** and **pleasant / unpleasant**. If block 3 required participants to respond to flower pictures and unpleasant words with a 'Q' key-press, and insect pictures and pleasant words with a 'P' key-press, block 5 would reverse this pairing: insect pictures and unpleasant words with a 'Q' key-press and flower pictures and pleasant words with a 'P' key-press. This was the same as block 3, except that the sorting was set to be the opposite to that of block 3.

After testing was complete, the data was uploaded to tables on the database, and the results were presented. Feedback was given on the participants' relative association of flowers and insects, to the two categories using the conventional IAT characterisation of score levels (see Nosek, Greenwald, Banaji, 2005 for a discussion of the interpretation of relative levels of the d statistic. This statistic is similar to Cohen's d, and it is calculated in the IAT by subtracting

the block 5 mean from the block 3 mean, divided by the combined standard deviation (Greenwald, Nosek, Banaji, 2003). In the current experiment, block 3 always refers to what is typically the 'easy' sort (for example, flowers and pleasant words). Block 5 always refers to what is typically the difficult sort (for example, insects and unpleasant words). In every case that these blocks are mentioned, these sorts are implied. Although the ordering of the blocks is varied, the name of the block and the task is always the same. Calculation of the d score involves subtracting the block 5 mean from the block 3 mean, divided by the combined standard deviation. This means that the score is usually negative, which is equivalent to a positive score on other IAT experiments. The scores are essentially similar if they are simply negated (-0.4 in the current experiments is equivalent to 0.4 in the standard notation). It is therefore an idiosyncracy of this experiment and the score interpretation is identical. The categorisation of *d* is conventionally 0.2: small effect/mild preference; 0.5: medium effect/moderate preference; 0.8: large effect/strong preference). There were no alterations to the latencies, in terms of adding penalties for very fast responses, but any response ≥ 10 000 ms was assigned the mean latency score for that block.

4.2.5 Results

The overall mean for the d score was -0.54 (SD = 0.37), indicating a moderate preference for flowers over insects. The distribution of d did not deviate significantly from normality (Kolmogorov-Smirnov (46) = 0.095, $p = 0.2$). An ANOVA comparison showed a significant difference between males and females, with males showing more neutral scores than females: $F(1) = 6.15$, $p = 0.032$.

The block means are plotted in Figure 110. Note the similar means and 95% confidence intervals for Block 2 and 4. These blocks involved the same stimuli (shuffled into a different presentation order), but with the response key swapped.

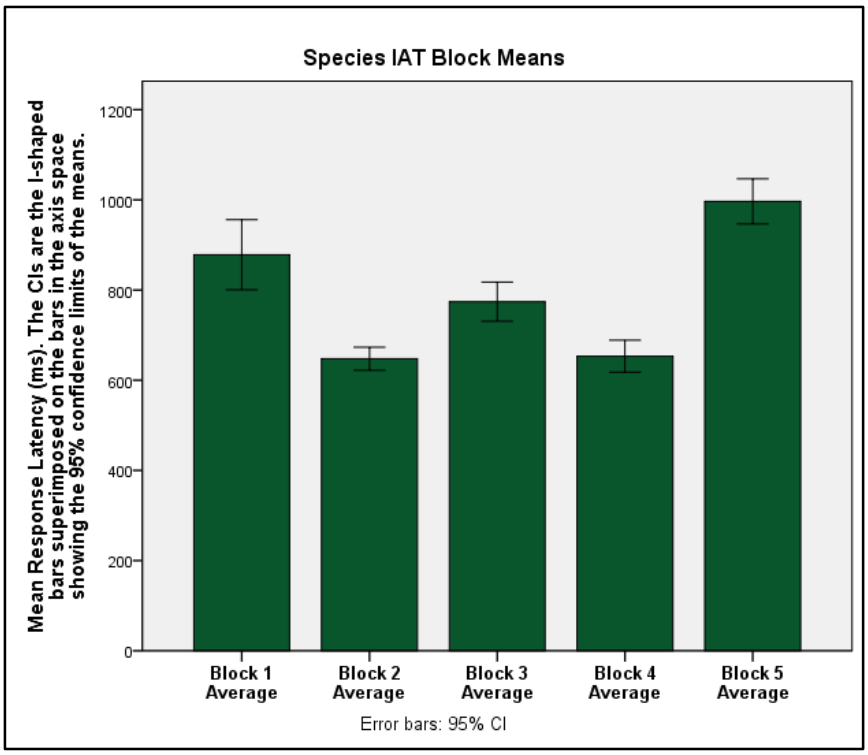


Figure 110. Mean block response latencies for Species experiment. Error bars show the 95% confidence intervals of the mean.

4.2.6 Discussion

The effect is comparable to results reported by Greenwald, McGee, & Schwartz, 1998 who report a mean difference (between the Block 5 and Block 3 mean response latencies) of 166.8 (SD=140.2) vs. a mean difference (996.63 - 774.2 mean latencies for Block 5 and 3 respectively) of 222.43 (SD = 156) in this experiment. However, the d score was more neutral than that reported by the Greenwald *et al.*, (1998) study (absolute d values of 0.54 / 1.19 respectively). Greenwald's study used flower/insect words, whereas this study used images. It is a widely acknowledged, but unexplained finding that effects are stronger with word-based IAT experiments (Greenwald, 2001; Greenwald, 2004). This seems counter-intuitive, and may be incompatible with the finding of Lieberman, *et al.*, (2005) that verbal encoding attenuated amygdala activation in comparison with perceptual encoding. It also seems to disagree with the finding of Tabibnia, Lieberman & Craske (2008) that exposure to images with affective labels tends to attenuate autonomic arousal.

A possible explanation for this might lie in the possibility that the word-based IAT is primarily a semantic associative task, and that the difficulty of associating a disliked word exemplar with a positive descriptor is made more difficult because of increased cognitive

load than affective interference or incongruency. This would make it easier to reconcile the finding of increased effects associated with word-based IAT experiments with the finding of Lieberman, *et al.*, (2005), which does not imply that verbal encoding, although it appears to attenuate amydala arousal, reduces effects of incongruency. Greenwald (2004) admitted that how '...the IAT measures association strengths is not yet well understood.'

The curious argument about the relationship between the IAT and explicit measures is highlighted by Greenwald (2004). In a discussion of the 'Top 10 things not actually wrong with the IAT', one of these is that 'The IAT lacks validity because it is (un)correlated with explicit measures'. The entire exercise of fine-tuning the scoring algorithm, and dealing with boundary latencies was based on how it **improved** the correlation of the IAT result with explicit measures (Greenwald, Nosek & Banaji, 2003). This ambivalence on the relationship between implicit and explicit measures was also criticised by Fazio & Olson (2003). They commented: 'In our view the variability regarding the correspondence between implicit and explicit measures indicates that discussion of whether a relation exists is not very productive. We already know enough to be able to say that the question has no simple answer. That is, the answer is "it depends."' (p. 303).

An additional factor which may influence IAT test results is the choice of items. It is easy to see that presenting the most poisonous, or (from a human perspective), noxious kinds of insects introduces a confound when one of the sorting categories is 'unpleasant', since they would usually be regarded as unpleasant. For example, many people would regard scorpions as unpleasant because they are venomous. Equally, one can present flowers that are less pleasant aesthetically, or in their ecological role (such as carnivorous plants, or alien species). This is an important point that Govan & Williams (2004) make.

For example, Collet (n.d.) describes the *Arum dioscoridis* as one of the 5 most disgusting plants, saying that it has '…the smell of rotting flesh, but it also looks like it has rotting flesh inside it, with the dark mottled look of decomposition.' See Figure 111.

Figure 111. Arum dioscoridis (http://www.environmentalgraffiti.com/news-most-morbid-plants-earth).

In conclusion, it appears that the Species IAT was successfully replicated, and although the effects are smaller than those of Greenwald, *et al.* (1998), they are in the predicted direction, and in basic agreement with the accepted pattern of findings. The replication was important to do because it tested the software, the IAT procedure that was used and collected local data.

4.3 IAT Experiment 2: Race test

4.3.1 Introduction

The 'Race' IAT was done in order to establish baseline IAT data and make comparisons between populations in different regions, namely the Western Cape and KwaZulu-Natal. It was also a prelude to an intervention which follows from the rationale of the earlier research on the effects of perceptual systems in object and word recognition tasks. The classic protocol was followed, as described previously for the Species IAT (see Table 6 for an outline of the procedure). I will provide a detailed review and discussion of the IAT in Section 4.4.1.

4.3.2 Participants

Participants were recruited from the University of Cape Town (UCT) in a similar manner to the previous experiment, and they volunteered in response to a web notice, and received course credit for participation. Another group was recruited from a third year undergraduate

class at the University of KwaZulu-Natal (UKZN). Course credit was given to the UKZN participants for attendance and participation in the experiment, but they were at liberty to decline participation, or withdraw their results after the experiment, if they chose to. There were 108 participants from the UCT site, and 94 participants from the UKZN site (n = 202). There was considerable variation in the gender ratio across sites (UCT: n females = 107, n males =1; UKZN: n females = 74, n males =20). The mean age was 20.1 (SD = 2.29), 22 (SD = 3.45) respectively. There was a significant difference in age between the samples ($F(1)$ = 22.614, $p < 0.001$), but no significant interaction between males/females in terms of age for the combined sample ($F(1)$ = 3.425, p = 0.067).

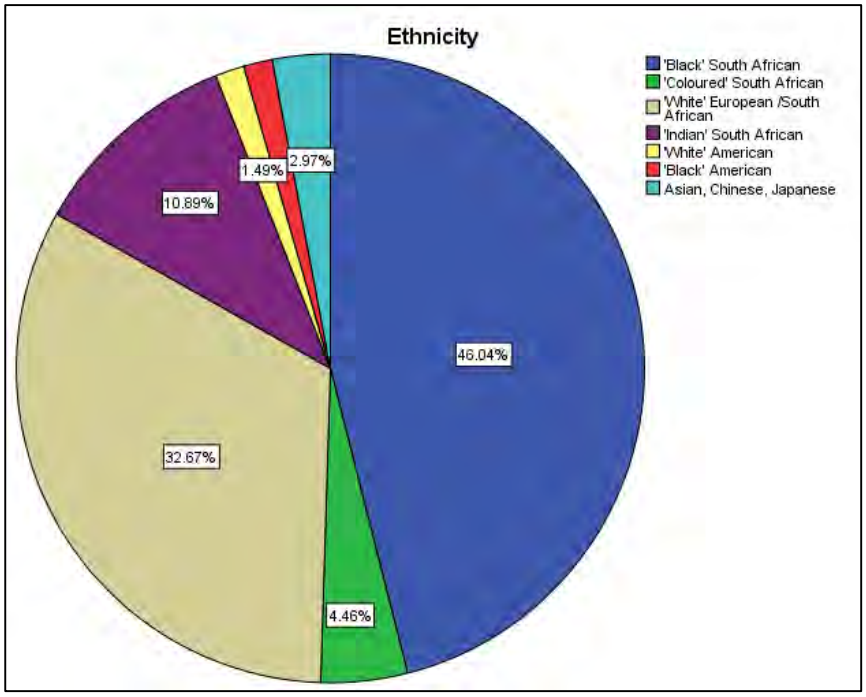


Figure 112. Proportion of ethnic groups in the 'Race' experiment - both sites.

Figure 112 shows the relative proportion of the ethnic groups for the whole sample.

4.3.3 Materials

The same application was used to upload the new stimuli and re-configure the experiment remotely - see 'IAT Experiment 2: Race Test' illustration in Appendix 1. This shows the images used for this IAT experiment. The faces were chosen from a carefully standardised library of pictures from a collection of photographs maintained at UCT by Colin Tredoux, and collected using a photographic protocol that attempted to ensure standardized illumination, colour temperature, exposure aperture and duration. The backgrounds were standardised by using masks that were traced around the faces, and recoloured to a neutral black. Some minor retouching and smoothing was done to eliminate any rough edges or

hairlines, and the pictures of black faces were lightened slightly to enhance their visibility. The 'Race' experiment was configured in exactly the same way in both locations.

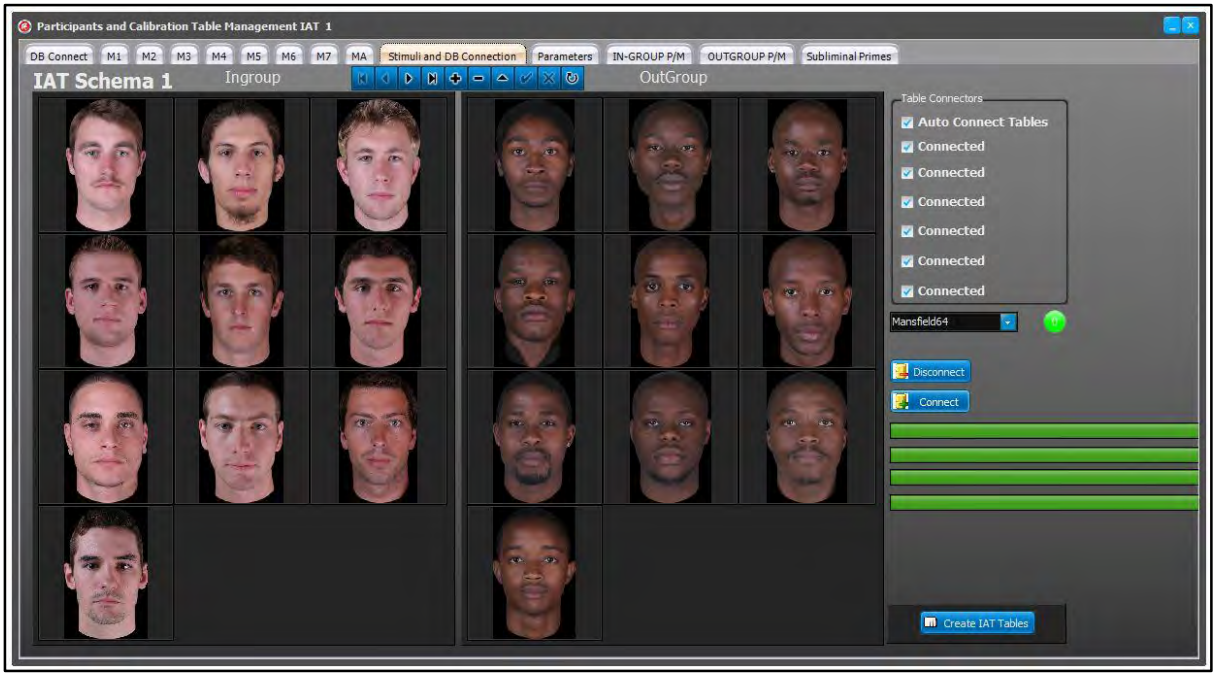


Figure 113. IAT remote database management utility showing stimuli for race experiment.

No changes in the software installed at the UCT laboratory were necessary, as all settings were determined by tables which were uploaded remotely with the database utility shown in Figure 113. The same 'Pleasant' and 'Unpleasant' words were used, and the format of the experiment was identical to the previous one, except for the instruction screens, the stimuli, the feedback phrases to match the performance level in terms of the d statistic (described previously).

The program was installed on 20 workstations at the laboratory at UKZN, and the connection set to a database server on the local area network.

4.3.4 Procedure

The same procedure was followed as described in the Species experiment, except insects and flowers were replaced with 'black' and 'white' photographs, and the concepts were changed accordingly.

The experiment involved 5 stages:

1. Sorting **pleasant and unpleasant** words: when an unpleasant word was presented (the category names were on the upper part of the screen), the participants were

required to press the 'Q' key. When a pleasant word was presented, they were required to press the 'P' key. If they made an error, a large red X was displayed until they corrected their response with the appropriate key-press. The words were gleaned from previous experiments which showed them to be effective in terms of clarity and representative of the category.

2. Sorting **black / white**. This block was randomly swapped (in terms of the key position), with block 4. Participants would either press a Q for black, and a P for white, or vice versa, depending on the order randomisation.
3. Sorting **black / white** and **pleasant / unpleasant**. The pairing was randomly swapped between block 3 and 5. If block 3 required participants to respond to white pictures and unpleasant words with a 'Q' key-press, and black pictures and pleasant words with a 'P' key-press, block 5 would reverse this pairing. If block 3 required participants to respond to white pictures and pleasant words with a 'P' key-press, and black pictures and unpleasant words with a 'Q' key-press, block 5 would reverse this pairing.
4. This was the same as block 2, but the opposite key-presses were made to sort the pictures of white and black. If block 2 involved pressing Q for black and a P for white then block 4 would require pressing P for black and Q for white or vice versa.
5. Sorting **black / white** and **pleasant / unpleasant**. If block 3 required participants to respond to white pictures and unpleasant words with a 'Q' key-press, and black pictures and pleasant words with a 'P' key-press, block 5 would reverse this pairing: black pictures and unpleasant words with a 'Q' key-press and white pictures and pleasant words with a 'P' key-press. This was the same as block 3, except that the sorting was set to be the opposite to that of block 3.

4.3.5 Results

4.3.5.1Analysis of summary data

The summary data for each participant is one row consisting of average reaction time scores for each block, other summary statistics and the d score which is calculated by subtracting the block 5 mean from the block 3 mean, divided by the combined standard deviation. This analysis is done by the IAT program at run time. Each case is a summary of 140 lines of data.

Block 1, 2 and 4 each comprise 20 trials, Blocks 3, 5 each comprise 40 trials (20+20+20+40+40 =140).

The distribution of d in both samples did not deviate from normal. (UKZN: Kolmogorov-Smirnov (94) = 0.071, p = 0.2; UCT: Kolmogorov-Smirnov (108) = 0.035, p = 0.2). The overall means of both regions appear similar (-0.35, SD = 0.39; -0.39, SD = 0.39) for UKZN, and UCT regions respectively. An ANOVA test shows that the d score does not vary between the two sites (F(1) = 2.57, p = 0.613). See Figure 114.

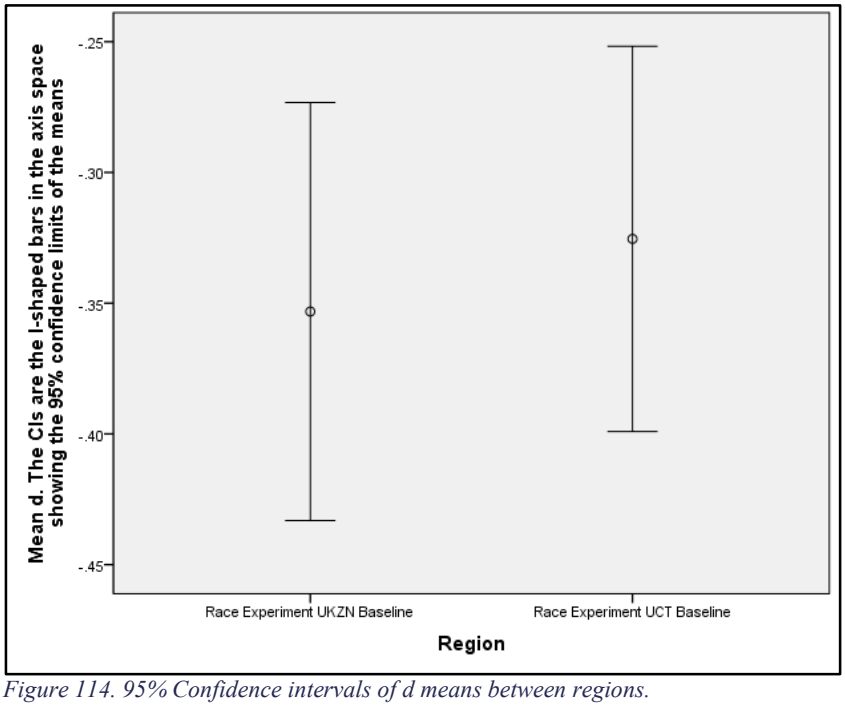


Figure 114. 95% Confidence intervals of d means between regions.

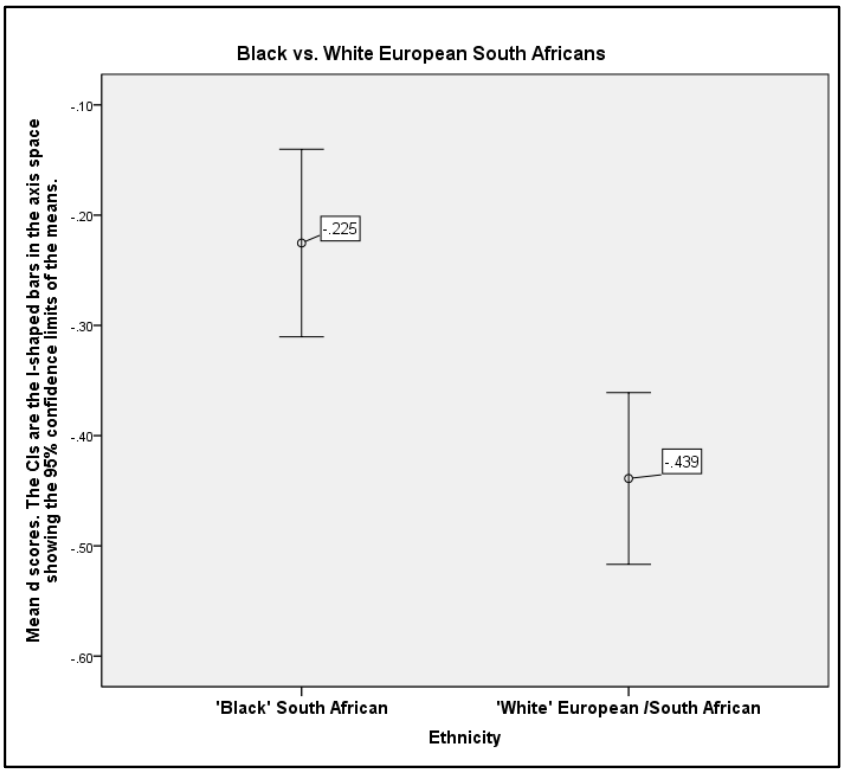


Figure 115. 95% CI of d means between 'Black' / 'White' groups.

It was interesting to compare 'Black' and 'White' South African groups as it seemed likely that the social and psychological aftermath of apartheid would be reflected in this implicit measure. Indeed, there is a clear difference between these groups (Figure 115) and an analysis of variance is significant $F(1) = 12.43$, $p = 0.001$. The Levene test for homogeneity of variances is non-significant, $(1, 157) = 2.799$, $p = 0.096$.

Although it was interesting to compare these two primary South African groups which were probably the most polarised by apartheid, discarding 'Coloured' and 'Indian' South Africans, Asian, Chinese, 'White' Americans and 'Black' Americans, would result in *n* being reduced to 159. It was difficult to predict *a priori* how these other groups would behave. The apartheid racial classification considered 'Coloured' 'Indian' and Chinese as 'non-White' and probably presumably 'Black' Americans would have been considered 'Black'. However, they might share more in common with 'Coloured', 'Indian' and other groups in terms of culture and language. The apartheid racial classification system which divided South African society so deeply was decidedly arbitrary and frankly, although this experiment was designed to test attitudes to 'Whites' vs. 'Blacks', it was not known how other ethnic groups besides so-called 'Blacks' and 'Whites' would behave. How these ethnic groups fit into the outmoded notion of 'Black' and 'White' was not clear, but it seemed most unsatisfactory to exclude 43 (21%) of

the sample on the dubious assumption that they are racially *sui generis,* or dissimilar to any other group.

A decision had to be made about how to group participants so that justice could be done to the full data set. The 'Black' South African and 'White' European / South African groups have non-overlapping confidence intervals but the 'White', 'Indian' South African and 'Asian, Chinese, Japanese' group means appear similar. The 'White American' and 'Coloured South African' groups seem similar to each other, but their confidence intervals overlap with all of the other groups. Although it may seem logical to group 'Black Americans' with 'Black' South Africans, the confidence intervals overlap with all of the other ethnic groups – see Figure 116. It must be pointed out that *n* is low in several of these groups, so this analysis is not definitive.

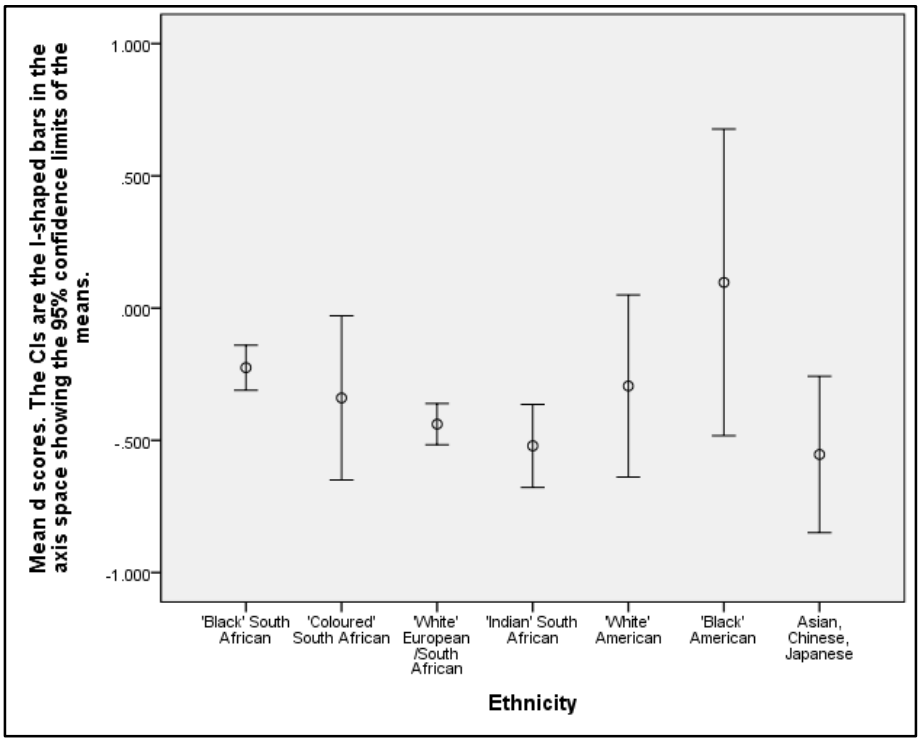


Figure 116. 95% confidence intervals of mean d scores for different ethnic groups.

The most parsimonious solution, given the complexity of the data, was to combine the minority groups into a 'super set' ethnic grouping and leave the 'Black' group unchanged. This resulted in two groups 'Black' South African (n = 93) and 'Mixed' (n = 109),

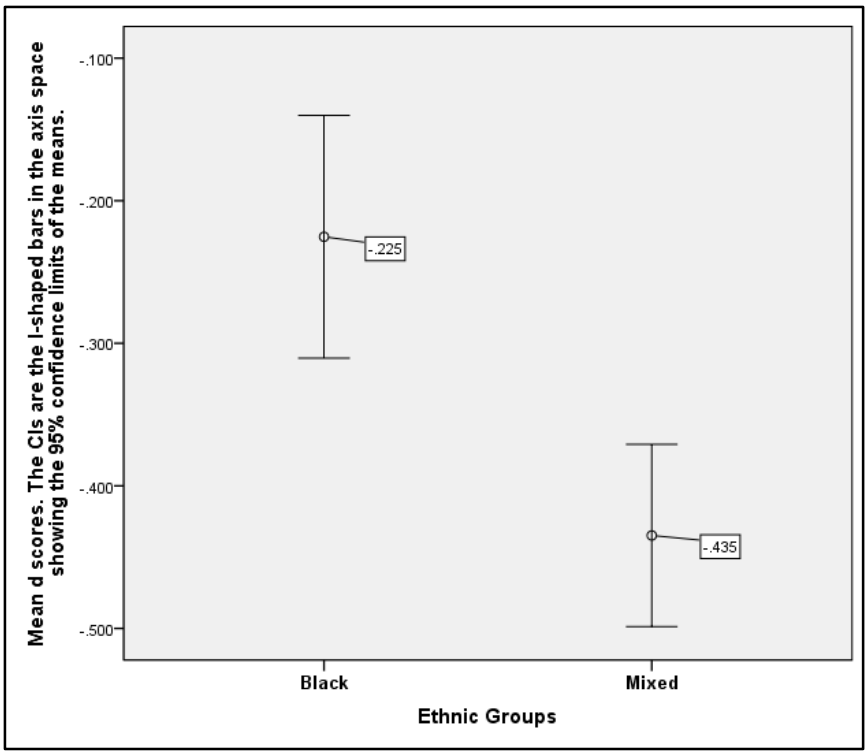


Figure 117. 95% CI of d means for 'Black' vs. 'Mixed' groups.

The result of this new grouping seems satisfactory (see Figure 117) as the 'Black' South African group is unchanged and the mean is therefore the same (-0.225, SD=0.413) and the means for the 'White' South African group and the new combined 'Mixed' group are almost identical (-0.439, SD = 0.317 vs. -0.435, SD = 0.336 respectively). A Oneway analysis of variance with Ethic Groups ('Black' and 'Mixed') as the IVs and d scores as the dependent variable is significant $F(1) = 15.76$, $p < 0.001$. The Levene test for homogeneity of variances is non-significant, $(1, 200) = 2.6$, $p = 0.108$. A graph of the 95% CI of the means for the 'Black', 'White' and combined minorities groups ('Coloured', Asian, Chinese, Japanese, 'Indian South African', 'White American' and 'Black American') is in Appendix 5 (Figure 145). This plot illustrates the fact that the 'Minorities' and 'White' groups are almost indistinguishable.

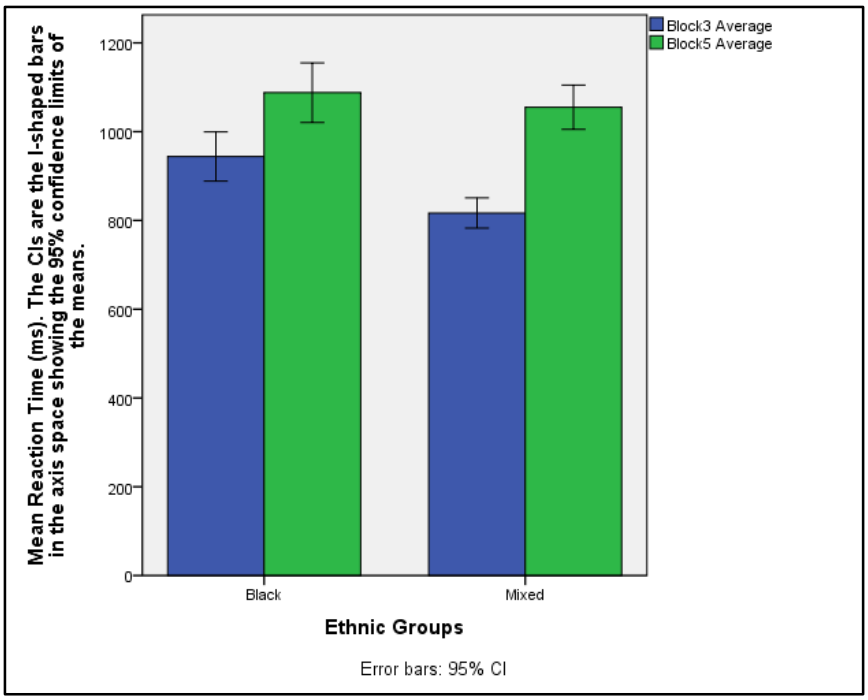


Figure 118. Block 3/5 mean latencies by ethnic group in 'Race IAT' error bars (95% CI of mean).

The final picture of the response latencies is informative (see Figure 118). The combined (Black and Pleasant / White and Unpleasant) trial (block 5) mean latency is similar between groups, but the other combined trial (White and Pleasant / Black and Unpleasant) (Block 3) mean latency is relatively lower in the 'Mixed' group. Shorter block latencies suggest that participants found these items easier to sort together. In effect, it seems that both groups found block 5 similarly difficult to sort, but the 'Mixed' group found block 3 relatively easier to sort. This difference in response latency for the different blocks is the basis of the d measure which constitutes the summary score of this implicit task.

4.3.5.2 Reaction time analysis

While it is conventional to analyse the summary data only, consideration was given to the fact that the summary data is a massive reduction of the total amount of data generated by an IAT experiment. Each case in the summary data set is derived from 140 trials and there is a considerable loss of information. For example, reaction times for all stimulus types are simply averaged together (pleasant/unpleasant words, pictures of Black and White faces). The raw data set records all stimulus items for all trials. Since each IAT case consists of 140 trials, the raw data set consists of a total of 28280 trials (140 trials * 202 participants). In order to do a more fine grained analysis, each individual record table was extracted from the database with a *SQL* query.

The distribution of the reaction time data in the raw data set has a positive skew. This is also evident in the summary data for the block averages (see Figure 119). Interestingly, the distribution of d, which is calculated from these averages, was normal (see Figure 120). Various transformations were done to improve the distribution: log (base 10), log (natural), square, square root and reciprocal (see Figures 121, 122). The best distribution was seen with a reciprocal transformation.

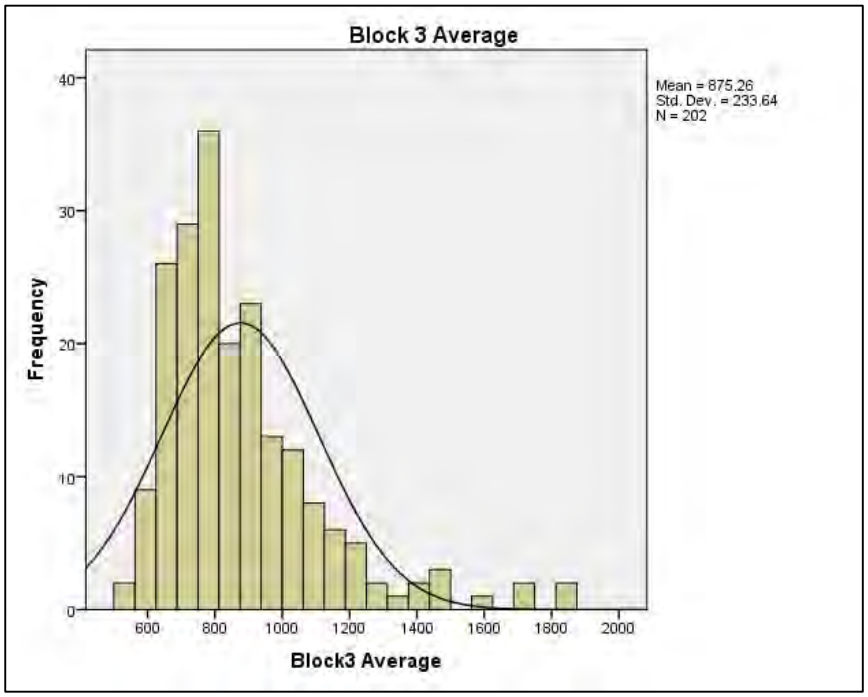


Figure 119. Frequency plot of Block 3 reaction time means from summary data set.

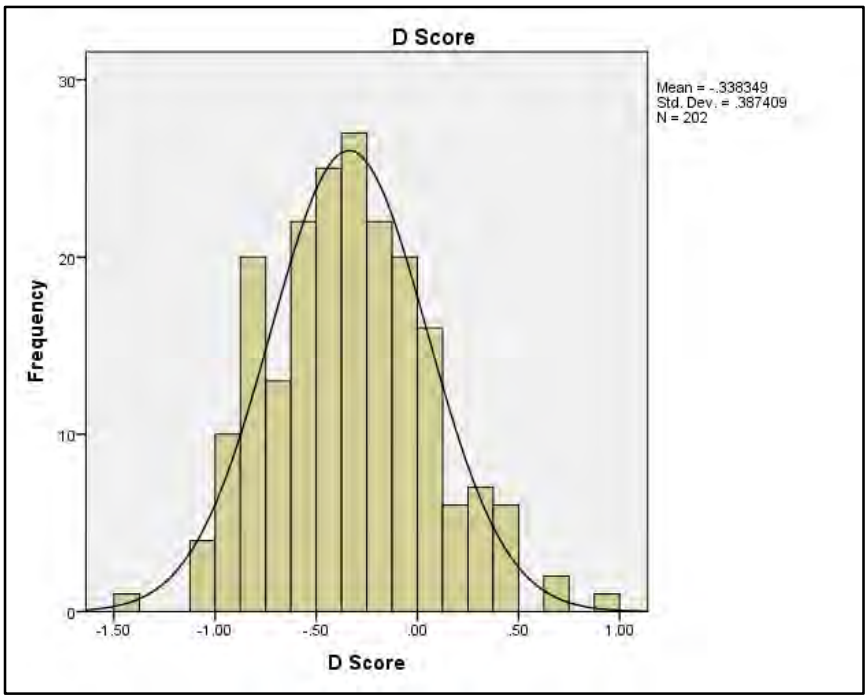


Figure 120. Frequency plot of d score from summary data set.

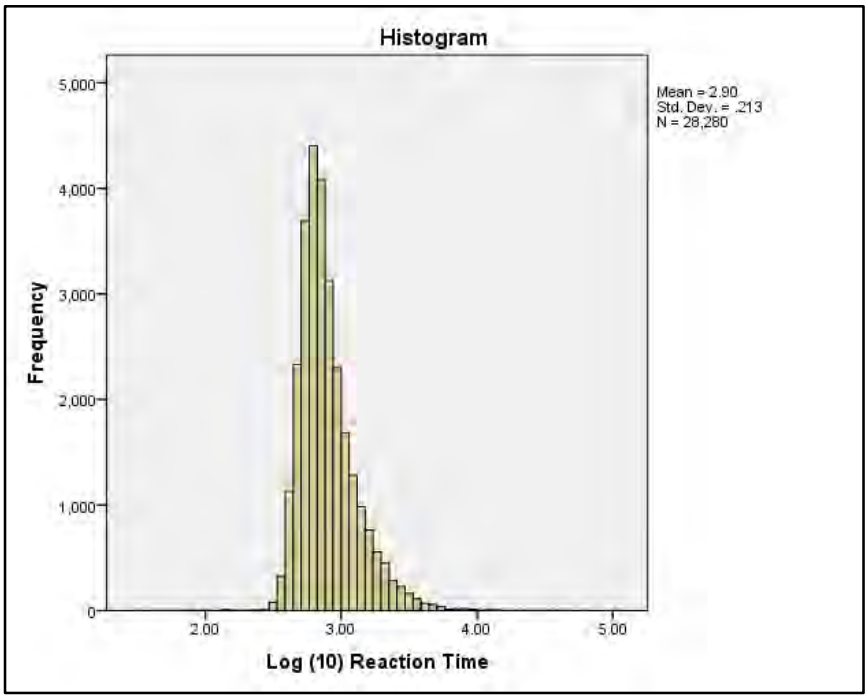


Figure 121. Frequency plot of log transformed data (reaction time).

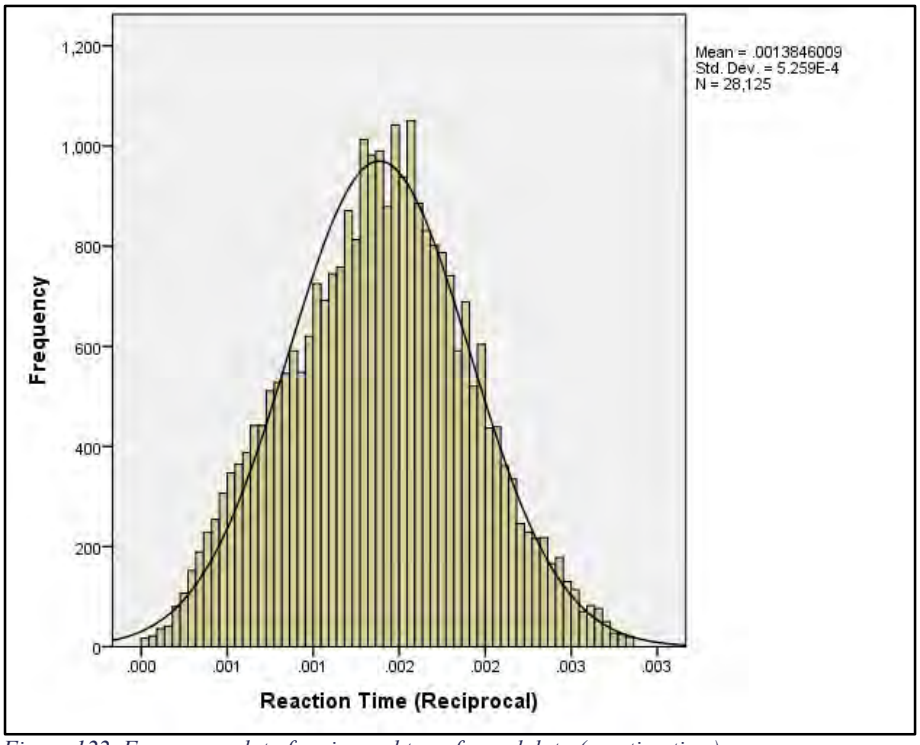


Figure 122. Frequency plot of reciprocal transformed data (reaction time).

A total of 155 extreme values were removed to further improve the distribution since these values were in the tail area. The decision to eliminate these values was guided by a stem and leaf plot which showed 155 extreme values (≥ 0.00284). Since this is a reciprocal variable, this value is equivalent to 352 ms. Eliminating cases with latencies < 352 ms meant discarding about 0.5% of the total number of cases. However, short latencies (<300 ms) are considered to be invalid (see Greenwald, Nosek & Banaji, 2003) and there have been various approaches to dealing with this problem. At one stage it was common practice to eliminate trials with latencies of < 300 ms, or to recode them to a defined boundary value. Greenwald, *et al*. (2003) suggest eliminating '…subjects for whom more than 10% of trials have latencies less than 300 ms' (p. 214). In short, it is most likely that these short reaction times are invalid responses and eliminating them improved the distribution further. The final distribution parameters are: kurtosis = -0.41, skewness = -0.002, $\bar{x}$ = 0.00138, median = 0.0014, mode = 0.0015, SD = 0.0005.

To analyse patterns of variance in the raw data set a factorial (Univariate) ANOVA was designed with *Reaction Time* as the dependent variable and *Block* (3, 5), *Gender*, *Item Type* and *Ethnic Grouping* ('Major Groups') as the independent variables. Because blocks 3 and 5 are the critical trial blocks from which the d score is computed, they were included in the analysis while blocks 1, 2 and 4 were excluded. The variable *Item Type* shows whether word

or picture stimuli were presented. The *Ethnic Grouping* variable is the same variable that was constructed for the previous analysis and involves 'Black' and 'Mixed' groups.

The overall model is significant: $F(15) = 91.89$, $\eta\rho^2 = 0.079$, $p < 0.001$.
The following 3-way interactions were significant:
Gender * *Item Type* * *Ethnic Grouping* ($F(1) = 5.97$, $\eta\rho^2 < 0.0005$, $p = 0.015$)
Block * *Item Type* * *Ethnic Grouping* ($F(1) = 3.91$, $\eta\rho^2 = 0.0005$, $p = 0.048$)
Block * *Gender* * *Ethnic Grouping* ($F(1) = 4.59$, $\eta\rho^2 = 0.0005$, $p = 0.032$)

The following 2-way interactions were significant:
Item Type * *Ethnic Grouping* ($F(1) = 6.27$, $\eta\rho^2 = 0.0005$, $p = 0.012$)
Gender * *Ethnic Grouping* ($F(1) = 65.03$, $\eta\rho^2 = 0.004$, $p < 0.001$)

There were significant main effects for *Block*, *Gender*, *Item Type* and *Ethnic Grouping*.
Block: $F(1) = 154.88$, $\eta\rho^2 = 0.01$, $p < 0.0005$;
Gender: $F(1) = 56.6$, $\eta\rho^2 = 0.004$, $p < 0.0005$;
Item Type: $F(1) = 229.8$, $\eta\rho^2 = 0.014$, $p < 0.0005$;
Ethnic Grouping: $F(1) = 193$, $\eta\rho^2 = 0.012$, $p < 0.0005$;

I will discuss the 3-way interactions first, as they show a detailed picture. Since the dependent variable was a reciprocal, the untransformed variable (*Reaction Time*) will be plotted where possible.

*Gender * Item Type * Ethnic Grouping*
Figures 123 shows the reaction time variation associated with word vs. picture presentations for males and females in the different ethnic groups. The plot shows the reaction time unit in milliseconds, although the analysis was done with the reciprocal variable.

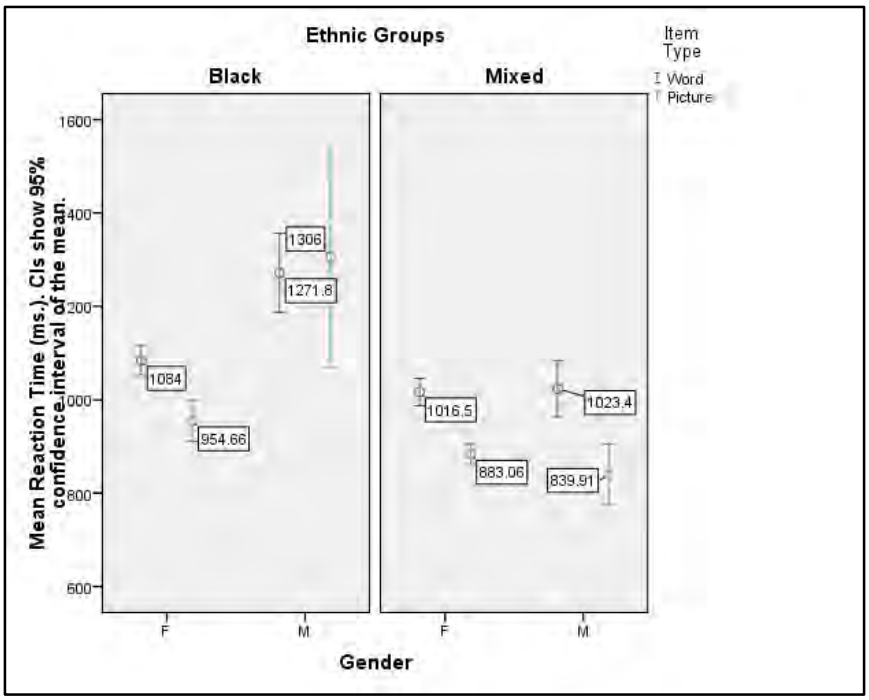


*Figure 123. Gender * Item Type * Ethnic Grouping interaction. The error bars show the 95% confidence interval of the mean for each group. Reaction time (Y-axis) is in milliseconds.*

The difference in response patterns between males and females to words vs. pictures between the two groups constitutes this interaction. Females' reaction times were longer for words than for pictures in both groups, but where 'Black' males' reaction times were similar for words vs. pictures, males in the 'Mixed' group had shorter reaction times for pictures than words.

*Block * Item Type * Ethnic Grouping*

This interaction is shown in Figure 124. Reaction time is shown in milliseconds.

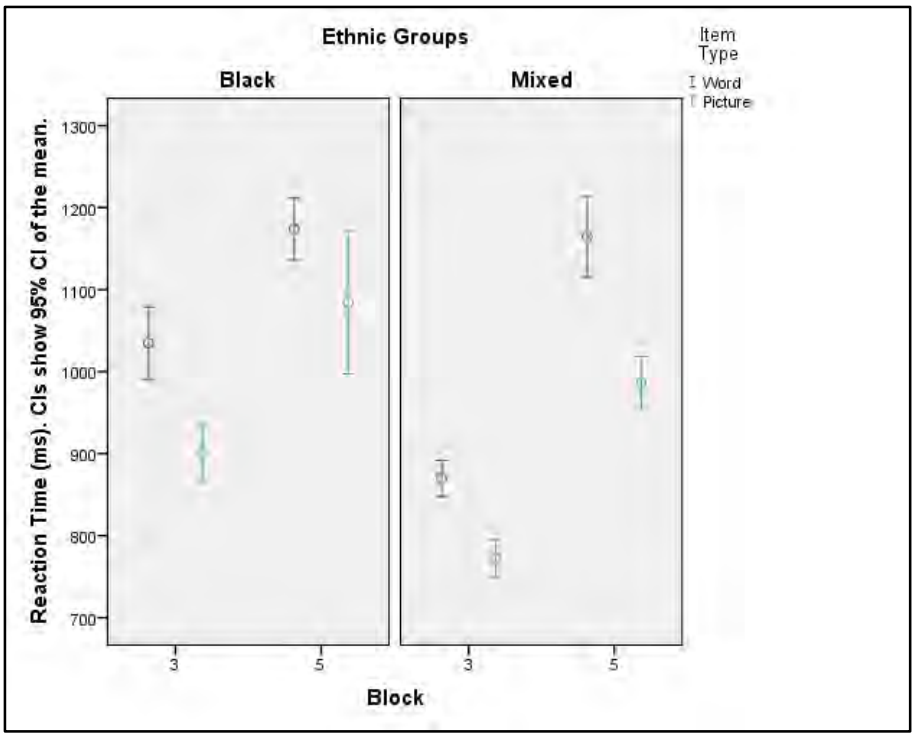


*Figure 124. Block * Item Type * Ethnic Group interaction. The error bars show the 95% confidence interval of the mean for each group. Reaction time (Y-axis) is in milliseconds. This plot shows the variation in reaction time for the different item types (words vs. pictures) associated with the ethnic groupings and trial blocks. The 'Black' group does not show the same difference in reaction times in block 5 as the other group and reaction times are faster for the mixed group in block 3.*

This interaction shows the variation in reaction time associated with word vs. picture presentations in trial blocks (3 or 5) for the different ethnic groupings. Word stimuli are associated with longer reaction times in both blocks for the 'Mixed' group and the reaction times for block 5 are longer for both stimulus types than block 3. This tendency is also seen in the 'Black' group but in Block 5, the difference in mean reaction times between stimulus types is much less.

*Block * Gender * Ethnic Grouping*

Plots of this interaction are shown in Figure 125.

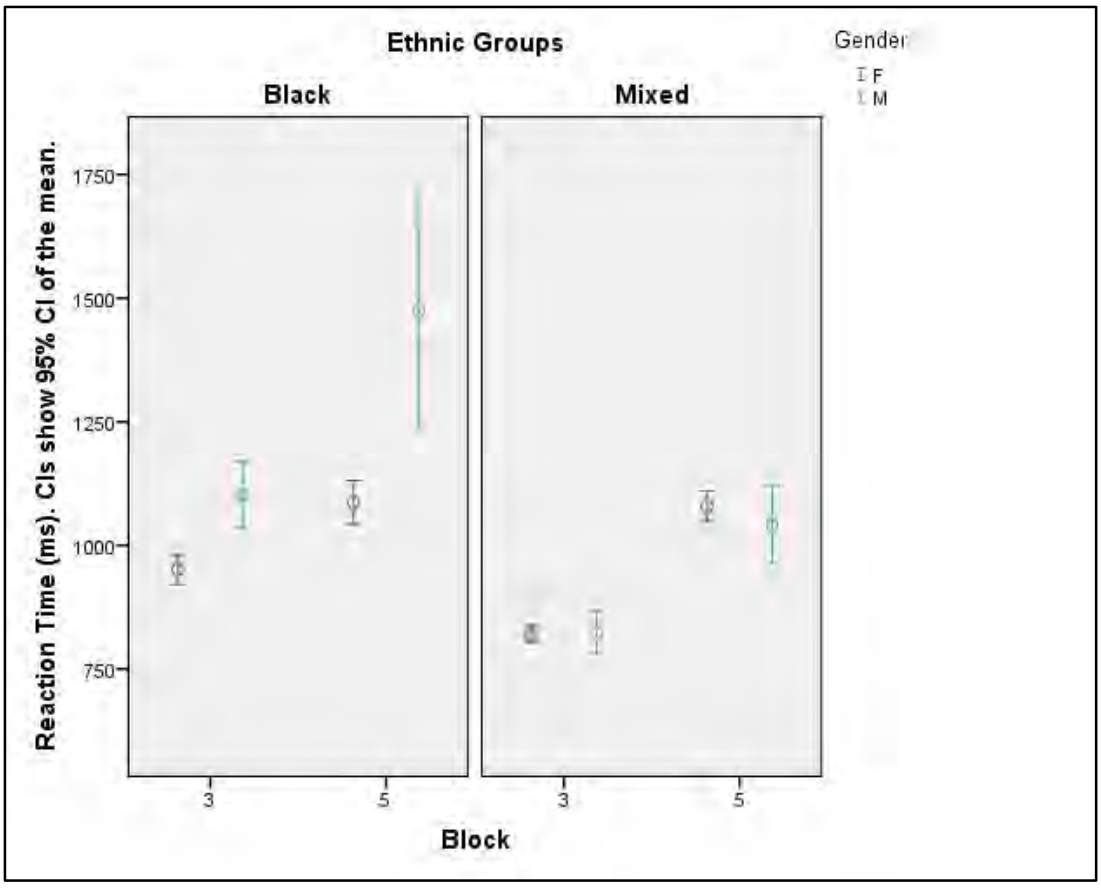


*Figure 125. Block * Gender * Ethnic Grouping interaction. 'Black' males show longer reaction times than 'Black' females in block 3. This difference is not evident in the 'Mixed' group.*

This interaction shows that reaction time was associated with gender differences between the two ethnic groupings in the different trial blocks. In the 'Mixed' group, males and females performed similarly between the different blocks, but in the 'Black' group males' reaction times were longer in both blocks, but especially in block 5 where the difference is approximately 389 ms.

4.3.6 Discussion

4.3.6.1 Summary data set analysis

The IAT mean scores are consistent between the two populations (UCT and UKZN) - see Figure 138. A comparison of 'Black' and 'White' South African participants shows the expected difference in preference, with 'Blacks' showing more neutral scores, and 'Whites' more negative scores. This can be interpreted as 'Blacks' showing low group preference, and 'Whites' showing higher group preference (i.e. a relative preference for 'White'). The re-grouping of participants into 'Black' and 'Mixed' groups meant that the entire data set could be used and there was surprisingly little difference in the mean scores of the 'White' South

African and ‘Mixed’ groups. It seems that the Asian, ‘Black American’, ‘White American’, ‘Indian’ and ‘Coloured’ groups’ implicit scores are very similar and the major difference is between ‘Black’ South Africans and this ‘mixed’ group with quite a diverse ethnic and cultural composition. Perhaps the relative homogeneity within this ‘Black’ group, contrasted with the relative homogeneity within the ‘Mixed’ group is a function of political dynamics and different patterns of group identity. The ethnic groups in the ‘Mixed’ group appear to show greater identification with ‘Whites’ than ‘Blacks’.

The general tendency is for ‘Black’ participants to show relatively neutral mean scores ($\bar{x}$ = -0.23, SD = 0.40) and for the ‘Mixed’ group consisting of various cultural and ethnic minorities to show more negative scores, which suggests a relative preference for ‘Whites’ compared with ‘Blacks’ ($\bar{x}$ = -0.43, SD = 0.34). The overall mean (d = -0.34, SD = 0.39) is lower than that reported by Nosek, *et al*., 2005 (d = 0.52, SD = 0.51). The mean in this study is reported as negative, so to compare this with the results of Nosek, *et al*., it should be negated (0.34). The research of Nosek, *et al.* has consistently shown preference for 'Whites' amongst both 'Whites' and 'Blacks'. The lower magnitude of the effect may be due to differences in the test items, the slight differences in scoring procedure and different population characteristics. It is also possible that students in South African tertiary education settings may be influenced by ongoing processes of transformation.

The fact that there is a robust difference between the ‘Black’ and ‘White’ South African grouping is certainly consistent with expectations and generally fits the findings of other researchers. It was difficult to predict how other ethnic and cultural minorities would respond, especially since the task was designed to measure implicit attitudes to ‘Black’ vs. ‘White’. The general conclusion to this question is that they tended to show a relative implicit preference for ‘Whites’.

4.3.6.2 Reaction time analysis

The results of the factorial ANOVA will now be considered and discussed with attention to the interactions which were seen.

*Gender * Item Type * Ethnic Grouping*

The interpretation of this interaction is that 'Black' males' response pattern was different to the pattern seen with the other groupings. 'Black' and 'Mixed' females' response pattern was similar in that their reaction times to words was longer than for pictures. However, 'Black' and 'Mixed' males' reaction time pattern was different. In the 'Mixed' group, males' reaction times were faster for both categories, but relatively faster to pictures and 'Black' males' reaction times were similar between stimulus types. In general, 'Black' males' reaction times were generally longer.

This finding is difficult to interpret. Perhaps 'Black' males had a more leisurely approach to the task than 'Black' females, but this is mere conjecture. This sub-group did not fit the pattern seen with 'Black' females and both genders in the 'Mixed' group and their reaction times were longer for both stimulus types. The interpretation may be clearer when the *Block * Gender * Ethnic Grouping* interaction is discussed.

*Block * Item Type * Ethnic Grouping*

This interaction shows that reaction times between the item type (words/pictures) varied in the different blocks as a function of the race grouping. The reaction time difference between the different stimulus types in the 'Mixed' group in block 5 was greater than that for the 'Black' group. In short, stimulus type was not associated with reaction time difference in the 'Black' group in block 5, whereas it was in the 'Mixed' group. Reaction times for words between two groups were similar in block 5, but the mean reaction time for pictures was faster in the 'Mixed' group. In this block, pleasant words are sorted with black faces and unpleasant words are sorted with white faces, so it suggests that sorting pictures in this condition was easier for the 'Mixed' group than it was for the 'Black' group. However, the mean reaction times for sorting words is similar between the different groups for this block. It is interesting that 'Blacks' seemed to have more difficulty sorting black faces with pleasant words and white faces with unpleasant words than they did with the opposite sort in block 3 (black faces and unpleasant words, white faces and pleasant words). The overall pattern suggests that performance in block 5 was similar between 'Black' and 'Mixed' groups (word means are similar, confidence intervals overlap for picture means), but different in block 3. The fact that reaction times were much shorter for both item types in block 3 for the 'Mixed' group but similar in block 5, suggests that their performance in block 5 was not as much inhibited by the nature of the sorting task (since it was similar to that of the 'Black' group), as

it was facilitated by the nature of the sorting task in block 3. For the 'Black' group, this facilitation effect was less in block 3 than it was for the 'Mixed' group.

The traditional explanation of the IAT does not stipulate whether the effect is derived from the sorting task in block 5 being more difficult for 'Whites' than 'Blacks', or the sorting task in block 3 being more easy for 'Whites' than 'Blacks'. This experiment suggests that both groups have similar difficulty with block 5, but the 'Mixed' group finds the block 3 task easier than did the 'Black' group.

The interpretation of this interaction probably has to do with ease of sorting in block 5 where the task involves sorting Good or Black Faces together and bad or White faces together. The 'Black' group mean for sorting words is similar to the 'Mixed' group, but sorting pictures to the various evaluative conditions seems to have been a difficult task for this group.

*Block * Gender * Ethnic Grouping*

This interaction showed a different pattern of reaction times associated with gender and race groups between the different blocks. 'Black' participants showed a gender difference in both blocks, whereas this was not evident in the 'Mixed' group. 'Black' males' reaction times were generally longer than 'Black' females' times. This was especially evident in block 5 where 'Black' females' reaction times were similar to those of both genders in the 'Mixed' group, but 'Black' males' mean reaction time was more than 300 ms longer than the other groups.

The inference may be made that 'Black' males found the sorting task more difficult than any other sub-group, especially block 5. Interpreting this finding is more difficult. The fact that the task involved sorting male faces may be a partial explanation, but this does not explain why 'Black' males' response pattern was so different to that of males in the other major group.

4.3.7 General conclusion

The main results of this experiment are consistent with predictions and this experiment replicates the results of other research. The Race IAT experiment was validated and the software, stimuli and procedure appeared to be satisfactory.

Analysis of the reaction times revealed some complex dynamics that were not anticipated. These concerned the interaction between the different trial blocks, gender, item type and participant groups. In general, females' response patterns across the different racial groups tended to be more similar to each other and males' response patterns were less similar. 'Black' males' reaction times stood out as being relatively slow. The detailed analysis of reaction times suggests that the 'IAT effect' may be based on differential facilitation in one of the trial blocks involving positive 'White' in-group sorting and negative 'Black' out-group sorting. There is evidence that facilitation is stronger for the 'Mixed' race group than the 'Black' race group in this block. Performance between ethnic groupings is roughly similar in the other critical trial block involving the converse sort. It may be that the IAT effect is based on facilitation of the traditional in-group-good, out-group-bad sorting task (white positive / black negative) for the 'White/Mixed' group rather than differential task difficulty associated with doing the traditional outgroup-good/ingroup-bad (black positive / white negative).

4.4 IAT Experiment 3: IAT Race test with M/P-biasing intervention

4.4.1 Introduction

Having established that both the Species, and Race IAT were yielding consistent and predictable results and that the IAT program created to run remotely was working reliably, this experiment built on the previous series of experiments in Chapter 3 which used P-biased, and M-biased presentations of stimuli (text and graphical) and applied this methodology to a Race IAT, in order to see if the implicit response 'bias' which the IAT is supposed to measure, and which was evident, as predicted, in the previous experiment, could be systematically manipulated.

The IAT was designed as measure of implicit bias or association and the hope was that it might prove to have ecological validity, perhaps from the way that it places participants under pressure to make quick, spontaneous decisions (see Poehlman, Uhlmann, Greenwald, & Banaji, 2009). This may tend to inhibit the effects of consciously held beliefs about social attitudes and evoke a more automatic style of responding, perhaps similar to how one behaves in everyday life (Bargh, Chen, & Burrows, 1996 in Amodio, & Lieberman, 2006). One might argue that everyday realities which appear to be structured by racial prejudice may simply be guided by simple social heuristics, coarse stereotypes or some kind of perceived probability, rather than actual racial hatred or overt prejudice. For instance, Anwar, & Fang,

(2006) determined that 'troopers' (law enforcement officers) '…tend to search a higher proportion of minority motorists than white motorists' (p. 127). The question they pose is whether this reflects racism, or 'statistical discrimination'. In other words, is this a pragmatic strategy which serves to maximise the number of 'successful' searches (for contraband), or is it motivated by stereotypes about minority groups? Perhaps this behaviour may be driven by more covert racial attitudes, sometimes characterised as 'symbolic racism'. This concept is attributed to Kinder, (1971) in Durrheim (2003).

Durrheim & Dixon's (2003) study of informal segregation on a South African beach is interesting because the behaviour in question was spontaneous and is probably a visible example of an entire class of social behaviours which are a function of a racist legacy. They found evidence of sharp segregation patterns which increased as 'whites' progressively became outnumbered by 'blacks'. The reasons given by some 'whites' for maintaining distance from 'blacks' was not framed as opposition to racial desegregation, but as a question of 'personal space' and cultural norms (p. 11). Others expressed frank prejudice. The interview data seems to exemplify both 'Old Fashioned' racism and a more subtle way of expressing racial preference. The latter possibly raises questions of whether attributions about 'personal space', value judgements or cultural norms made in the context of the encounter were motivated by social desirability (Durrheim & Dixon, 2004). The probability exists that whatever attributions were expressed when respondents were asked for their comments, the self-segregation was related to the apartheid legacy of racial separation in almost all spheres of life and that the behaviours were at least partly motivated by prejudice, antipathy and negative evaluations of 'blacks'.

It is believed that responses to questions about racial prejudice are influenced by social context and are not well correlated with actual behaviour (a claim which Banaji, Nosek & Greenwald, 2004 echo). The IAT was constructed with this in mind, and Banaji, *et* al., argue that it is a better predictor of actual behaviour than more traditional self-report measures. They argue that the reliance on self-report measures was based more on their availability and convenience than the premise that attitudes '…operate only as conscious entities.' (p. 280).

Banaji, *et al.* argue that implicit, or at least indirect measures, measure 'more automatic forms of attitudes' (p. 280) and even go as far as to claim that they may be dissociated. They also state, perhaps paradoxically, that correlation coefficients between implicit and explicit

attitudes range between $r = 0.50$ and $r = 0.86$. Does the finding that implicit and explicit measures are associated somewhat variably mean that they are measuring a single underlying construct with varying efficiency? This could imply that implicit measures are not just better predictors of discriminatory behaviour (*ibid.*), but that they produce better, or perhaps more valid estimates of what is commonly referred to as an 'attitude'.

Perhaps the very notion of an 'attitude' has, on the one hand insufficient scope and on the other hand precision, to make it an adequate construct in terms of explaining and predicting human behaviours. Durrheim & Dixon (2004) contended that

> '…because researchers do not have a direct and unambiguous measure of race prejudice that is, a measure that is both independent of observed attitude expressions and unequivocally indicative of racism this endeavour is haunted by problems of operational and definitional circularity…' (p. 627).

On the other hand, there may simply be a degree of inherent randomness or indeterminacy in how 'attitudes' influence and predict behaviour. It is not clear that prejudice is a stable property of an individual that will be expressed consistently across various social contexts.

Earlier I argued that non-human social behaviours include elements of strategic self-presentation and deception which amount to tactical deception. Tactical deception is not seen pejoratively, but more as a means by which individuals deal with complex social interactions, especially when there is fierce competition (see Byrne, & Corp, 2004). Social contexts may evoke deceptive behaviours that are not deliberate or conscious but which facilitate adaptive functioning and may be an alternative to physical conflict. The suggestion that the relative size of the neocortex (Byrne & Corp, 2004) is a predictor of deception rates in primates underscores the idea that the primary function of a big brain is not so much physical 'tool' usage, but social tool usage where individuals are manipulated towards some objective (see Byrne & Whiten, 1991).[8]

In the human realm, self-presentation is much more nuanced and sophisticated. Vohs, Baumeister & Cirarocco (2005) state that 'Self-presentation consists of behaviors designed to make a desired impression on others.' (p. 633). It is a necessary social skill and one which

[8] However, the association between various measures of encephalisation and cognitive ability in primates is still a matter of debate (see Deaner, Isler, Burkart & van Schaik, 2007).

predicts success in life (*ibid.*). Although it is more benign than frank deception and manipulation, it is done as a means of self-advancement. To this extent, there may be variation in the expression of prejudice between different social contexts. As Durrheim & Dixon write, 'Variation is thus explained in terms of tactical expression and repression: individuals will keep their genuinely prejudiced attitudes from public awareness if such attitudes contravene self-presentational goals…' (p. 629). There have been major ideological shifts in the last 50 years or so that have resulted in the scrapping of race-based legislation, but the 'racial inferiority' beliefs have given way to more 'subtle' variants which it has been claimed, are still pervasive (see Dasgupta, McGhee, Greenwald & Banaji, 2000) and in post-apartheid South Africa 'Old Fashioned Racism' has become progressively more taboo and socially unacceptable. It is not difficult to see that aligning or positioning oneself as a social egalitarian in post-apartheid South Africa is wiser than revealing racist ideas and behaviours. Masking prejudice as a function of social adaptation is pragmatic, whether or not it is done consciously or with insincerity.

I have suggested that suppressing expressions of prejudice is a reasonable strategy in terms of social impression management. Whether the inhibition of prejudice is necessarily a product of conscious intention is difficult to answer. The question of why there is so much variation in correlations between implicit and explicit measures has been debated and Hofmann, Gawronski, Gschwendner, Le & Schmitt (2005) offered 5 possible explanations which are paraphrased below:

1. Implicit measures are unbiased by motivational influences, but explicit self-reports may be influenced by social desirability concerns.
2. Assessment of implicit representations may vary in the degree to which they are susceptible, or available for introspection. Participants may thus vary in the degree to which they are aware of their own implicit attitudes.
3. Explicit and implicit measures access different and independent internal representations. The amount of cognitive effort required to access these different representations may vary. Representations that are activated automatically (upon exposure to a particular stimulus) may be associated with less cognitive effort and more spontaneity, whereas accessing another kind of representation takes more effort.
4. Methodological variation, including trial randomisation and counterbalancing the order of various blocks could influence error variance and observed correlations.

5. Explicit and implicit measures may assess completely independent constructs which may correlate variably.

The general conclusion these authors reached did not support any of the above propositions, except for 4. They found evidence that the practice of counterbalancing the order of critical blocks was associated with increased levels of correlation with explicit measures. Their principal conclusions were that there was no support for the notion that '…explicitly and implicitly assessed measures are completely dissociated…' (p. 1380) and that '…variations in correlations can be explained by the degree of spontaneity of explicit self-reports' (*ibid*).

Another study which supports this position was done by Nier (2005). Nier's experiment involved different conditions under which the IAT was administered: in the 'accurate' condition, participants were informed that the IAT was an accurate way of measuring attitudes and in the alternate condition, they were not given any information. In the so-called 'accurate' condition, the relationship between the implicit and explicit measures was significant, whereas in the alternate condition, it was non-significant. Nier attributes the strengthening of the association in the 'accurate' condition to greater motivation to report explicit attitudes accurately. The general conclusion of this research is that implicit and explicit attitudes are probably not as dissociated as previously thought and the correlation between them is moderated by the level of motivation to accurately disclose racial attitudes.

Another study which attempted to address the question of how participants' knowledge affects their performance in the race IAT was done by Frantz, Cuddy, Burnett, Ray & Hart (2004). Their research rationale was informed by stereotype threat theory and they predicted that the ('white') participants who were aware of what the IAT aimed to measure would produce more extreme scores. The two hypotheses that are of most interest to this discussion were expressed as follows:

a. 'White people who are threatened by the possibility of appearing racist will have elevated effects compared to individuals who are not threatened.'
b. '…those who are motivated to control prejudiced responses are more susceptible to this stereotype effect…'

Their findings from a series of 3 experiments were that both hypotheses were supported and they concluded that when there is a threat of appearing racist, 'white' participants produced

more extreme IAT scores. Their concluding comment is interesting: 'These results indicate that people who care very much about not appearing racist will likely have increased IAT effects when they think the study examines racial attitudes' (p. 1621). One of the possible implications of this finding is that it is difficult to *control* responses to a test which is believed to measure racism, especially when people wish to appear non-racist. Whether or not this is associated with poor self-knowledge (i.e. not realising the extent to which they are racist) is not clear.

This study seems to support Nier's (2005) finding, but the nuances of the study done by Frantz, *et al.* suggest that the improvement in correlation between implicit and explicit measures when participants feel that their attitudes are under scrutiny, may be at least partly a function of more extreme IAT scores.

These studies either argue against, or do not support the idea that implicit and explicit attitudes are dissociated. The major point which Frantz, *et al.* raise is related to the question of controlling responses in implicit measures of racism. When there is an increase in focus on the measurement of racism and especially when participants believe the implicit measure is accurate, there seems to be an increase in both implicit and explicit scores and a closer correspondence between the measures. In the IAT measure, there seems to be an increase in scores whereby motivation to be seen as non-racist exerts a paradoxical effect. When participants' motivation to under-estimate their self-appraisals of racism is reduced, scores on explicit measures tend to increase.

The contention that some attitudes, knowledge and other information may be inaccessible to introspection is critical to the Washington researchers' (Greenwald, *et al.*) understanding of implicitness. They argued that some attitudes may indeed be unavailable to introspection (e.g. Banaji, Nosek & Greenwald, 2004; Greenwald & Banaji, 1995). Their reasoning about the implicit/explicit dichotomy was drawn from memory research which showed, for example, that previous exposure to primes increased the probability that a word stem task would be completed utilising these prime words, even though participants showed poor ability to recall or even recognise these prime words. This influence of previous 'introspectively unidentified' past experiences on current responses was a major part of the basic template for implicit attitudes. A major critique which Greenwald & Banaji (1995) levelled against the traditional conception of attitudes was the implication that they were held consciously. They argued that

'…the observed high level of reliance on direct measures of attitudes indicates a widespread (even if not widely stated) assumption that attitudes operate primarily in a conscious mode.' (p. 7). Citing the 'Mere exposure effect' (Zajonc, 1968 in *ibid.*) in a report on a meta-analysis of the 'Mere exposure effect' research, they point to Bornstein's conclusion that stimulus unawareness tends to enhance positive feelings which occur with exposure (see Bornstein & D'Agostino, 1992).

The implication of the Washington researchers' position is that implicit attitudes are to some extent 'unconscious'. The psychoanalytic conception of the 'Unconscious' is not what these researchers wish to connote and they seldom refer to the term 'unconscious' except in their earlier publications. However, many commentators do discuss the IAT as an instrument for measuring 'unconscious' bias or racism (Blanton & Jaccard, 2008) as this is a common perception about the IAT. Ironically, the Washington researchers' stance (regarding implicit and explicit attitudes as being dissociated and 'unavailable to conscious introspection') seems logically equivalent to a claim that implicit attitudes are *unconscious*. This is very different to saying that what the IAT measures is attitudes or behavioural dispositions that people are unwilling to report. As Fazio & Olson (2003) point out, 'Participants may be unaware that their attitudes are being assessed, but that does not mean that they are unaware that they possess those attitudes' (p. 300). Their point is that the term implicit does not necessarily mean 'unaware'.

Fazio & Olson suggest a shift in the focus of research to understanding the role of awareness of automatically activated racial attitudes, so that the question of correction can be addressed. This suggests that one of the major issues is dealing with how attitudes are activated.

Is racism an unconscious (inaccessible), but a potent factor in determining everyday behaviours? Or is it potentially 'unavailable to introspection' except when it becomes a conscious focus for reflection in a social or research situation? Are these positions incompatible? Much of the research evidence points to the role of motivation in accurately reporting attitudes. Indeed, it seems that conscious focus is required for many people to reflect on their attitudes and the prospect of finding evidence for the dissociation, or inaccessibility of attitudes seems poor (Blanton & Jaccard, 2008). Blanton & Jaccard contend that '…people may sometimes lack knowledge of and control and consequences of their racial biases' (p. 277) but conclude that the evidence for 'unconscious racism' is poor. This

does not mean that the phenomenon of racial prejudice and discrimination does not exist. To a significant extent, the question of how prejudice is expressed, is a function of how people choose to exhibit, or inhibit particular tendencies. Inhibition of racist behaviours requires social judgement and conscious effort. In some social situations, including the proverbial 'braai' or barbeque, racist talk may be heard, probably because people feel they can relax, drop their guard and express themselves candidly. Alcohol undoubtedly plays a role in undermining inhibition (see Cyders & Smith, 2008). Verwey & Quayle (2012) based their research on social discourses in post-Apartheid South Africa on a methodology that accessed the private domain of the 'braai'. They write, 'The social context of a private braai attended exclusively by white Afrikaners at a friend's house produced a racially and culturally homogeneous private space, and elicited the type of in-group talk that – as Annelie Botes pointed out – is rarely spoken in public or mixed settings.' (p. 551).

In short, it does not seem likely that implicit measures express some underlying attitude-like construct that is 'unconscious' in any elaborate sense. However, this also does not mean that attitudes are necessarily accessed in everyday life to guide behaviours and decisions that are made. Durrheim & Dixon's (2003) study seems to illustrate several things:

a. Spontaneous behaviours may occur in response to a situation when 'whites' and 'blacks' shared a space (in this case, a beach) which was previously an exclusively 'white' domain. The behaviours were 'racial clustering' (races tended to occupy different areas of the beach and did not mix) and when the number of 'black' beach-goers increased, avoidance and withdrawal by 'whites'.
b. Attitudes about race were accessed very easily when people were interviewed, and there did not seem to be any evidence of dissociation that made them inaccessible. The types of attributions varied from simple, frank racist beliefs to more nuanced ideas about space and culture.
c. If a substantial number of 'blacks' had not appeared and the beach had remained the domain of 'whites' as it had in the past, racial beliefs, attitudes and behaviours would probably have remained latent and in a certain sense, 'unconscious' because they were simply not activated.

If the preceding discussion has established that people, for the most part have conscious control over how and when they express prejudice and that control is exerted in the service of self-presentational goals, the question that follows has to do with *how* control is exerted and

what factors potentially undermine conscious control. When conscious control is diminished, due to internal or situational factors, or the interaction of both, it follows that more spontaneous behaviours will occur. This seems to be one of the implications of the stereotype threat research, where task-related anxiety appears to be associated with increased IAT effects.

Conscious, cognitive control may also be related to cognitive resources and this is one of the implications of the stereotype threat research. Empirical evidence is available to show that self-regulation is compromised under conditions of fatigue and depletion (Baumeister, *et al.,* 1998 (in Hoffman, *et al.,* 2009); Vohs, *et al.,* 2005), cognitive load (Boon, Stroebe, Shut, & Ijntema, 2002; Ward & Mann, 2000 (in Hoffman, *et al.,* 2009)) and low self-monitoring (Collins, 1978 (in Hoffman, *et al.,* 2009). Hoffman, *et al.* remark that 'If control resources are low, experimental evidence has shown that automatic object evaluations and corresponding behavioral schemas of approach or avoidance are indeed coactivated' (p. 165).

In the final IAT 'race' experiment, some procedural innovation was introduced in order to investigate hypotheses about the role of two visual pathways on implicit cognition. The definition of the term 'implicit' tends to be problematic, so a definition that is fairly general in psychology was sought. Gawronski & De Houwer, (in press), suggest that 'A central characteristic of implicit measures is that they aim to capture psychological attributes (e.g., attitudes, stereotypes, self-esteem) without requiring participants to report a subjective assessment of these attributes.').

This definition is compatible with the reasoning from implicit memory. Dew & Cabeza (2011) remark that 'A common distinguishing feature between explicit and implicit memory centers on the control of retrieval. Although tests of explicit memory instruct subjects to think back to the studied information, tests of implicit memory make no mention of the study episode.' (p. 177).

The conclusion of the preceding discussion is that the evidence for the existence of 'unconscious' prejudice, where people are simply unaware of, or cannot access their own attitudes, is poor. There is more evidence available to support the idea that people are aware of their attitudes, but that they tend to exert control over how they are expressed (Blanton & Jaccard, 2008; Fazio & Olson (2003); Frantz, Cuddy, Burnett, Ray & Hart (2004); Nier

(2005); Hofmann, Gawronski, Gschwendner, Le & Schmitt (2005)). Alternatively, they may exert control when their attitudes are brought to awareness, for example, when they are interviewed and even then, they may '…confabulate …reasons for their actions and preferences' (see Wilson & Bar-Anan, 2008). The term 'unconscious' raises so many problems that it seems wiser to use constructs that are used and understood more broadly. Frequently, the term 'conscious' is used as a predicate of what De Houwer & Moors (2007) describe as 'goal-related terms' like 'intentional' or 'controlled'. However, it does not add much value to describe a process as being 'consciously intentional', rather than 'simply intentional' because the terms 'intentional' and 'conscious' are more or less synonymous.

The purpose of this research was to investigate the role of the magnocellular system in implicit cognition and this brings the problem of defining 'implicit' into sharp focus. The debate about what 'implicit' means (whether this is understood as a social phenomenon, such as unstated prejudice, or the product of an experimental procedure or measure) is difficult to resolve. De Houwer & Moors (2007) contend that an implicit attitude is the product of some process of measurement and the term should only be used in relation to that measurement procedure. Their second major contention is that if some measure is to be regarded as implicit, it is necessary to specify why, or in what sense, that measure is implicit. They suggest that '…one can say that racial IAT scores provide an implicit measure of racial attitudes in the sense that participants have little control over these scores…' (p. 180). Furthermore, they argue that it is necessary to justify the claim that the measurement is implicit by showing how it is the product of relevant measurement processes. Perhaps this is equivalent to saying that one needs to explain why the linear expansion of mercury in a tube is relevant to measuring the construct of 'temperature'. Their suggested definition of an implicit measure is: 'An implicit measure is a measurement outcome that reflects the to-be-measured construct by virtue of processes that have certain features' (*ibid*.).

One feature which De Houwer & Moors suggest is important in supporting a claim of implicitness, is demonstrating that participants '…do not intentionally produce a certain score…'. This implies that control (or rather, the lack of control over the underlying processes) is one of the key aspects of an implicit measure. At one point, De Houwer even suggested that 'implicit' and 'automatic' are synonymous concepts (De Houwer, 2006 – in De Houwer & Moors, 2007).

While there is simply no consensus about what constitutes 'implicitness', one of the themes which has consistently surfaced in the literature in the last two decades has to do with control vs. automaticity. Wittenbrink, Judd & Park (1997) noted that

> …it is argued that stereotypic knowledge may affect social judgment and behavior differently, depending on whether the response is based on a controlled, conscious consideration of available information or whether stereotypic knowledge is activated spontaneously, outside of the perceiver's conscious control. Specifically, discrepancies between controlled responses to a given social group and implicit influences of stereotypic knowledge may arise if stereotypic associations that come to mind spontaneously differ from one's explicit attitudes. (p. 262).

These authors cite Devine's (1989) work on the way that negative stereotypes are automatically activated in 'White Americans'. Although they criticised the methodology used in this study (the implied hostility of the subliminal primes), their study replicated Devine's work and showed stereotype congruent effects in a semantic priming task. This kind of experiment was a precursor of other implicit research techniques and although about 16 years of research has accumulated data which mostly confirms their conclusions, the basic idea which distinguishes between 'intentional control and conscious thought processes…' and '…spontaneous, effortless activation of knowledge contents that is driven by cues in the stimulus environment rather than by an active memory search' (p. 262) persists. They remark that 'More recently, a rather different explanation has been suggested, according to which a person may, at the same time, hold positive attitudes toward a social group and nevertheless be influenced by negative group stereotypes' (*ibid*). This referred to Devine's (1989) work, which ironically, was done about 24 years ago.

There has been considerable reflection on the question of automatic vs. controlled processing and some attempts to define, or 'decompose' it (Bargh, 1994). Bargh cautioned against the idea that automatic and controlled processes could be distinguished dichotomously and pointed out that behaviours that could become automatic (such as driving a car), could also be subjected to control (one can choose to stop driving). Applications of dual process theories have continued to burgeon, to the extent that Gawronski & Creighton (2013) remark that 'it is difficult to imagine what contemporary social psychology would look like without the theoretical guidance of dual process theories' (p. 282).

One of the points which Bargh's (1994) article reflects is that when strict criteria are applied to automaticity, especially the 'all-or-none assumption', the result is confusion and misunderstanding (p. 2) because it does not seem possible that these criteria can be met by any experimental contrivance. There now seems to be general agreement that the all-or-none view is not viable (De Houwer & Moors, 2007; Gawronski, Balas & Creighton, 2013; Gawronski, & Creighton, 2013) Bargh's 1994 article reviewed a number of different kinds of automaticity: preconscious, postconscious, goal-dependent and this raised questions about awareness, controllability and efficiency, which Bargh believes are fundamentally important principles for humans in that they have major legal and moral implications (p. 7).

Bargh's remarks are interesting:

> 'Automated social cognitive processes categorize, evaluate, and impute the meanings of behavior and other social information, and this input is then ready for use by conscious and controlled judgment and decision processes, yet those judgments and decisions are not uncontrollable or predetermined by that automatic input…' (p. 29).

Bargh conceded that some products of perception are automatic and yet this does not mean that '…they are impossible to control or adjust for when one is aware of them, if one desires' (p. 30). Bargh argues further, 'The considerable body of research on motivational control over stereotypes and other judgemental biases has shown, for the most part, the use of automatically supplied input in consciously produced judgemental output is not mandatory…' (*ibid.*). Bargh concludes that automatic processes are neither an 'unqualified blessing' nor an 'unqualified curse'. Although they can be the basis on which '…people stereotype others and often misunderstand the reasons for their own feelings and behaviour' (*ibid.*), the 'automatization of routine thought processes frees one's limited attentional resources for nonroutine matters, and enables a reduction of the massive amount of stimulation and information bombarding one at any given moment into a more manageable subset of important objects, events and appraisals…' (p. 31). The challenge, according to Bargh, is not letting this efficiency of thought lead to a lack of awareness or 'a loosening of one's grip over decisions and judgements' (*ibid*.).

From the point of view of this research, the most interesting form of automaticity mentioned by Bargh is 'Preconscious Automaticity'. This is one form of automaticity which seems to be associated with perceptual processing and it seems that all that is required for this to occur, is that a given stimulus be registered as part of the perceptual field. Bargh makes one

qualification in this description: 'A preconsciously automatic process requires only that the person *notice* the presence of the triggering stimulus in the environment' (p. 4 – emphasis added). The qualification is repeated: 'These processes occur automatically when a stimulus is noticed, as part of the act of figural synthesis…' '…and do not require a deliberate goal or intention' (*ibid*.). How can one *notice* something that is not yet perceived? Bargh states that one may be aware of the end result of 'this fast preconscious construction of the percept' and is seems clear that the actual process of perception ('the microgenesis of its perception') occurs 'in the absence of conscious or deliberative response to the stimulus…' (*ibid*).

What is meant by the idea that all that is required for preconscious automaticity is that 'the person 'notice' the presence of the triggering stimulus in the environment'? Logically, it does not seem possible for something to be 'noticed' before it is perceptually processed. Bargh's meaning seems to be that some object merely has to be registered perceptually and this would be the sense in which the term 'noticed' is used. This would be an unusual application of the term 'noticed' as it normally implies some degree of awareness (for example, one might say 'I noticed that there was snow on the mountains' and this implies that one's attention was drawn to the snow and one was consciously aware of it). The simple presentation of information does not necessarily lead to conscious perception - as the 'Attentional Blink' research shows (Shapiro, *et al,* 1997). One possibility is that preconscious (visual) processing occurs perpetually (with a high degree of redundancy), but when some object of interest appears in the perceptual field (the familiar face of a friend while one is waiting in a queue, for example), attention is recruited by bottom-up processes (Itti, Koch & Niebur, 1998; Lin & He, 2009), or perhaps the interaction of bottom-up and top-down processes (Bar, Kassam, Ghuman, Boshyan, Schmid, Dale & Hämäläinen, 2006).

Whatever the exact interpretation might be, Bargh draws attention to what may be termed 'preconscious' processing of perceptual information and the implication is that at some stage, it occurs involuntarily and without conscious effort. It seems clear that at least some early visual processing occurs in a manner that is compatible with this concept of 'Preconscious Automaticity'. It is simply not plausible that all aspects of visual processing (for example, the signal processing in the retina, at least to the level of the ganglion cells and probably beyond) require conscious effort. Some early cortical processing appears to be influenced by top-down signals, including face specific processing (see Desjardins & Segalowitz, 2013). Baars (1994) pointed out that when '… a stimulus is degraded so that automatic pre-perceptual processing

is blocked subjects often begin to perform conscious hypothesis-testing.' Baars used the example of upside-down reading as a 'conscious analogue of a process that normally takes places quickly, automatically, and unconsciously'. This seems equivalent to disabling some cognitive mechanism that makes reading almost effortless and automatic, so that letters and words have to be effortfully and consciously decoded before meaning can be hypothesised or inferred. It is thus easy to appreciate that reading is a skill that has to be acquired and practiced, before it becomes automatic.

The IAT experiment 3 is based on the idea that a particular visual sub-system (the magnocellular or M system) mediates some kinds of perceptual and cognitive automaticity. The properties that make this system a likely candidate, probably include its conduction speed, sensitivity and spatial summation properties. Bar, 2003; Bar, 2004; Bar, *et al.*, 2006; Kveraga, *et al.*, 2007b; Kveraga *et al*., 2007 have suggested that the M system underlies some of the fast processing which facilitates object recognition and their research has shown results that are compatible with this idea (neuroimaging data and reaction time / accuracy data). This visual recognition system may be related to the reading system which Baars (1998) suggested makes reading take place 'quickly, automatically, and unconsciously'.

Disabling the M system may result in a loss of cognitive speed, accuracy and automaticity in visual object recognition that is somewhat analogous to Baars' illustration of the task of trying to read when the page is turned upside-down. Although the magnitude of the effect would be different, the idea is similar in that the hypothesis testing which takes place automatically to speed up object recognition when the M system contribution is present, would have to be done consciously and deliberatively when it is disabled. In other words, when the automaticity of this system is inhibited, there may be a disruption of the spontaneous, evaluative processes which potentiate and facilitate rapid responding, and the opportunity may be created for a more conscious evaluation and response. This may be related to the finding by Amodio & Lieberman (2006) that verbal encoding of race images was associated with the inhibition of amygdala activation (and what they believe to be 'race-related emotion') compared with perceptual encoding. Their experiment may illustrate the idea that conscious (verbal) processing may exert moderating or inhibitory effects compared with responses based on perceptual processing when higher cognitive (verbal) representational processes are not initiated.

Bargh has argued that ‘The automatization of routine thought processes frees one's limited attentional resources for nonroutine matters…’ and this is effectively an optimisation which helps us manage massive amounts of information with less cognitive effort. The disadvantage of this optimisation, according to Bargh, is that ‘…with the increased efficiency of thought also comes a lack of awareness of engaging in that process, leading to a likelihood of misattributing the causes of one's feelings and a loosening of one's intentional grip over decisions and judgments.’ (Bargh, 1994, p. 31). This might be one of the ways that we fast-track decisions that are based on gender, race, ethnicity and other visual features that are part of our social reality.

Kveraga, *et al*., (2007b) have provided evidence from both behavioural (reaction time and accuracy data) and neuro-imaging (fMRI) that supports the idea of a fast, top-down visual recognition system involving the magnocellular pathway. This system seems to facilitate and probably adds a significant degree of automaticity to the process of object recognition. The rationale for the experiment which follows, is based on the idea that the same magnocellularly-based system also facilitates processing of race-based facial features.

While object recognition and evaluation is potentially more neutral than recognition and evaluation of social objects (like faces), it seems possible that if a significant visual/perceptual and cognitive system is involved in facilitation of visual object recognition, the same system might be involved, at least at an early stage, in face processing.

There is a large body of research which supports the idea that facial expressions (especially negative expressions) are processed rapidly by the brain. There are suggestions of a ‘quick and dirty’ pathway to the amygdala (Vuilleumier, Armony & Dolan, 2003) and Jetha, Zheng, Schmidt & Segalowitz (2012) suggest that this may be transmitted by the magnocellular pathway and might involve a sub-cortical colliculo-pulvinar-amygdala circuit. Adolphs (2008) suggested that very brief (39 ms) masked exposures to facial threat may evoke neural responses and bias information processing. While there is debate about whether amygdala activation is preconscious or follows cortical processing (*ibid*.), one of the hypotheses frequently mentioned in the literature and which Adolphs refers to as the ‘strong hypothesis’ states that ‘…certain aspects of visual information (contrast, visual motion, and low spatial frequency) are conveyed…’ by a ‘subcortical route’ (p. 168). Some of the recent debate in the literature focuses on the role of the amygdala and evidence for the existence of a direct,

subcortical pathway to the amygdala (see Pessoa & Adolphs, 2010). This debate is important, but is not of central interest to the current discussion.

The question which this research attempts to explore, concerns the visual processing of race-related facial information and the preceding discussion sets out a rationale which states that the magnocellular system has an important role in the projection of low spatial frequency information which is utilised to activate race-related knowledge and facilitate relevant cognitive processing and responses. This hypotheses is similar to the object recognition hypothesis proposed by Bar, 2003; Bar, 2004; Bar, *et al.*, 2006; Kveraga, *et al.*, 2007b; Kveraga *et al*., 2007. The proposal is made that this visual sub-system is involved in the automatic perceptual and cognitive processing which activates stereotypical knowledge about race and informs relatively automatic responses. When the contribution of this system is present, the prediction is that more spontaneous, stereotype-congruent responses will be evident in the scores of an implicit race measure like the IAT. The second prediction is that when the contribution of the magnocellular system is not present, responses to racial features will not be processed with such a high degree of automaticity and participants' responses will be slower and more effortful. This will offset the influence of stereotypical knowledge and increase the controllability of participants' responses. This reasoning is based on a suggestion by Bargh (1994) that when automatic processing inputs are disrupted, there is an increased probability that the response will be informed by knowledge other than mere stereotypes. As Bargh writes:

> If one processes information about the target person more effortfully, even if there are stereotypic or categorical inputs into one's judgements (as when an influence exists that the perceiver is not aware of and therefore does not engage in an adjustment process…), those judgements will at least be moderated by the additional individuating information collected and will not be determined solely by the stereotypical input' (p. 29).

This experiment aimed to produce two exposure conditions:

1. P-biased condition. In this condition, the magnocellular system's processing contribution was compromised, but not that of the parvocellular system. This condition involved filtering the stimulus photos so that there was virtually no luminance contrast that would create a magnocellular signal.

2. M-biased condition. In this condition the stimulus photos were filtered so that there was no colour contrast, but there was luminance contrast. This ensured that the stimuli would create a reasonably strong signal in the magnocellular system, but not immediately in the parvocellular system.

It was predicted that when stimuli were presented in the **P-biased** condition, the traditional IAT effect would be attenuated because the contribution of the M-system which theoretically predisposes one to a more automatic, reflexive response style would be inhibited. This condition would facilitate more active, deliberative processing and responses would be less determined by stereotypical, perceptual automaticity.

It was predicted that when stimuli were presented in the **M-biased** condition, responses would be unchanged from the usual pattern of IAT scores seen and the same familiar pattern of in-group preference evident in the previous 'Race' experiment would be seen. This was effectively the control condition.

4.4.2 Participants

A total of 151 participants were recruited from two sites: UCT (n = 58), UKZN (n = 93). There was a significantly higher proportion of female than male participants from UCT (56 Females, 2 males) and a slightly lower proportion of female than male participants at UKZN (74 Females, 19 Males) χ^2 (1) = 8.6, p = 0.003 (asymptotic, 2-sided). The mean age for participants was 19.4 (SD = 1.1) and 22.0 (SD = 3.46) for UCT and UKZN participants respectively. There was a significant difference in age between the sites F(1) = 4.043, p = 0.046. Participants were given course credit at both sites for participation.

.

The same grouping strategy used for the previous experiment was repeated since the ethnic composition of the samples were very similar. The major groups are comparable with the previous experiment ('Black': 40.4% / 46%; 'Mixed' 59.6 % / 54% respectively).

4.4.3 Materials

The same application was used to upload the new stimuli and re-configure the experiment remotely – see Figure 113. The faces were the same as those used in the previous experiment

(column 1 in Figure 126). They were converted first to line drawings in Adobe Photoshop™ by using the High Pass Filter function set to 4 pixels width, then this was converted to an outline with the Photocopy filter with detail and contrast settings of 24 and 50 respectively. A threshold setting of 235 was used to darken the outline to pure black: RGB(0,0,0) against a pure white background: RGB(255,255,255) (column 2 in Figure 126). Finally, the pictures were transformed with the application described previously with the background colour to RGB(100,183,240) and the details to RGB(95,121,185) for the practice presentations (column 3 in Figure 126). The practice pictures were not presented in full colour because the other presentations were in P-biased or M-biased colours and it seemed wise to present the practice pictures in a line format that was similar to the P/M presentations but with colours that were easily visible. The other photos were M-Biased (RGB(43,43,43/30,30,30)) or P-Biased (RGB(0,216,230/0,255,0)) (column 4 - 5 in Figure 126).

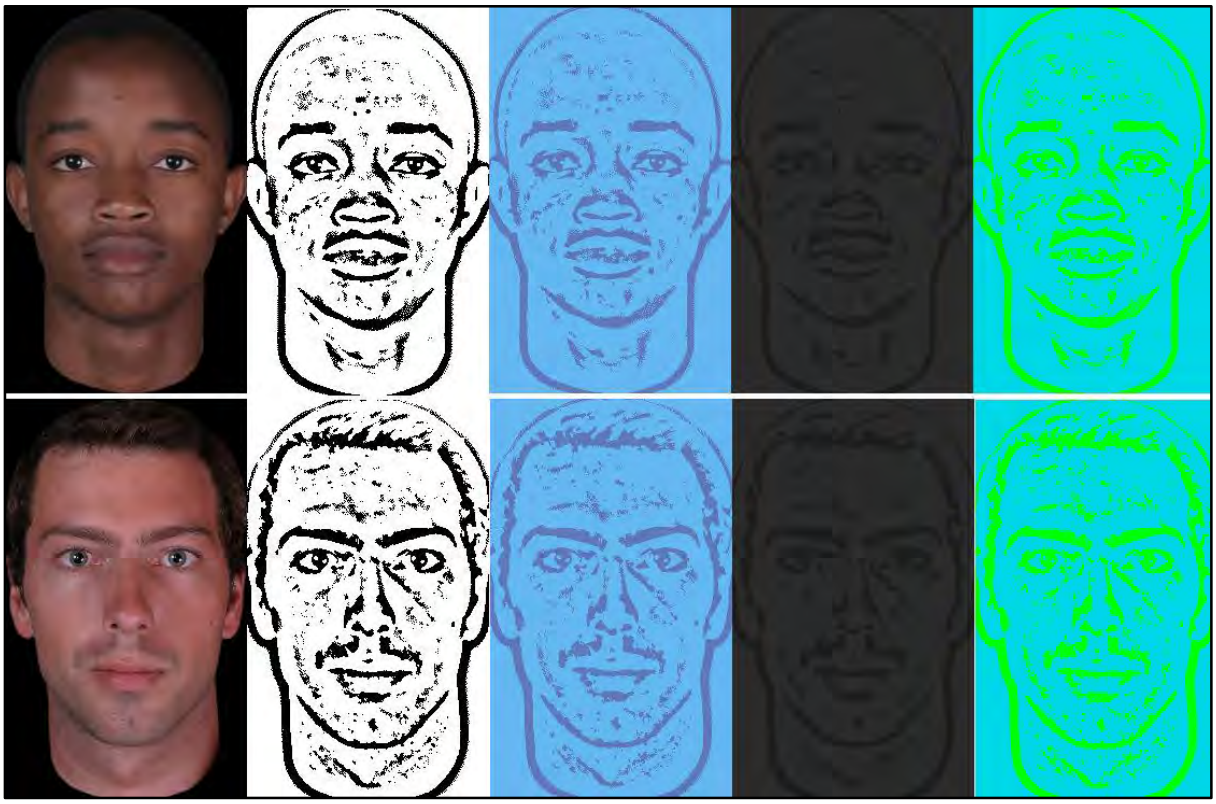


Figure 126, Original (column 1), line, practice (column 2), M-biased (column 3), P-biased drawings (column 4).

The colours were not determined from the P-calibration procedure from the previous experiment since I was not confident that there was enough consistency in the recorded data. The 'Prescribed Solution' used in the replication study was conservative, although detection levels were still reasonable at RGB(0,150,70/0,150,0). However, a compromise between the P-biased colours used for Experiment 3 (RGB(0,255,255 / 0, 255, 0)), the M-biased colours in Experiment 3 (RGB(119,119,119/145,145,145)) was made by darkening the M-biased

presentations significantly to RGB(43,43,43/30,30,30), and the P-biased presentations to RGB(0,216,230/0,255,0) for this IAT experiment. Both the M-biased and P-biased presentations were thus at a relatively high contrast level in this experiment compared with the values that had actually been determined in the calibration procedure (the replication determined that a contrast level of $\bar{\times} = 3.6$, (SD = 0.96) was adequate.

No changes in the software installed at the UCT laboratory was necessary, as all settings were determined by data in a database table which the program would download to local files after the participants logged in. The stimuli and instruction screens were loaded from these files at runtime. The same 'Pleasant' and 'Unpleasant' words were used, and the format of the experiment was identical to the previous one, except for the instruction screens. The new stimuli that were uploaded (see Figure 126 for an example) were derived from the same original set used in IAT experiment 2. There were 10 pictures of ‘Black’ males, and 10 pictures of ‘White’ males used. The same word stimuli were also used as described in Appendix 1 ‘Pleasant/unpleasant word lists’. The instruction screens were also changed slightly to alert participants to the fact that the images would be presented in different colours.

The program was installed on 20 workstations at the laboratory at UKZN, and the connection was set to a database server on the local area network. It was installed on about 6 workstations at the UCT laboratory where the database server was visible on the local network, but had a static IP, so that it could be accessed remotely.

4.4.4 Procedure

The same procedure was followed as described in the Race experiment, except that the images for the different phases varied in the manner described. Participants at the UCT site were recruited via a web notice, and the UKZN participants were recruited from an undergraduate course. All participants received course credit for participating.

4.4.5 Results

4.4.5.1 Analysis of summary data

A comparison of the confidence intervals of the mean d scores for only the 'Black' and 'White' South African groups reveals a clear group difference (Figure 127).

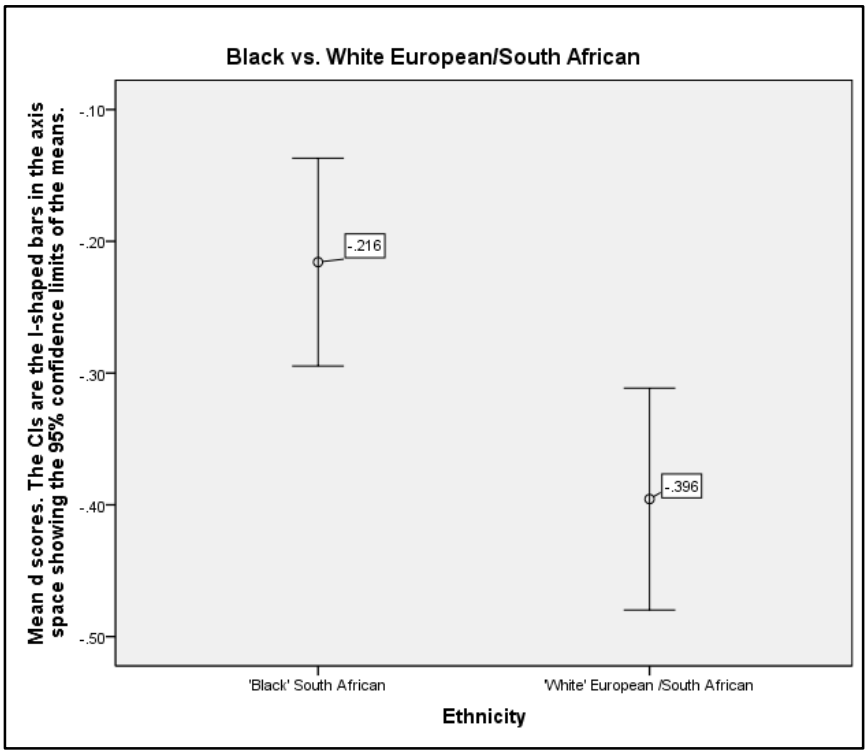


Figure 127. 95% CI of d means between 'Black' / 'White' groups.

An analysis of variance shows a significant difference ($F(1) = 9.71$, $p = 0.002$). The levels of d for these groups are similar to those seen in the previous experiment ('Black' South African, $\bar{x}$ = -0.216 (SD= 0.305) / -0.225 (SD=0.413); 'White' South African $\bar{x}$ = -0.396 (SD=0.33) / -0.439 (SD=0.317)).

The 'White' South African group means and the 'Mixed' ethnic group means are similar (-0.396, SD=0.33) vs. -0.386, SD=0.3) respectively). A plot of the confidence intervals of the means reveals a group difference (Figure 127). The ANOVA is significant ($F(1) = 11.56$, $p = 0.001$). The scores of this grouping compares favourably with those seen in the previous (IAT experiment 2) where there was no manipulation of the images (-0.439, SD = 0.317 vs. -0.435, SD = 0.336, for 'White' and 'Mixed' groups respectively). See Appendix 5, Figure 146 for a plot of the 95% CI of the d means of each sub-group. This illustrates the similarity between the 'Minorities' and 'White' group and the differences between these groups and the 'Black' ethnic group.

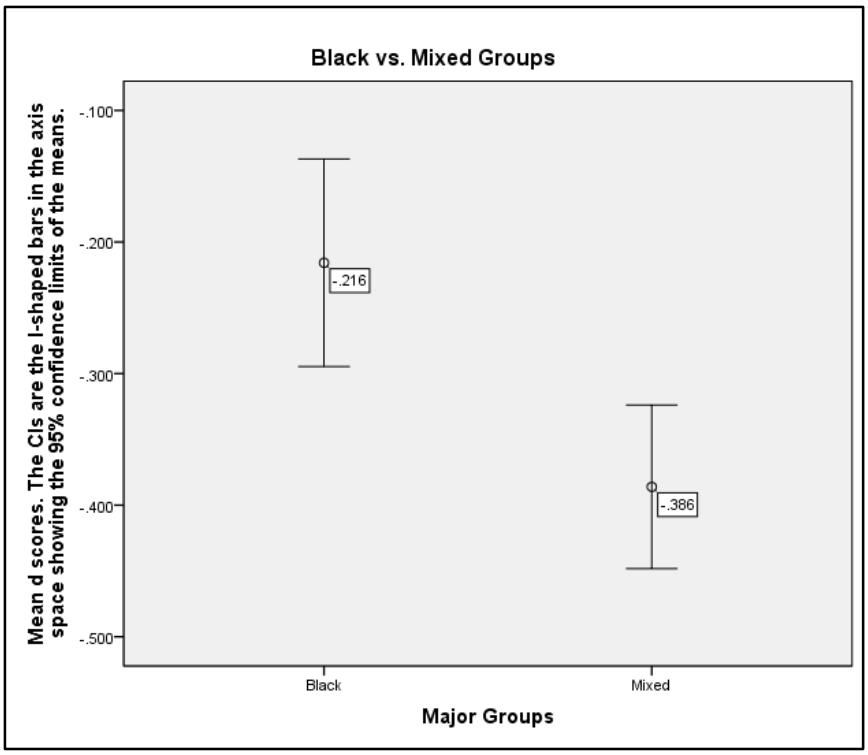


Figure 128. 95% CI of d means between 'Black' / 'Mixed' ethnic groups.

An analysis of variance shows a non-significant difference ($F(1) = 2.583$, $p = 0.11$). Figure 129 shows overlapping confidence intervals between the group means and if any effect exists, it is modest. This may be partly due to the distribution of participants between the two colour conditions and although this does not deviate from chance, there were more P-biased than M-biased trials in the UCT sample (see Figure 130) and the groups are unbalanced. The null hypothesis is retained.

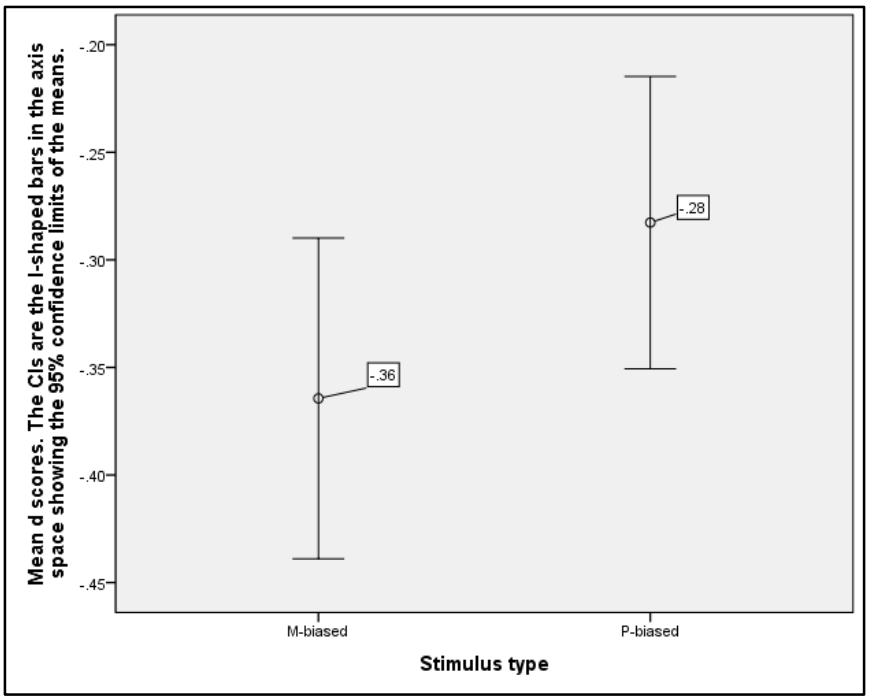


Figure 129. 95% CI of mean d scores for M- and P-biased conditions. Error bars show the 95% confidence limits of the mean for each group.

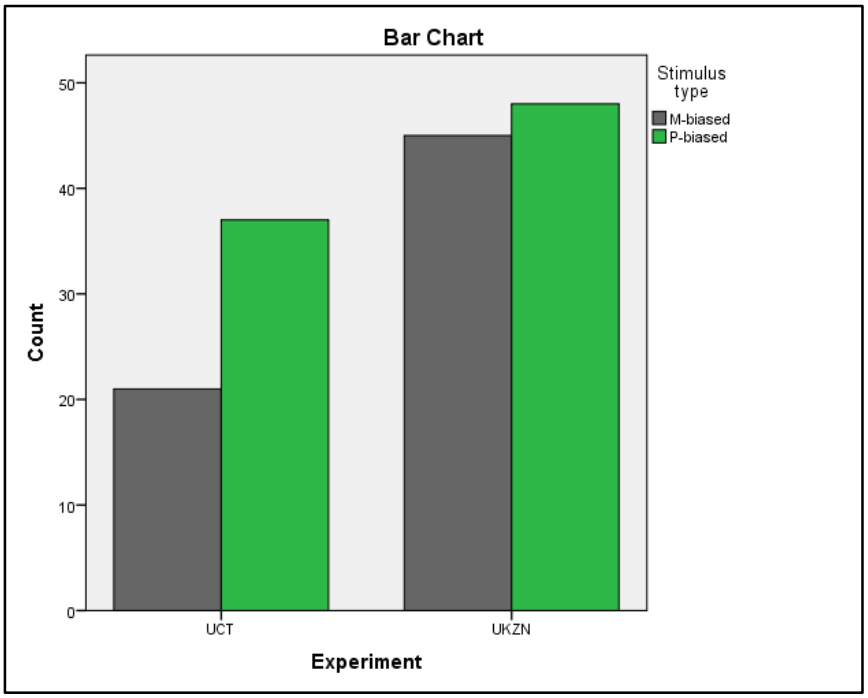


Figure 130. Frequencies of participants and Stimulus types between UCT and UKZN sites.

4.4.5.2 Reaction time analysis

It proved useful in the previous experiment, to analyse the entire data set comprising all the trials for all participants. This analysis illustrated interactions and nuances that were not apparent in the summary data set where for each case, the reaction time for each block is averaged and the d score is calculated from the two critical trial blocks (3, 5).

It was predicted that the d score mean would be more neutral when the items were presented in the Parvo-biased condition than in the Magno-biased condition and this does appear to be the case although the effect is small. Since the 'Black' group has been observed to show low levels of bias ($\bar{x}$ = -0.225), it is not clear how much these scores, which are already close to neutral, could be attenuated by presenting items in the P-biased condition. The reaction time analysis from the previous experiment also showed that 'Blacks' mean reaction time for pictures was slower in block 3. Their response pattern was also different in block 5 where, unlike the 'Mixed' ethnic group, there was little difference between their responses to words vs. pictures. There were also differences associated with gender, whereby males' mean reaction times were generally slower than females, especially in block 5. These findings suggested that more complex dynamics could be revealed by the detailed reaction time analysis.

The IAT reaction time data set consisted of 21140 rows (140 trials * 151 participants). The distribution of the reaction time data was also skewed and was similar to that of the previous reaction time data. The best distribution was seen with a reciprocal transformation. A total of 170 cases (0.8%) in the tail area were listed as extreme values and these were eliminated. The reciprocal value of 0.00255 was used as a cutting score and this corresponds to 392 ms. The distribution parameters were then as follows: skewness = 0.022, kurtosis = -0.41, $\bar{x}$= 0.001244, median = 0.001263, mode = 0.001565. The histogram is shown in Figure 131.

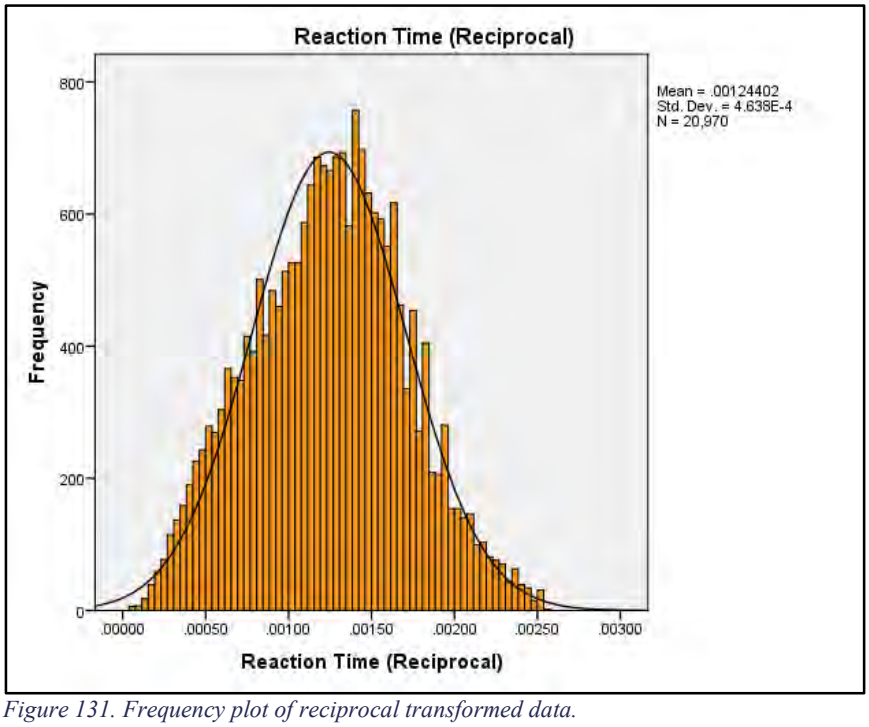


Figure 131. Frequency plot of reciprocal transformed data.

A factorial ANOVA was designed with *Reaction Time* as the dependent variable and *Block* (3, 5), *Gender, Stimulus Type (*M-biased/P-biased), *Item Type* (word/picture) and *Ethnic Grouping* as the independent variables. Reaction time data from blocks 3 and 5 were used since these are the critical blocks from which the d score is computed, and blocks 1, 2 and 4 were excluded.

The overall model was significant: $F(31) = 27.73$, $\eta\rho^2 = 0.067$, $p < 0.001$.
The following 3-way interactions were significant:
*Gender * Stimulus Type * Ethnic Grouping* ($F(1) = 19.64$, $\eta\rho^2 = 0.002$, $p < 0.001$)
*Gender * Item Type * Ethnic Grouping* ($F(1) = 6.206$, $\eta\rho^2 = 0.001$, $p = 0.013$)

The following 2-way interactions were significant:
*Block * Ethnic Grouping* ($F(1) = 7.07$, $\eta\rho^2 = 0.001$, $p = 0.008$)
*Gender * Item Type* ($F(1) = 10.66$, $\eta\rho^2 = 0.001$, $p = 0.001$)
*Gender * Ethnic Grouping* ($F(1) = 41.25$, $\eta\rho^2 = 0.003$, $p < 0.001$)
*Stimulus Type * Item Type* ($F(1) = 6.83$, $\eta\rho^2 = 0.001$, $p = 0.009$)
*Stimulus Type * Ethnic Grouping* ($F(1) = 87.21$, $\eta\rho^2 = 0.007$)
*Item Type * Ethnic Grouping* ($F(1) = 4.21$, $\eta\rho^2 = 0.001$, $p = 0.04$)
There were significant main effects for *Block, Stimulus Type, Item Type and Ethnic Grouping*.

I will discuss the 3-way interactions, then refer to the 2-way interactions. The untransformed variable (*Reaction Time*) is plotted. Although the main effects are qualified by the interactions, I will refer to them for interest's sake as they illustrate the effects of various variables in an accessible manner.

*Gender * Stimulus Type *Ethnic Grouping*
This interaction shows that 'Black' participants' reaction times tended to be longer when items were presented in the M-biased condition, but participants in the 'Mixed' ethnic group had shorter mean reaction times in the M-biased condition and longer reaction times in the P-biased condition (see Figure 132). The interaction shows how *Stimulus Type* was associated with different reaction time patterns between males and females in the different ethnic groupings. Males in the 'Mixed' group had the fastest reaction times ($\bar{x}$ = 853 ms) in the M-biased condition and males in the 'Black' group had the slowest ($\bar{x}$ = 1313 ms). This is a

mean difference of 460 ms. ‘Black’ males and females showed similar a response pattern to each other with longer reaction times to M-biased presentations than P-biased presentations and the opposite tendency was seen in the ‘Mixed’ group.

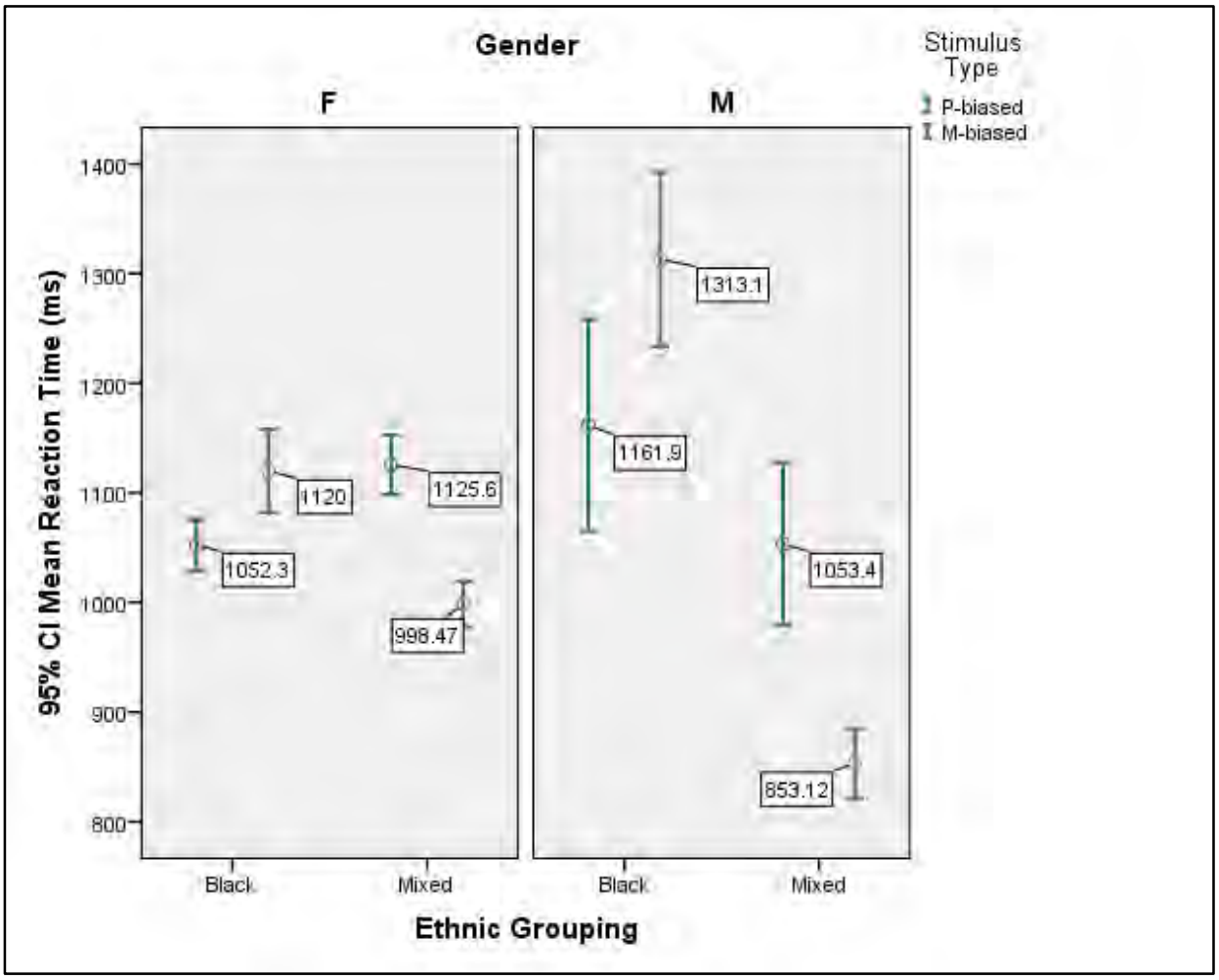


*Figure 132. Gender * Exposure Type * Ethnic Group interaction. Error bars show the 95% confidence limits of the mean for each group. Note that ‘Black’ participants, both male and female, show the same tendency to longer reaction times in the M –biased condition, whereas participants in the ‘Mixed’ group show relatively shorter reaction times in the M-biased condition. There is a large difference between males’ scores in the M-biased condition (460 ms), but not in the P-biased condition.*

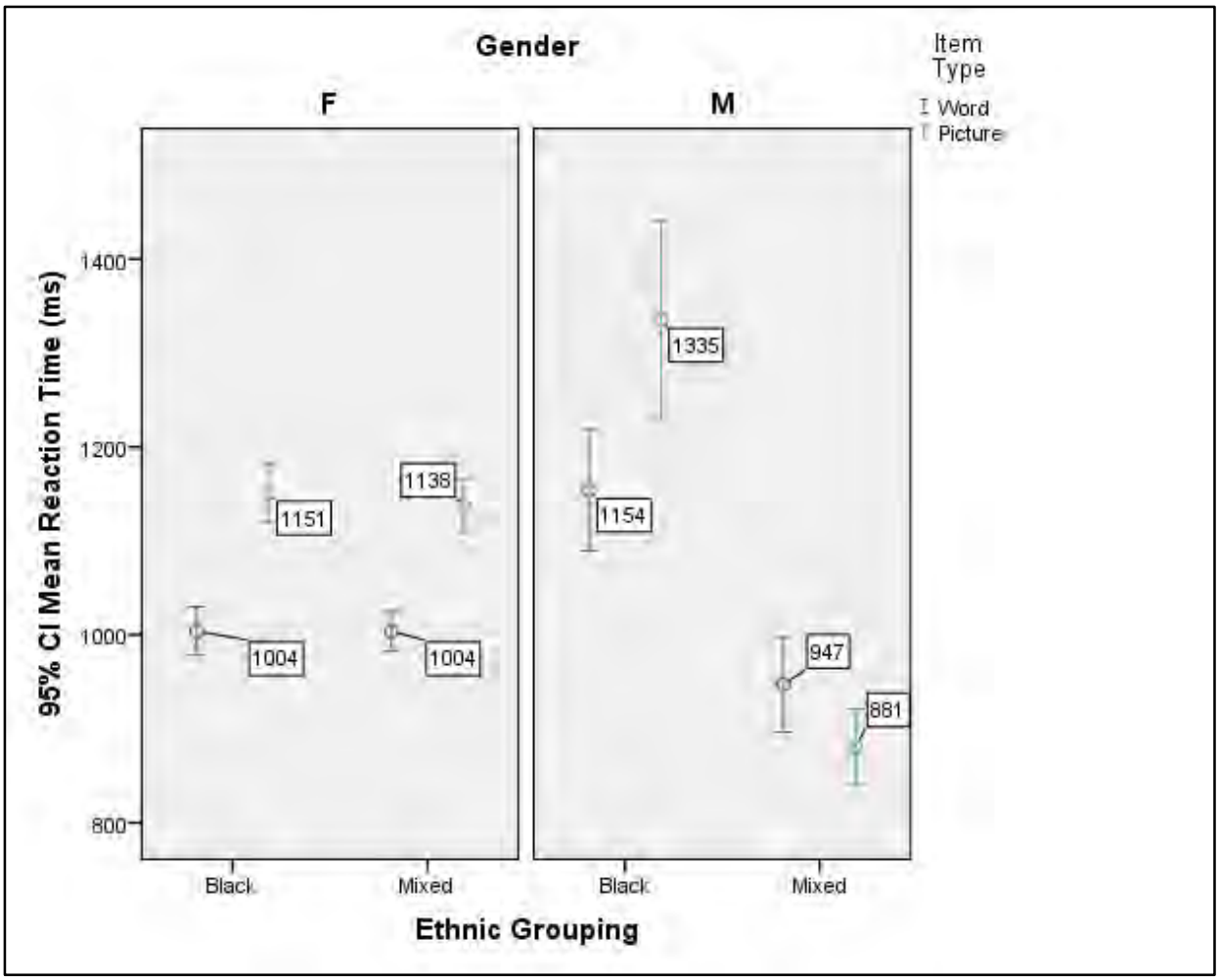


*Figure 133. Gender * Item Type * Ethnic Grouping Interaction. 'Black' males and males from the 'Mixed' ethnic group differ markedly with faster reaction times, especially for faces.*

*Gender * Stimulus Type * Grouping*

Figure 133 shows that female participants' response patterns were similar between the different ethnic groups with regard to the dependent variable 'Item Type' (words or pictures). Males' response patterns were strikingly different. In the 'Black' group, reaction time was longest for pictures ($\bar{x}$ = 1335 ms) and mean reaction time for pictures in the 'Mixed' ethnic group was 881 ms. This ('Mixed') group tended to be different to all the other sub-groups in that their reaction times were faster. They showed the opposite tendency from the other group which had shorter mean reaction times to words and longer reaction times to pictures.

*Block * Ethnic Grouping Interaction*

This interaction shows that both ethnic groupings had similar mean reaction times in block 5, but the 'Mixed' ethnic group had a relatively shorter mean reaction time in block 3 (Figure 134). This suggests that the 'IAT effect' involves differential performance in block 3, rather

than block 5 and it involves facilitation for both groups, but more so for the 'Mixed' ethnic group. A similar picture was seen in the previous 'Race' IAT where a higher degree of facilitation is seen for the 'Mixed' ethnic group in block 3 (see Figure 124).

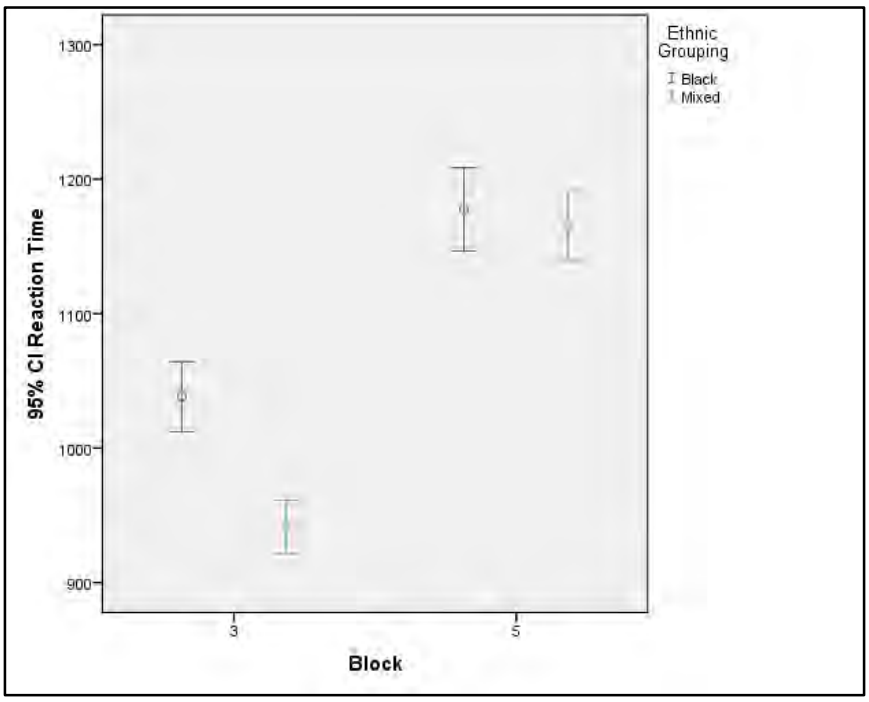


*Figure 134. Block * Ethnic Group interaction. Note the shorter reaction time for the 'Mixed' group in Block 3.*

*Stimulus Type * Item Type interaction*.

Figure 135 shows that the mean reaction time to pictures shown in the P-biased condition was 115 ms longer than for pictures shown in the M-biased condition. The reaction time for words is similar between exposure conditions and this was expected because the colour condition did not affect word exposures. The mean reaction time for pictures shown in the M-biased condition was 60 ms longer than the mean reaction time to words.

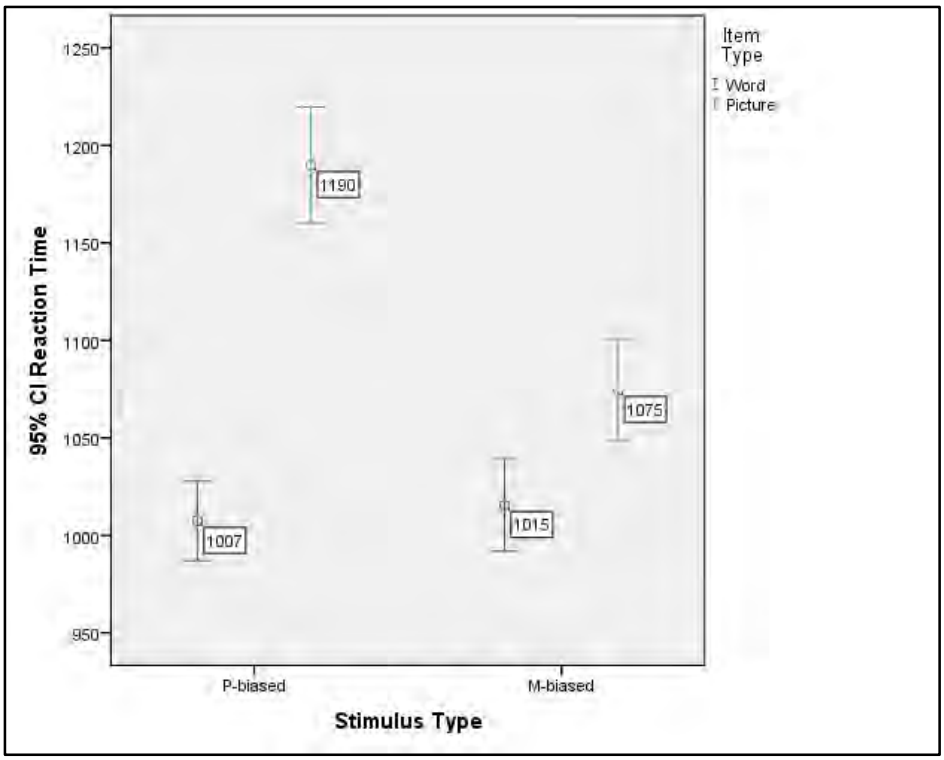


*Figure 135. Exposure Type * Stimulus Type interaction. Reaction times were 115 ms faster when faces were presented in the M-condition.*

The question which has not yet been answered is about whether presenting pictures in the P-biased condition slowed perception and identification of the faces in such a way that the 'automatic' response bias was reduced. It was expected that any top-down processing speed advantage and response 'potentiation' would be eliminated in this condition and that this would facilitate a slower appraisal and response with reduced automaticity. The effect seen when the d scores were compared between the M- and P-biased conditions is not significant and although there is a trend, this general conclusion cannot be drawn.

In order for the P-biased condition to interfere with this implicit response bias, reaction times to an 'easy' sort (e.g. block 3 where 'White' participants find it easy to sort the following items together: white faces and pleasant; black faces and unpleasant) should be **increased** and reaction times to a 'hard' sort (e.g. block 5 where 'White' participants find it more difficult to sort the following items together: black faces and pleasant; white faces and unpleasant) should be about the **same**. In other words, there should be a differential effect whereby the P-biased condition should slow sorting in one block (e.g. block 3) but not in the other block (e.g. block 5). Thus, the easy sort should be inhibited or slowed in the P-biased condition because the M-system mediated processing speed advantage is cancelled and if there is no fast projection of low spatial frequency information to activate an early 'initial

guess', the response should be delayed until a percept is built from the slower bottom-up signal. The critical point is that the top-down process is thought to facilitate rapid recognition and response and if this process is not available, decisions would have to be made more slowly, deliberatively and perhaps with more conscious effort.

The following analysis is an ***illustration*** of how response bias might be reduced in the IAT if the presumed contribution of the M-system was blocked by using P-biased stimuli. The data for this analysis was drawn from a sub-group of the IAT experiment, using the summary data. This sample consisted of participants who showed a relatively high level of implicit bias (as measured by the d score). The remainder of the sample showed low relatively low levels of bias. Figure 136 shows the means of these groups.

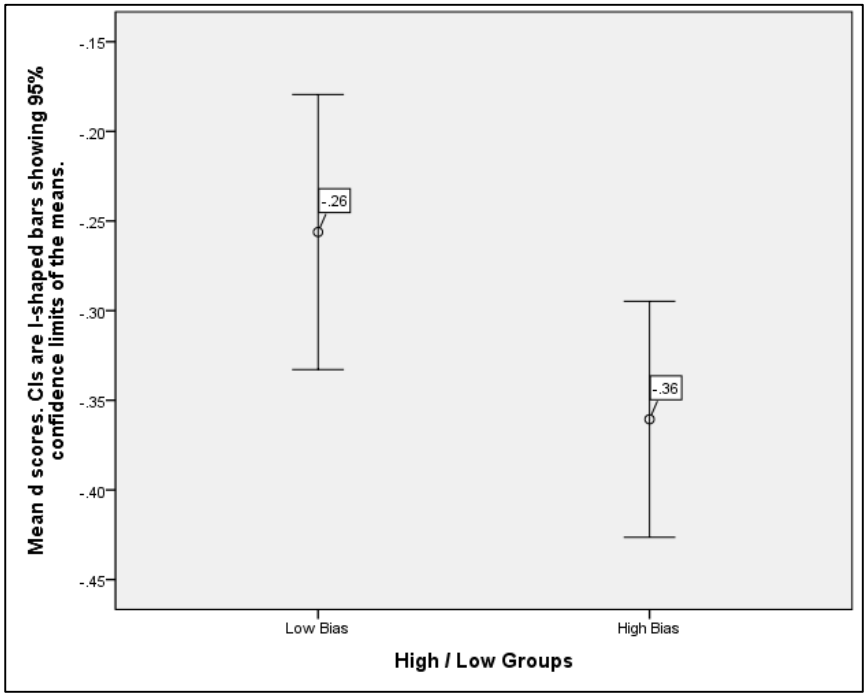


Figure 136. Low and High bias sub-groups'd scores.

There is a significant difference between these groups' d scores ($F(1) = 4.17$, $p = 0.043$).

The *High Bias* and *Low Bias* groups contain 90/61 participants respectively (no cases were removed from the dataset). See Appendix 5, Figure 147 for the composition of these groups. Figure 137 show the IAT scores of participants in the M- and P-biased conditions. In the M condition, d scores are more negative ($\bar{x} = -0.45$) and in the P condition, scores are less negative ($\bar{x} = -0.30$). This is consistent with predictions. Figure 138 shows the mean reaction times for blocks 3 and 5 for M- and P-biased conditions. Note that the mean reaction time for block 3 for the P-biased condition is higher than the reaction time mean for the M-biased

condition ($\bar{x}$ = 1000, $\bar{x}$ = 888 ms respectively) and the confidence intervals do not overlap. In block 5 however, the mean reaction time for the P-biased condition is not substantially (or significantly) higher than that for the M-biased condition and the confidence intervals overlap.

This illustrates the prediction made about how elimination of the top-down, magnocellularly based signal by the use of P-biased stimuli would tend to slow responses in the sorting task where the implicit association between the objects and categories is 'easy'.

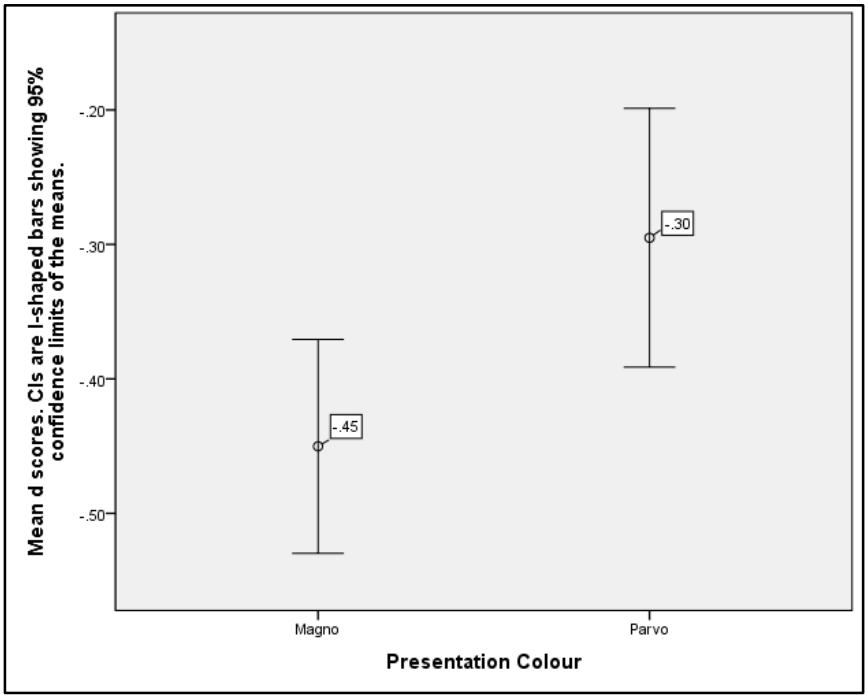


Figure 137. D scores for M- and P-biased conditions.

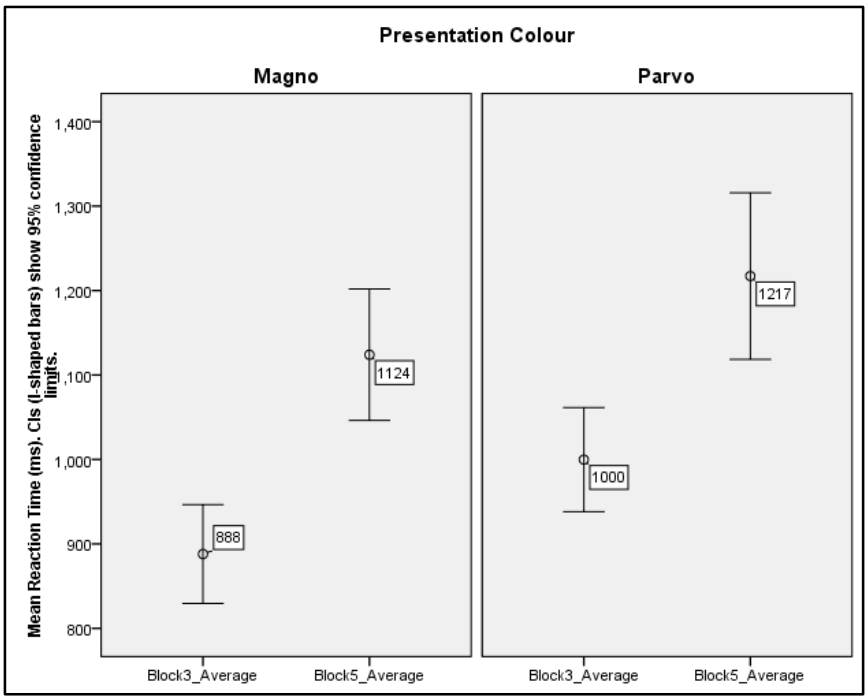


Figure 138. Reaction time means for critical trial blocks for M- and P-biased conditions.

While this analysis uses authentic data from the experiment, it was not presented to establish that the predicted effect *actually exists*. Rather, it illustrates and visualises *how* such an effect might appear in the IAT.

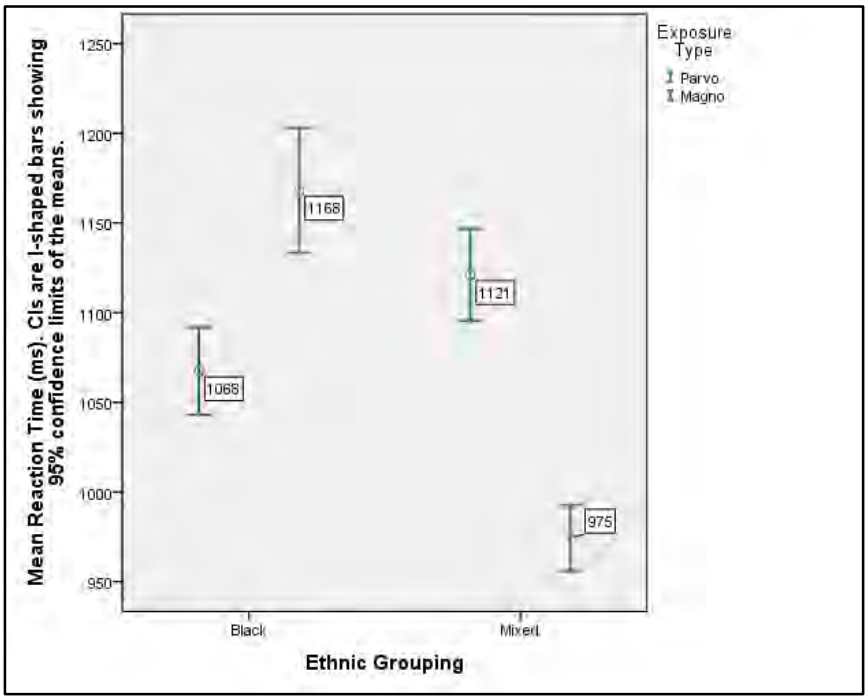


*Figure 139. Ethnic Group * Exposure Type interaction.*

Returning to the reaction time analysis, the mean reaction times for the critical blocks (3, 5) shows that while the M-biased condition facilitated faster responses in the 'Mixed' ethnic

group, it tended to have the opposite effect in the 'Black' group. This might explain why the effect of the P/ M-biasing manipulation on the IAT summary (d) score was weak. The effect was cancelled out because its direction was opposite in the different sub-groups. This is illustrated in Figure 139.

What accounts for this difference in reactivity to the colour condition? Part of the answer may lie in the fact that the 'Black' group showed low basal levels of implicit bias (in both experiments) and this may explain why they were not reactive to manipulation (see Figure 140).

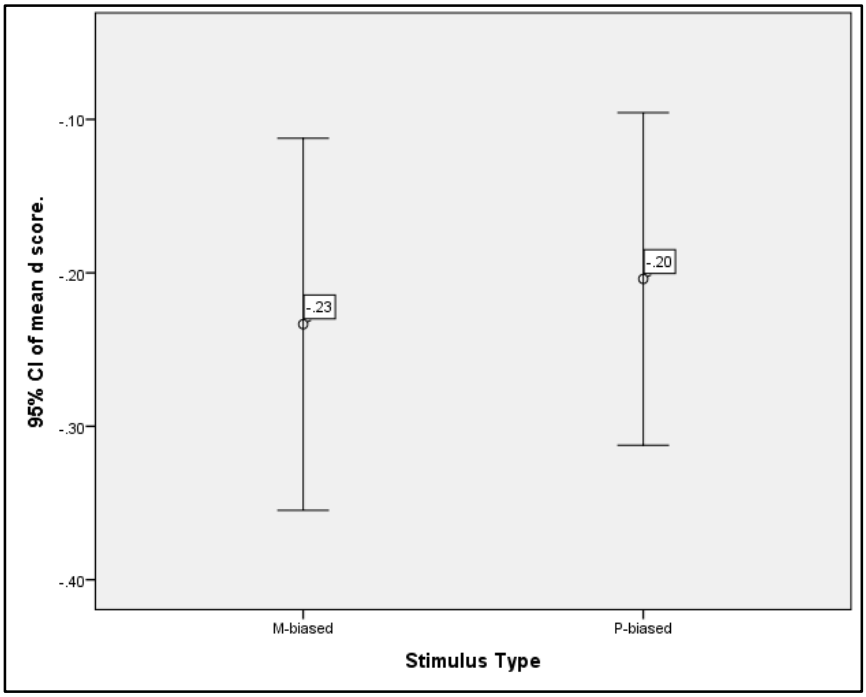


Figure 140. Mean IAT d scores for 'Black' group for M- / P-biased conditions.

Response times varied between the different types of stimuli (pictures/words), but only pictures were presented in the M- or P-type condition, so it was interesting to examine the reaction time data for only the pictures. There were two categories: pictures of black / white faces. Is there any evidence that the different groups showed any same/other race group reaction time differences in a colour condition that was not biased towards any visual system (M/P)? Blocks 2 and 4 required participants to sort faces as black or white with a key press. The face pictures were presented in an easily visible dark/light blue (column 3 in Figure 126). This is illustrated in Figure 141.

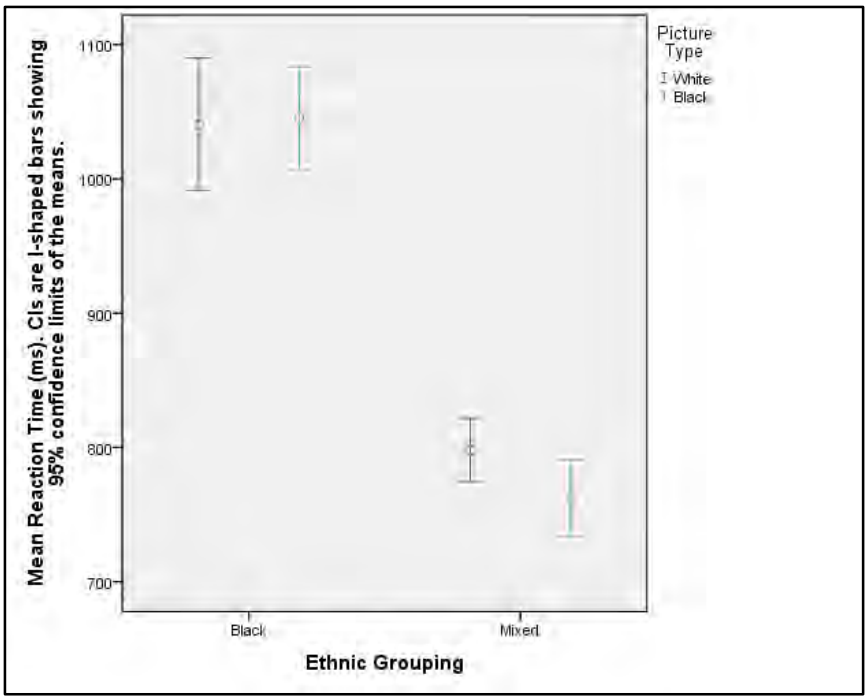


Figure 141. Mean reaction times for neutrally coloured (neither M- or P-biased) pictures (blocks 2 and 4).

There was no clear race-specific effect in the task-neutral blocks where pictures were presented in the easy dark/light blue condition although there is a trend for slightly faster sorting of black faces in the 'Mixed' group. Reaction times in the 'Mixed' group are faster by just over 200 ms.

The next question that was explored concerns reaction times associated with the different Stimulus Types, Picture types (black/white), Ethnic Groups and trial Blocks. Reaction times for word stimuli were excluded as these were not affected by the M- / P- manipulation. This is shown in Figure 142.

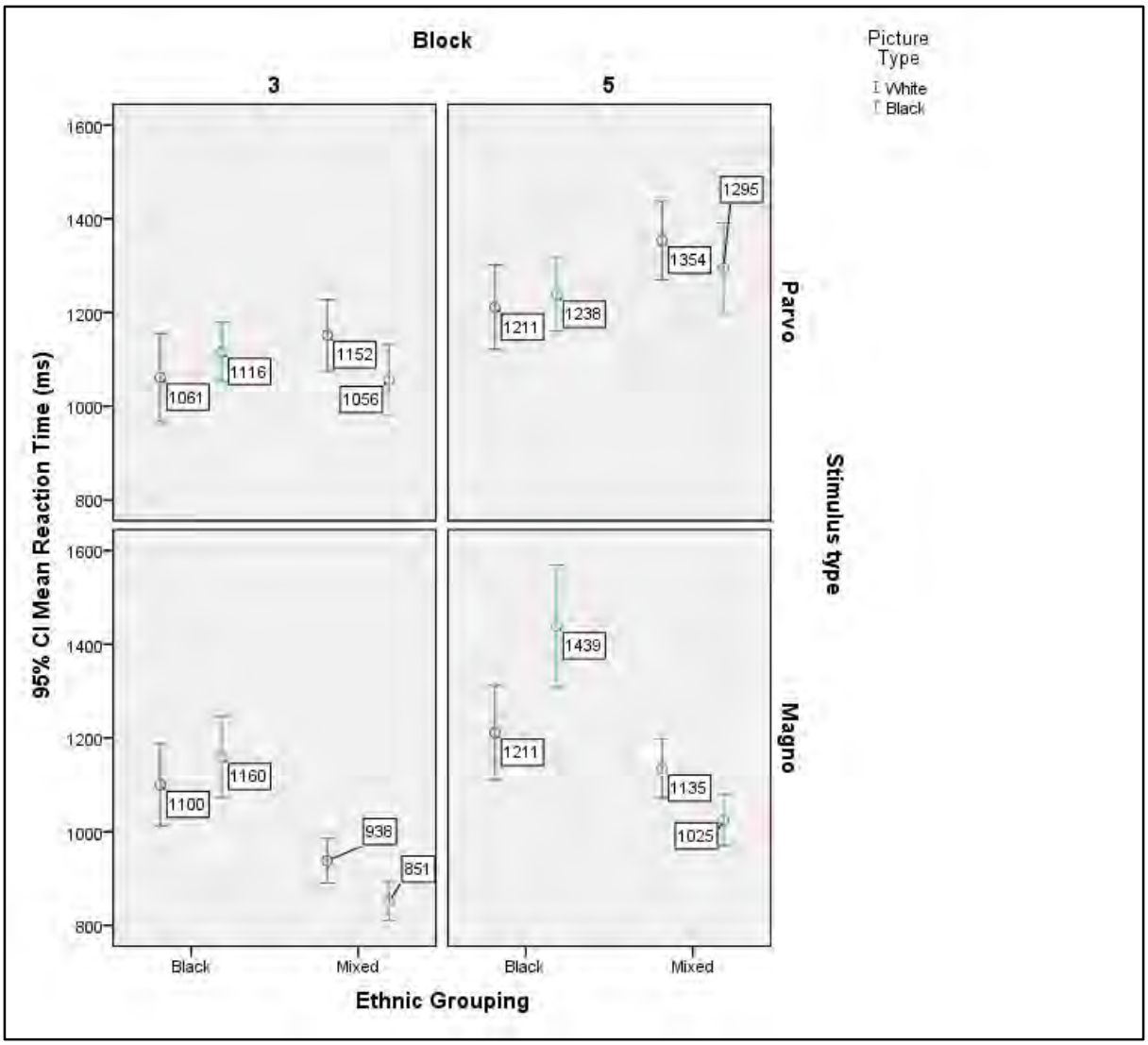


*Figure 142. Ethnic Grouping * Stimulus Type * Picture Type * Block analysis. Note the relatively even reaction times within (and even between) the different trial blocks in the Parvo condition. This is not true for the Magno condition where the different ethnic groups respond differently – most noticeably in block 5 where 'Blacks' and the 'Mixed' groups respond oppositely.*

This analysis is shown in Figure 142. Reaction times within blocks 3 and 5 are relatively even in the P-biased condition, especially within the different trial blocks. There do not seem to be any differences in reaction time for Picture type (white/black faces) for any of the groups in either block 3 or 5 in the P-biased condition. It seems as though there is no perceptual discrimination in this condition and the reaction times suggest that the task difficulty was fairly even for both ethnic groups, for the different picture types within each block. This pattern is different in the M-biased condition. In block 3 there is a difference between the reaction times of the 'Black' and 'Mixed' ethnic groups. In the 'Mixed' group, the net effect of presenting pictures in the M-biased condition seem to be associated with facilitation in both blocks. There is in increase in reaction time differences between black and white pictures, suggesting increased perceptual discrimination. In the 'Black' group, the only

shift is seen in block 5 where reaction time to black pictures is substantially slower than in block 3. This is probably associated with the sorting task, which involves sorting black faces with pleasant words.

There seems to be some task facilitation in the M-biased condition for the 'Mixed' ethnic group whose IAT scores were 'pro-white'. The facilitation, in terms of reaction time was about 200 ms in both blocks. Comparing block 5 between the P- and M-biased conditions, the most striking difference is seen in the mean reaction times for sorting black faces. In the P-biased condition, reaction times are similar between the 'Black' and 'Mixed' groups, but in the M-biased condition there is a mean difference of over 400 ms, whereas the confidence intervals for sorting white faces overlap in both the P- and M-biased conditions. This seems to have been a particularly difficult sort for the 'Black' group in the M-biased condition, and a relatively easy sort for the 'Mixed' group.

The analysis which follows confirms the pattern which was observed above. A factorial (univariate) ANOVA was done with 4 independent variables (*Ethic Group, Picture Type, Block, Stimulus Type*). *Picture Type* refers to 'white'/'black' faces, and *Stimulus Type* refers to the M-/P-biased condition.

The overall model is significant: $F(15) = 25$, $\eta\rho^2 = 0.059$, $p < 0.0005$.
The following 4-way interaction is significant: *Ethnic Group * Picture Type * Block * Stimulus Type:* $F(1) = 4.38$, $\eta\rho^2 = 0.001$, $p = 0.036$.
The following 2-way interactions are significant:
*Ethnic Group * Picture Type*: $F(1) = 46.5$, $\eta\rho^2 = 0.008$, $p < 0.0005$.
*Ethnic Group * Stimulus Type*: $F(1) = 83.7$, $\eta\rho^2 = 0.014$, $p < 0.0005$.

The following main effects were significant:
Ethnic Group: $F(1) = 56.2$, $\eta\rho^2 = 0.009$, $p < 0.0005$ (the 'Black' group's reaction times were significantly longer).
Block: $F(1) = 112.2$, $\eta\rho^2 = 0.018$, $p < 0.0005$ (reaction times were significantly longer in block 5).
Stimulus Type: $F(1) = 28.9$ $\eta\rho^2 = 0.005$, $p < 0.0005$ (reaction times were significantly longer in the P-biased condition).

The following plot (Figure 143) shows the reaction times in ms (not the reciprocal value):

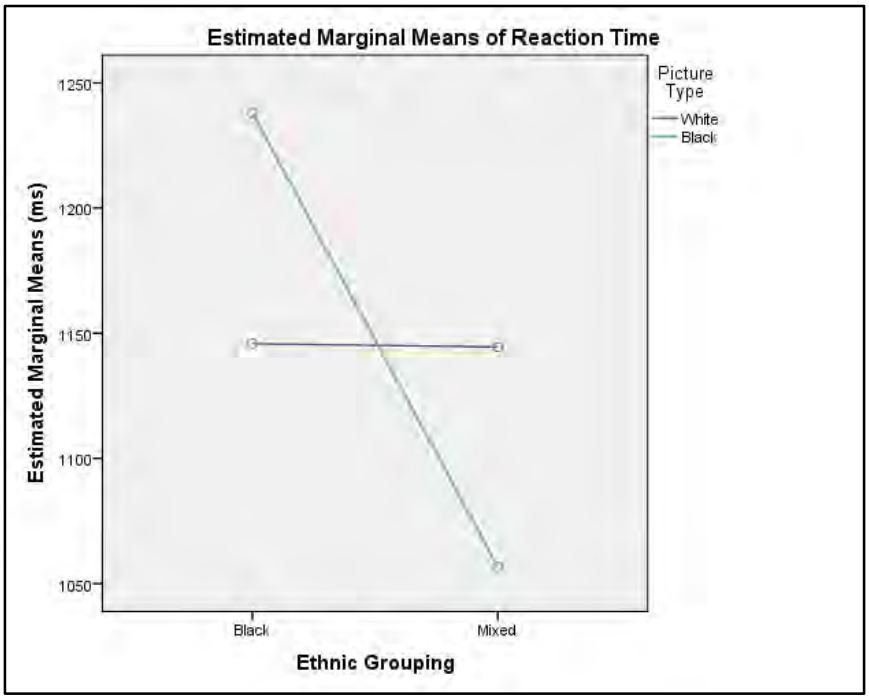


*Figure 143. Ethnic group * Picture Type interaction. Reaction times were similar for pictures of white faces between the ethnic groups, but the 'Black' group's mean reaction time for pictures of black faces was nearly 200 ms longer than that of the 'Mixed' group.*

Figure 144 shows the interaction between ethnic group and presentation colour (M/P).

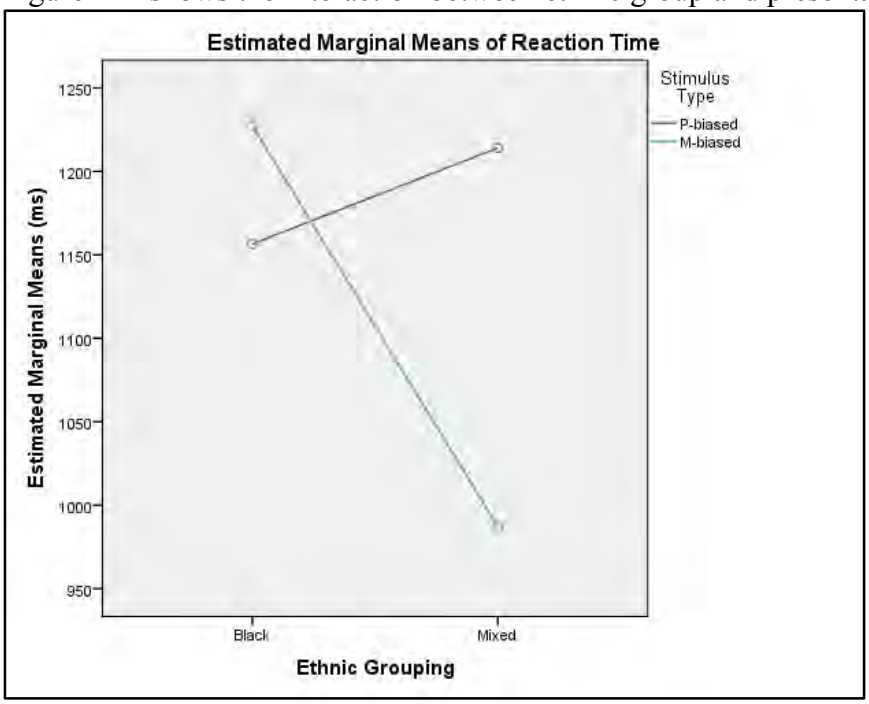


*Figure 144. Ethnic group * presentation colour condition. There was a much greater difference in reaction time for the 'Mixed' ethnic group between the M/P conditions, with a faster mean reaction time in the M-biased condition. The 'Black' group was hardly reactive to the colour condition.*

The 4-way interaction was illustrated in Figure 142.

I will attempt to interpret this pattern of findings.

1. The ‘Black’ group is unreactive to the Stimulus condition in block 3 and the reaction times are similar for black and white faces. Their performance is not facilitated in the M-biased condition in block 3. This group is not reactive to race in terms of the picture type.
2. The most difficult sort for the ‘Black’ group is in block 5 where black faces are sorted to pleasant words. The mean reaction time for sorting pictures of white faces to unpleasant words is the same between the P- and M-biased conditions (1211 ms) but the reaction time for sorting black faces to pleasant words is 200 ms longer. It was similarly easy to sort black faces to unpleasant words and white faces to pleasant words in block 3, so these associations seem to have been unproblematic. Yet associating black faces with pleasant words seems to have been more difficult.
3. This analysis suggests that participants in the ‘Black’ group have basically low levels of bias compared with the ‘Mixed’ group. In fact, they find the black-unpleasant / white/pleasant sorts equally easy, but the black/pleasant sort most difficult. This is contrasted with the white/unpleasant sort where the confidence intervals overlap with all of the other tasks.
4. The ‘Mixed’ group is reactive to Stimulus condition and their performance is facilitated in the M-biased condition. A tendency towards stimulus (race) discrimination emerges in the M-biased condition. It is difficult to characterise this as simple ‘race’ preference in terms of the reaction times, since the reaction times tend to be shorter for black faces than white in both blocks. However, the ability to distinguish the race identity of the faces seems to be enhanced in the M-biased condition.
5. The easiest sort for this group was black and unpleasant and the most difficult was white and unpleasant.
6. The significance of point 5 should not be overstated, but it does appear that this group is far more reactive to the racial identity of the faces. This appears to be a salient stimulus characteristic which is accentuated in the M-biased condition. The general pattern of bias is a preference for white race, but the most interesting feature is the sensitivity of this group towards race.

4.4.6 Discussion

4.4.6.1 Summary data set analysis

Comparison of the 'Black' and 'White' ethnic groups revealed the expected difference in the d score measure and re-confirmed expectations that 'White' European South Africans would show in-group bias (pro-white). Despite the fact that the stimulus pictures were varied between being M- and P-biased, the overall d score means are similar to the first race experiment. Combining the various other ethnic minority groups with the 'Whites' into a 'Mixed' ethnic group did not result in a substantial change to the mean d score (-0.386 vs. -0.396) and the difference between this group and the 'Black' South African group was significant.

Although the ANOVA comparison showed a non-significant difference associated with colour condition (M- vs. P-biased) when the summary IAT d score was used and the difference was in the predicted direction, the null hypothesis was retained. This analysis did suggest a trend and there were indications of more complex dynamics - see Figure 139 - which shows that the different ethnic groups reacted oppositely to each other with regard to the exposure condition. This probably cancelled out the overall effect of the exposure condition in the summary d score.

4.4.6.2 Reaction time analysis

The mean reaction time to pictures shown in the P-biased condition was 115 ms longer than for pictures shown in the M-biased condition. This supports the prediction that processing time would be longer when the M system contribution was inhibited. This needs to be further discussed in relation to the specific interactions that were seen.

There were differences in response to the different exposure conditions which suggest race-specific effects. In general, the 'Mixed' ethnic group's performance was facilitated when pictures were presented in the M condition, but the reverse was true for the 'Black' group. Detailed analysis provides some evidence of race-specific facilitation effects for the mixed ethnic group in the M condition, but not in the P condition. In the mixed ethnic group, reaction time patterns in the M condition suggest that there was task facilitation in both critical blocks and this facilitation was related to distinguishing race features in the stimulus

pictures. Response times were considerably more reactive in the M- than the P-condition, regardless of ethnic group and the interpretation may be that race features were processed more efficiently in that condition, whereas they tended to be masked in the P-condition. In other words, perceptual discrimination seems to have been poorer and slower in the P-condition.

There is a difference in reaction patterns that suggests different perceptual dynamics between the groupings. The interpretation may be that ‘Black’ participants show less perceptual discrimination. Interestingly, the most difficult sort (inferred from the long reaction times) for this group was block 5 where the sorting task was black/pleasant. Reaction times for the reverse sort in block 3 (black/unpleasant, white/pleasant) were similar.

Chapter 5. General Discussion and Conclusion

5.1 Introduction

In this chapter, I will summarise and briefly review the findings of all the experiments. I do not wish to simply reiterate the theoretical discussions and points which were made previously for each set of findings, but I would like to synthesise and integrate them. The research process has explored a number of phenomena related to the magnocellular system: word recognition, object recognition and implicit cognition. There has also been extensive consideration on how to go about using flicker photometry methodology and approaches to altering text and images in order to bias them towards the magnocellular or parvocellular systems. I have made some suggestions about methodology that are informed and qualified by the visual perceptual theory I reviewed. This has taken the process of theoretical and practical reflection into some complex and specialised areas. Since this has already taken up much time, energy and space in this dissertation, I will devote most of this discussion to the psychological aspects of implicit cognition and especially the findings which are potentially of most interest to the current research literature and milieu.

5.2 Summary of findings

Experiment 1 produced some evidence that word recognition accuracy was impaired under an experimental condition which Chase *et al.* (2003) predicted would lead to suppression of the M system. The finding agreed with that of Chase *et al*., although the sample size was small and the effect was modest.

Experiment 2 was done to test different colour combinations for the M / P –biasing conditions. Based on various computer models and simulations, the best combination seemed to be cyan/green in order for the M system to be effectively inhibited. Word recognition accuracy was significantly impaired in this condition, although reaction time did not appear to be affected.

Experiment 3 explored object recognition as a prelude to running an attempted replication of the research of Kveraga, *et al.*, (2007b). A similar cyan/green combination was used to inhibit the magnocellular system. The results showed significantly faster response latencies for the M-biased vs. the P-biased condition, but the accuracy scores did not specifically show that recognition accuracy was worse in the P-biased condition.

Experiment 4 was a follow up study to Experiment 2, but the colour combination (red/green) suggested by Kveraga *et al.* (2007b) was used so that the results could be compared with those seen when the apparently more feasible blue/green colour combination designed to inhibit M-system functioning was used. The experiment showed that the red/green isoluminance values did not significantly impair word recognition and there was no difference between the M- and P-biased conditions with regard to either accuracy or reaction time.

Experiment 5 (Replication) was a successful replication of the experiment by Kveraga, *et al.*, (2007b). The same flicker photometry method was used to calibrate values for the isoluminant (P-biased) condition and data was collected that supported the cyan/green colour combination, but not the red/green combination. The object recognition accuracy results agreed with Kveraga *et al*. and the reaction times were compatible with their results.

IAT Experiment 1: Species bias test. This experiment implemented an early version of the IAT which examines implicit associations between two kinds of species (insects/flowers) and various evaluative categories (pleasant/unpleasant). The experiment was carried out to test that the IAT program worked reliably and that the results were comparable to those obtained by other researchers. The experiment was successful and the results were compatible with those of Greenwald, *at al.* (1998).

IAT Experiment 2: Race test. This experiment was carried out in order to obtain baseline data from participants in two locations (Western Cape, KwaZulu-Natal). The experiment showed classic 'implicit race' results that have been seen in many other experiments. 'White' participants had scores that suggested implicit ingroup preference, whereas 'Black' participants had more neutral scores. The 'Race' IAT is usually done with 'White' and 'Black' participants, so there was a question about how to group 'Coloured', 'Indian' and other ethnic minorities. The alternative was to exclude data from these groups. Grouping these minorities with 'White' participants hardly change the mean scores that were seen in the 'White' group and it appears that this 'Mixed' ethnic group is fairly homogenous. Score patterns were distinctly different from those of the 'Black' group and this may be partly due to the political legacy of this country where 'Blacks' were in a uniquely different position politically under the *Apartheid* regime. A decomposition of the IAT scores suggests that the summary IAT score for the 'White' and 'Mixed' sub-groups results from sorting facilitation

in block 3 (sorting white/pleasant, black/unpleasant), rather than task difficult in block 5 (sorting black/pleasant, white/unpleasant). This inference is based on the fact that similar reaction time means were seen between these groups in block 5, but lower mean reaction times were seen for the 'Mixed' group in block 3. In short, this suggests that the 'IAT effect' seen in the 'Mixed' ethnic group which showed pro-white bias, is based on facilitation of in-group-good/out-group-bad sorting rather than task difficulty associated with out-group-good/in-group-bad sorting. Detailed reaction time analysis was helpful and established that females' reaction times were more consistent across ethnic groups and males' reaction times were sharply different (with 'Black' males showing much longer reaction times than the other sub-groups). Reaction times were generally shorter for picture stimuli than word stimuli.

IAT Experiment 3: IAT Race test with M/P-biasing intervention.

In this experiment some procedural innovation was used to investigate hypotheses regarding automaticity mediated by the M system. The IAT summary scores were consistent (in magnitude and direction) with the previous experiment and very similar scores were seen in the ethnicity groupings. The effect of the intervention was weak in the summary (d) score and the null hypothesis was retained. Analysis of the reaction times again proved useful. One of the most interesting findings is that there is a consistent and robust effect of Stimulus Type (M / P) between the different ethnic groupings. Responses were generally slower in the 'Black' group in the M condition (suggesting non-facilitation) but faster in the 'Mixed' ethnic group in the M condition (suggesting facilitation). Differences in facilitation seemed to be most marked between the 'Black' male and 'Mixed' males sub-groups which followed the same pattern of non-facilitation and facilitation, respectively. It seems likely that the opposite response patterns between the different groups cancelled out the effect at the level of the summary d score.

Presenting pictures in the M-biased condition was associated with shorter reaction times (about 115 ms), bearing in mind that this (2-way) interaction needs to be interpreted and qualified by the 3-way interactions that were recently discussed.

The differences in patterns of facilitation / non-facilitation associated with the Stimulus type and ethnic group, together with the different pattern of reactivity between item type suggest that some perceptual and cognitive dynamic was activated when faces were presented in the M-biased condition. The dynamic was evident as facilitation in the 'Mixed' ethnic group, and

non-facilitation in the 'Black' group. As the 'Black' group tended to show neutral IAT scores, it is possible that the task facilitation seen in the 'Mixed' group when pictures were shown in the M-biased condition, is associated with a perceptually-based mechanism (or form of 'automaticity') that reflects biased processing of race features.

5.3 Discussion

The goal of the research project was to design a series of experiments which progressively and incrementally tested ideas about the magnocellular system and its possible role in implicit cognition.

What has this research achieved?

Evidence has been presented which shows that the M system has a role in facilitating word recognition and object recognition. When word stimuli were presented in an isoluminant (P-biased) condition, recognition accuracy was degraded. When words were presented in the low luminance contrast (M-biased) condition, recognition accuracy was significantly better.

The question of how to achieve isoluminance was addressed 1) theoretically and 2) empirically.

1) The theoretical answer to this question seems to be that the overall light levels have a bearing on what colour combinations are perceived as isoluminant. While the traditional red/green colour combination is often used, there are theoretical reasons why a cyan/green combination is more suitable.

2) Research data from Experiment 2 supports the idea that isoluminance may be achieved by using the cyan/green colour combination whereas Experiment 4 does not support the red/green isoluminance values suggested by Kveraga, *et al*., (2007b). Experiment 5 provides further evidence that this colour combination is a valid solution in that it was the most commonly selected one in the calibration phase.

Replication of the Kveraga, *et al*. (2007b) experiment was important and the fact that compatible results were achieved suggests that they might be compatible with their research findings. In another sense, because the Kveraga, *et al*. study included fMRI imaging as well as data from the cognitive task and the data collected in Experiment 5 agrees closely with

theirs, it would be interesting to see if similar imaging data could be found in a replication of the current study.

The idea which Bar (2003), Bar (2004), Bar, *et al.* (2006), Kveraga, Ghuman & Bar (2007), Kveraga, Boshyan & Bar (2007b) have outlined involves bi-directional processing in the visual system. They argue that 'the traditional view that visual input is processed serially, in a bottom-up cascade of cortical regions that analyze increasingly complex information…' (Kveraga, *et al.,* 2007, p. 145) does not seem viable and cannot account for the efficiency of visual perception. They suggest that top-down processes are needed to facilitate the processing of a visual stream of information, in order to speed up recognition and to allow for the integration of other kinds of sensory information so that predictions can be made. Predicting things that might happen involves accessing knowledge and applying this quickly to interpret the current state of the world and how it might change from moment to moment (*ibid.*). Their proposal involves the application of associative knowledge which is thought to be available in the orbitofrontal cortex (OFC) and which is considered to be a 'key multimodal association region' (Barbas, 2000; Kringel-bach & Rolls, 2004 – both in Kveraga, *et al.*, 2007). As described in the Review of Literature, the essential idea is that 'a coarse, low spatial frequency representation of the input image is rapidly extracted and projected to OFC from early visual or subcortical regions' (Kveraga, *et al.,* 2007). They regard the M system as the most likely source of this low spatial frequency information and suited to the task of conveying rough, 'gist' information which involves global properties about an object's shape (Bar, 2003). This gist is back-projected to the temporal cortex where it is thought to constrain the number of possible object representations (Kveraga, et al., 2007b; Kveraga, et al., 2007; Bar, 2003). Neuroimaging data supports these propositions (Kveraga, et al., 2007; Kveraga, et al., 2007b).

Bar (2004) considers it likely that this low spatial frequency information is the basis for contextual information which facilitates object recognition, especially when the objects within the visual context are congruent with it (in other words, when objects fit the context). Bar argues that context information of this kind can be processed implicitly, that it can be learned 'incidentally' or 'without explicit intentions' and that '…subjects can recognise visual objects in contextual scenes in the "near absence of attention"' (p. 620).

An important question which this dissertation sought to address, concerns the processing of faces by this same (magnocellular) system and whether race-related facial features could be processed by this system.

There is no agreement about the existence of a specialised 'face recognition' area in the brain. Haxby, Hoffman & Gobbini (2000) have argued that it is distributed and that 'face-responsive' areas are not only responsive to faces, but they respond to other objects as well. This does not detract from the idea that face recognition is an important function of these areas, but Gauthier *et al.,* (1999 in Haxby, *et al.*) suggest that this might be an area that is utilised for recognising particular kinds of objects by experts or specialists (for example, cars and birds).

There are many references to face recognition in the literature, especially with regard to the role of the amygdala in recognising emotions, but there does not appear to be any research on the cognitive processing of race-related facial features. The literature that exists shows that to some extent, humans recognise faces from their own racial group better than from other races (Phelps, 2001) and that European Americans show a stronger same-race advantage than African Americans (*ibid*). The explanation of this usually involves familiarity and European Americans are thought to simply be more familiar with, and have more experience with their own racial group. Interestingly, a same-race advantage was found for inverted faces, but not for upright, or inverted faces from other races (Vizioli, Foreman, Rousselet & Caldara, 2010). These authors suggest that there is '…a fine grained neural tuning for same-race faces at early stages of processing…' (p. 1), and in their study, it was seen for both their Western Caucasian and East Asian participants. Their research raises some questions, not least the question of why the N170 signal[9] was not affected by upright faces and why it was sensitive to own-race inverted faces. They suggest that it is possible that some early processing is sensitive to race, but caution that further research is needed to confirm this. They note the possibility that the brain shifts from the usual holistic processing associated with upright faces and engages in featural processing with inverted faces (Rossion, 2008 in Vizioli, *et al*., 2010). Although this study produced evidence of race-related sensitivity in the N170, other studies have produced variable results (*ibid*.).

[9] The N170 is often studied because of its sensitivity to faces (larger responses to faces than other objects). Desjardins & Segalowitz, (2013) note that it is associated with 'automatic high-level integrative processing associated with expertise' (p. 2).

An earlier study (Holmes, *et al*., 2005) did not address the question of race sensitivity in ERP (event-related potentials) components, but investigated the role of spatial frequency for ERP components sensitive to faces and emotional facial expression. Part of their rationale for the study was based on previous findings that many subcortical structures, such as the superior colliculus, amygdala and pulvinar appear to be activated by low spatial frequency face-related signals, especially when they show fearful expressions. They reflected the suggestion that these signals emanate from the magnocellular system. Their results showed an increased positive signal at about 150 ms after stimulus onset (this appears to be associated with the P100 signal, although this is not specified), but no changes in the N170 component were seen.

They concluded that the N170 'face-sensitive' component was not reactive to any of the conditions (in terms of spatial frequency, or facial emotion) and suggest that detection of emotion does not involve these subcortical structures, but rather that it is processed by cortical brain systems. They also suggest that the positive component that was evident in frontal areas, might be associated with a pre-attentive mechanism involving the orbitofrontal cortex and amygdala. They state that this agrees with other research (unpublished at the time that their article went to press) that 'rapid attentional responses to fearful versus neutral faces were driven by LSF rather than HSF visual cues, in line with the role of the amygdala and orbitofrontal cortex in the mediation of attention towards threatening faces' (p. 517). The general conclusion they draw is that information processing in the prefrontal cortices is likely to be the correlate of the positive (150 ms) component and this activity is likely to reflect 'social interpretive processes, such as decoding the meaning of a specific facial expression' (p. 518). The co-activation of the orbitofrontal cortex and amygdala to emotionally charged stimuli may be important for 'priming autonomic and motor responses…' and '…modulating perceptual representations in sensory cortices…' (p. 517). It is this event-related positive component which has become the focus of some important research with regard to face processing.

In a recent article, Desjardins & Segalowitz (2013) described P100 components that reliably precede the N170 'face' signal by using independent component analysis (ICA) to extract these effects. They concluded that the P100 signal was face-specific and suggest that it is associated with low-level differences between stimulus categories and reflects early

perceptual processing in the visual system. They note the possibility that the P100 may be affected by top-down attention processes (Van Voorhis & Hillyard, 1977 – in Desjardins & Segalowitz, 2013).

The traditional view that the amygdala is involved in an early appraisal of LSF information via some 'quick and dirty' pathway has been challenged. Pessoa & Adolphs (2010) have argued that inputs to the amygdala are not directly from some part of the visual pathway. They suggest rather, that the amygdala receives input from a cortical circuit which is involved in the early processing of emotional visual stimuli, namely the orbitofrontal cortex (OFC). They suggest a revised role for the pulvinar, which has traditionally been seen as a passive relay and propose that it is active in coordinating information flow between various parts of the cortex and subcortex, especially the amygdala. Most of the input to the pulvinar comes from the cortex (p. 779) and the role of the pulvinar might be to amplify signals from the OFC, which they regard as having an important role in 'computing an object's biological value' (*ibid.*). This means that the amygdala receives efferent signals from the pulvinar that originate from the OFC and a circuit that involves the anterior cingulate. The pulvinar's role may be to modulate and coordinate responses to emotional stimuli (particularly the medial nucleus), and it may be involved in directing or recruiting attention when emotional stimuli are detected. In this early emotion detection circuit, the amygdala may be regarded as a 'convergence zone' for 'highly processed' sensory information and instead of it being a clever nucleus that detects danger, it is thought to have a role in the 'distribution and aggregation of information' (p. 780).

Pessoa & Adolphs agree with the idea that '…fast but coarse visual processing is just what an organism needs in a dangerous environment' (*ibid.*), and they agree with the proposal that the signal which initiates this fast response to emotional information is based on low spatial frequency information which originates from the early visual cortex and is projected via 'short-cuts' to the ventrolateral prefrontal cortex and particularly the OFC (*ibid*.), but they disagree that this *input* pathway involves the colliculus, pulvinar and amygdala. Lastly, their model supports the idea that LSF information provides a gist which facilitates recognition and regard the magnocellular system as the probable source of this information.

A recent article links face discrimination ability with the magnocellular pathway from a developmental perspective (Pallet & Dobkins, 2013). These researchers found a relationship

between age-related face vs. object discrimination and luminance contrast sensitivity. Interestingly, they obtained isoluminance values using red/green stimuli and applied the mean red/green value to the child participants. The deduction they made is that the changes in luminance (but not chromatic) contrast with developmental age were associated with the M pathway and that this weakly (and indirectly) supported their hypothesis that face processing is mediated at least partly, by the development of the M pathway. Although these researchers assert that their results '…add to the mounting evidence that the M pathway may influence face processing', they tend to draw their conclusion more from theoretical support than the actual results they found. They suggest 3 lines of evidence that support their conclusion.

Firstly, they assert that there are perceptual studies which show that face processing is better under stimulus condition that tap the M, more than the P pathway. They do not cite any studies which show this however. They then describe other approaches to biasing activity in the different visual pathways. They do make the point that the LSF M pathway might be involved in face processing because face recognition relies on holistic processing.

Secondly, they appeal to the traditional 'quick and dirty' subcortical pathway which involves the amygdala. They suggest that the amygdala 'face system' may be independent of the cortical face processing system. The fast subcortical visual pathway that projects to the amygdala is the LSF M system. Most of the literature they refer to has already been reviewed.

Thirdly, they refer to the top-down feedback pathway postulated by Kveraga, *et al.*, (2007). The role of the OFC is prominent in this idea and although the idea that top-down influences from OFC to the temporal cortex may be important in priming face representations in FFA (fusiform face area) is plausible, there is simply no empirical evidence which supports this idea directly.

5.4 Conclusion

I have presented a theoretical rationale and empirical evidence regarding the role of the M system in perception and cognition. There is a high degree of consensus in the literature on the structure and properties of the magnocellular system and this system was described in some detail, from the retina to the LGN and the cerebral cortex. The idea that information from the M system is the source for top-down processes which facilitate identification seems

plausible and there is research which supports this directly (fMRI imaging studies) and indirectly (reaction time and object recognition accuracy). Results of the research in this dissertation support the idea that the M system is involved in word and object recognition. I have also provided evidence that the M system facilitates recognition of race-related facial features. There appear to be race-specific processing patterns in terms of own-race vs. other race recognition and these appear to be associated with the M system. It seems that race-related feature sensitivity is associated with M system, but not P system processing. In the P condition, there was little sensitivity to race features, but in the M condition, this sensitivity appeared to be enhanced. There was evidence of race-related perceptual dynamics in the IAT experiment and these seem to signify that black participants process race-related features differently to other races, that they do not show such strong in-group preference and no own-race recognition advantage. The overall pattern suggests that black participants are simply less sensitive to race-related facial features than the other races in this study and this seems compatible with the general finding that they show less prejudice on implicit measures. The slower reaction times in this group may also be compatible with the finding that African Americans have greater amygdala activation when viewing 'Black' than 'White' faces which was evident in the perceptual encoding condition, but not in the verbal encoding condition (Lieberman, *et al*., 2005).

It seems likely that the automaticity which the IAT measures has to do with perceptual processes which are mediated by the M system and which involve top-down processes that authors like Bar (2003), Kveraga, *et al*., (2007, 2007b) describe. This is not a claim that perceptual automaticity is the basis for racial prejudice. Probably the reverse is true. As Phelphs & Thomas (2003) point out, 'Another way of phrasing the contact hypothesis is that face recognition is a skill that is learned over many years of practice in social situations.' (p. 747). It is probably the case that implicit race effects are difficult to control because there is a degree of automaticity in processes of visual perception which are difficult to disrupt. To some extent, the IAT may measure processes which are spontaneous and difficult to suppress, and which occur in everyday life. This may be part of the reason that it seems to predict behaviour better than explicit measures, which tap different cognitive processes.

This research has explored a number of areas of perception and cognition which involve the magnocellular system. The general conclusion of this research is that the M system seems to

be intimately linked with implicit perceptual and cognitive processes and the nature of this link could be described in terms of automaticity.

5.5 Limitations and future directions for research

Although the IAT seemed to be a reasonable choice for exploring implicit cognition in general and the role of the M system in particular, it turned out to be a difficult paradigm to work with. This is because within each block there are essentially different tasks that have to be performed. It does not involve simple disjunctive decisions (black or white) but decisions which involve combinations of adjectives and images (black/pleasant, white/unpleasant). This complicates inferences about perceptual decisions (black or white face) because each decision is coupled with a value judgement (pleasant or unpleasant). Although this method is basic to the logic of the IAT, the complexity of the tasks makes the difficulty of measuring simple reaction time greater. On the other hand, it is a fairly powerful and robust experimental technique and there are many measurements that can be done, directly and indirectly.

Reaction time data is very useful and is almost certainly associated with task difficulty, but future studies of this kind should probably go beyond reaction time and measure event related potentials as well. ERP technology has become a powerful way of exploring perceptual and cognitive phenomena. It would be important to replicate the findings of this study, using both reaction time and event related potentials so that race-related face processing can be studied using conceptual and technical tools that are in a newly advanced state of development. ERP and brain imaging may be useful in corroborating the reaction time data and yield further information on the brain regions that are associated with differences and nuances of face perception dynamics in different ethnic groups.

References

Adolphs, R. (2008). Fear, faces, and the human amygdala. *Current Opinion in Neurobiology*. *18*, 166–172. doi: 10.1016/j.conb.2008.06.006

Agnew, Z., Bhakoo, K., Puri, B. (2007).The human mirror system: A motor resonance theory of mind-reading. *Brain Research Reviews. 54*(2), 286 – 293. doi:10.1016/j.brainresrev.2007.04.003

Amodio, D., Lieberman, M. (2006). Pictures in our heads: Contributions of fMRI to the study of prejudice and stereotyping. In in T. Nelson (ed.) *Handbook of Prejudice, Stereotyping, and Discrimination*. NY: Earlbaum Press.

Amodio, D. (2014). The neuroscience of prejudice and stereotyping. *Nature Reviews Neuroscience*. *15*, 670-682.

Anderson, S., Bechara, A., Damasio, H., Tranel, D., Damasio, A. (1999). Impairment of social and moral behavior related to early damage in human prefrontal cortex. *Nature Neuroscience*. *2*(11), 1032-1037.

Anguera, J., Gazzaley, A. (2011). Dissociation of motor and sensory inhibition processes in normal aging. *Clinical Neurophysiology*. *123*(4), 730-740. doi:10.1016/j.clinph.2011.08.024

Anwar, S., Fang, H. (2006). An Alternative Test of Racial Prejudice in Motor Vehicle Searches: Theory and Evidence. The American Economic Review. *90*(1), 127 - 151.

Ashwin, C., Wheelwright, S., Baron-Cohen, S. (2005). Finding a face in the crowd: Testing the anger superiority effect in Asperger Syndrome. *Brain and Cognition*. *61*(1), 1-18. doi:10.1016/j.bandc.2005.12.008

Baars, B, (1994). A Thoroughly Empirical Approach To Consciousness. *PSYCHE*. *1*(6), 1-20.

Baars, B. (1998). *A Cognitive Theory of Consciousness*. New York: Cambridge University Press.

Banaji, M., Nosek, B., Greenwald, A. (2004). No Place for Nostalgia in Science: A Response to Arkes and Tetlock. *Psychological Inquiry*. *15*(4), 279–310.

Bar M., Kassam K.S., Ghuman, A.S., Boshyan, J., Schmid A.M., Dale, A.M., Hämäläinen, M.S. (2006). Top-down facilitation of visual recognition. *Proceedings of the National Academy of Science*. *103*(2), 449-454.

Bar, M. (2003). A Cortical Mechanism for Triggering Top-Down Facilitation in Visual Object Recognition. *Journal of Cognitive Neuroscience*. *15*(4), 600-609.

Bar. M. (2004). Visual Objects in Context. *Nature Reviews Neuroscience*. *5, 617-629*. doi:10.1038/nrn1476

Bargh, J. (1994). The four horsemen of automaticity: Awareness, intention, efficiency, and control in social cognition. In R. S. Wyer & T. K. Srull (Eds.), *Handbook of social cognition* (Vol. 1, pp. 1-40). Hillsdale, NJ: Erlbaum.

Barton, R. (2004). Binocularity and brain evolution in primates. *Proceedings of the National Academy of Sciences*. *101*(27), 10113–10115.

Bedny, M., Caramazza, A. (2011). Perception, action, and word meanings in the human brain: the case from action verbs. *Annals of the New York Academy of Sciences*. *1224*, 81–95.

Bedny, M., Caramazza, A., Grossman, E., Pascual-Leone, A., Saxe, R. (2008). Concepts Are More than Percepts: The Case of Action Verbs. *The Journal of Neuroscience. 28*(44), 11347-11353. doi:10.1523/JNEUROSCI.3039-08.2008

Bedny, M., Caramazza, A., Pascual-Leone, A., Saxe, R. (2012). Typical Neural Representations of Action Verbs Develop without Vision. *Cerebral Cortex. 22*, 286-293. doi:10.1093/cercor/bhr081

Blanton, H., Jaccard, J. (2008). Unconscious Racism: A Concept in Pursuit of a Measure. *Annual Review of Sociology*. *34*, 277-297.

Bordt, A., Hoshi, H., Yamada, E., Perryman-Stout, W., Marshak, D. (2006). Synaptic Input to OFF Parasol Ganglion Cells in Macaque Retina. *The Journal of comparative neurology. 498*(1), 46–57. doi:10.1002/cne.21040

Born, R., Pack, C. (2002). Integration of Motion Signals for Smooth Pursuit Eye Movements. *Annals of the New York Academy of Sciences. 956*, 453–455.

Bornstein, R., D'Agostino, P. (1992). Stimulus recognition and the mere exposure effect. *Journal of Personality and Social Psychology*. *63*(4), 545-552.

Botvinick, M., Cohen, J., Carter, C. (2004). Conflict monitoring and anterior cingulate cortex: an update. *Trends in Cognitive Sciences*. *8*(12), 539-545.

Breitmeyer, B. (1984). *Visual masking: An integrative approach.* Oxford, UK: Oxford University Press.

Breitmeyer, B.G. (2014). Contributions of magno- and parvocellular channels to conscious and non-conscious vision. *Philosophical Translations of the Royal Society of London*. *369*(1641). doi: 10.1098/rstb.2013.0213

Bullier, J. (2001). Integrated model of visual processing. *Brain Research Reviews. 36*, 96–107.

Burish, M., Kueh, H., Wang, S. (2004). Brain Architecture and Social Complexity in Modern and Ancient Birds. Brain, *Behavior and Evolution. 63*, 107–124. doi: 10.1159/000075674

Byrne, R., Corp, N. (2004). Neocortex size predicts deception rate in primates. *Proceedings of the Royal Society Biological Sciences. 271*, 1693–1699. doi: 10.1098/rspb.2004.2780

Byrne, R., Whiten, A. (1991). Computation and Mindreading in Primate Tactical Deception. In Whiten, A. (Ed.), *Natural Theories of Mind*. Oxford: Basil Blackwell.

Byrne, R., Whiten, A. (1991). Prediction and Mind-reading: The Evolution of Causal Understanding. In Whiten, A. (Ed.), *Natural Theories of Mind*. Oxford: Basil Blackwell.

Caceci, T. (2012). Processing of Visual Signals. Retrieved 07/04/2012 from: http://www.vetmed.vt.edu/education/curriculum/vm8054//eye/cnsproc.htm.

Cacioppo, J., Decety, J. (2011). Social neuroscience: challenges and opportunities in the study of complex behavior. *Annals of the New York Academy of Sciences. 1224*, 162–173. doi: 10.1111/j.1749-6632.2010.05858.x

Calkins, D., Sterling, P. (2007). Microcircuitry for Two Types of Achromatic Ganglion Cell in Primate Fovea. *The Journal of Neuroscience. 27*(10), 2646 –2653. doi:10.1523/JNEUROSCI.4739-06.2007

Callway, E. (2005). Structure and function of parallel pathways in the primate early visual system. *The Journal of Physiology*. *566*(1), 13–19. doi:10.1113/jphysiol.2005.088047

Chalmers, D.J. (1995). The Puzzle of Conscious Experience. *Scientific American. 273*, 80-86.

Chan, T., Martin, P., Clunas, N., Grünert, U. (2001). Bipolar Cell Diversity in the Primate Retina: Morphologic and Immunocytochemical Analysis of a New World Monkey, the Marmoset. *The Journal of Comparative Neurology. 437*, 219–239.

Chase, C., Ashourzadeh, A., Kelly, C., Monfette, S., Kinsey, K. (2003). Can the magnocellular pathway read? Evidence from studies of color. *Vision Research*. *43,* 1211-1222.

Cheng, A., Eysel, U., Vidyasagar, T (2004). The role of the magnocellular pathway in serial deployment of visual attention. *European Journal of Neuroscience. 20*, 2188–2192.

Coates, M., Campbell, K. (2010). Event-related potential measures of processing during an Implicit Association Test. *NeuroReport. 21*(16), 1029–1033. doi: 10.1097/WNR.0b013e32833f5e7d

Cole, G., Heywood C., Kentridge R., Fairholm I., Cowey, A. (2003). Attentional capture by colour and motion in cerebral achromatopsia. *Neuropsychologia. 41*(13), 1837–1846. doi:10.1016/S0028-3932(03)00184-2

Collet, M. 5 Most Disgusting Flowers on Earth. Retrieved 7 August 2013 from http://www.environmentalgraffiti.com/news-most-morbid-plants-earth.

Corbetta, M., Patel, G., Shulman, G. (2008). The Reorienting System of the Human Brain: From Environment to Theory of Mind. *Neuron. 58*(3), 306–324.

Coricelli, C., Critchley, H., Joffily, M., O'Doherty, J., Sirigu, A., Dolan, R. (2005). Regret and its avoidance: a neuroimaging study of choice behaviour. *Nature Neuroscience. 8*(9), 1255-1262.

Crick, F., Koch, C. (1995). Are we aware of neural activity in primary visual cortex? *Nature*. *375*(11 May), 121-123.

Crick, F., Koch, C. (2003). A framework for consciousness. *Nature Neuroscience. 6*(2), 119 - 126. doi:10.1038/nn0203-119

Crook, J., Manookin, N., Packer, O., Dacey, D. (2011). Horizontal Cell Feedback without Cone Type-Selective inhibition Mediates "Red–Green" Color Opponency. *The Journal of Neuroscience. 31*(5), 1762–1772. doi:10.1523/JNEUROSCI.4385-10.2011

Crook, J., Peterson, B., Packer, O., Robinson, F., Troy, J., Dacey, D. (2008). Y-Cell Receptive Field and Collicular Projection of Parasol Ganglion Cells in Macaque Monkey Retina. *The Journal of Neuroscience. 28*(44), 11277–11291. doi:10.1523/JNEUROSCI.2982-08.2008

Cyders, M., Smith, G. (2008). Emotion-based Dispositions to Rash Action: Positive and Negative Urgency. *Psychological Bulletin*. *134*(6), 807–828. doi: 10.1037/a0013341

Damasio, A.R. (1994). Descartes' Error: Emotion, Reason, and the Human Brain. New York: Putnam.

Dasgupta, N., Greenwald, A., (2001). On the Malleability of Automatic Attitudes: Combating Automatic Prejudice with Images of Admired and Disliked Individuals. *Journal of Personality and Social Psychology. 81*(5), 800-814.

Dasgupta, N., McGhee, D., Greenwald, A., Banaji, M. (2000). Automatic Preference for White Americans: Eliminating the Familiarity Explanation. *Journal of Experimental Social Psychology. 36*, 316–328.

Davidson, R., Irwin, W. (1999). The functional neuroanatomy of emotion and affective style. *Trends in Cognitive Sciences. 3*(1), 11-21.

De Gelder, B. (2010). Uncanny Sight in the Blind. *Scientific American. 302*(5), 60 - 65. doi:10.1038/scientificamerican0510-60

De Gelder, B., Van den Stock, J., Meeren, H., Sinke, C., Kret, M., Tamietto, M. (2010). Standing up for the body. Recent progress in uncovering the networks involved in the perception of bodies and bodily expressions. *Neuroscience and Biobehavioral Reviews. 34*, 513-527.

De Houwer, J., Moors, A. (2007). How to define and examine the implicitness of implicit measures. In B. Wittenbrink & N. Schwarz (Eds.). *Implicit measures of attitudes: Procedures and controversies* (pp. 179-194). New York: Guilford Press.

Dehaene, S., Changeux J., Naccache1, N., Sackur, J., Sergent, C. (2006). Conscious, preconscious, and subliminal processing: a testable taxonomy. *Trends in Cognitive Science. 10*(5), 204-211.

Desjardins, J., Segalowitz, S. (2013). Deconstructing the early visual electrocortical responses to face and house stimuli. *Journal of Vision*. *13*(5), 1–18. doi: 10.1167/13.5.22

Dew, I., Cabeza, R. (2011). The porous boundaries between explicit and implicit memory: behavioral and neural evidence. *Annals of the New York Academy of Sciences*. *1224,* 174–190.

Diller, L., Packer, O., Verweij, J., McMahon, M., Williams, D., Dacey, D. (2004). L and M Cone Contributions to the Midget and Parasol Ganglion cell Receptive Fields of Macaque Monkey Retina. *The Journal of Neuroscience. 24*(5), 1079-1088. doi: 10.1523/?JNEUROSCI.3828-03.2004

Dixon, J., Durrheim, K. (2003). Contact and the ecology of racial division: Some varieties of informal segregation. *British Journal of Social Psychology*. *42*, 1-23.

Dreher, B., Fukada, Y., Rodieck, R. (1976). Identification, Classification and Anatomical Segregation of Cells with X-like and Y-like Properties. *Journal of Physiology. 258*, 433 - 452.

Duncan, S., Feldman Barrett, L. (2007). Affect is a form of cognition: A neurobiological analysis. *Cognition and Emotion*. *21*(6), 1184-1211.

Durrheim, K. (2003). White opposition to racial transformation. Is it racism? *South African Journal of Psychology*. *33*(4), 241-249.

Durrheim, K., Dixon, J. (2004). Attitudes in the Fiber of Everyday Life: The Discourse of Racial Evaluation and the Lived Experience of Desegregation. *American Psychologist*. *59*(7), 626 –636. doi: 10.1037/0003-066X.59.7.626

Eger, E., Henson R., Driver, J, Dolan, R. (2007). Mechanisms of top-down facilitation in perception of visual objects studied by fMRI. *Cerebral Cortex. 17*(9), 2123–2133.

Eisenberger, N., Lieberman, M. (2004). Why rejection hurts: a common neural alarm system for physical and social pain. *Trends in Cognitive Sciences. 8*(7), 294-300.

Engel, A.K., Fries, P., Konig, P., Brecht, M., Singer, W. (1999). Temporal Binding, Binocular Rivalry, and Consciousness. C*onsciousness and Cognition. 8*, 128–151.

Fazio, R.H. Olson, M.A. (2003). Implicit Measures in Social Cognition Research: Their Meaning and Use. *Annual Review of Psychology. 54*, 297–327

Fabre-Thorpe, M. (2011). The characteristics and limits of rapid visual categorization. *Frontiers in Psychology*. Published online 2011 October 3. doi: 10.3389/fpsyg.2011.00243 <http://dx.crossref.org/10.3389%2Ffpsyg.2011.00243>

Fazio, R.H. Olson, M.A. (2003). Implicit Measures in Social Cognition Research: Their Meaning and Use. *Annual Review of Psychology*. *54*, 297–327.

Feldman Barrett, L., Russell, J. (1999). The Structure of Current Affect: Controversies and Emerging Consensus. *American Psychological Society. 8*(1), 10 - 14.

Field, G., Greschner, M., Gauthier, J., Rangel, C., Shlens, J., Sher, A., Marshak, D., Litke, A. (2009). High sensitivity rod photoreceptor input to the blue-yellow color opponent pathway in macaque retina. *Nature Neuroscience*. *12*(9), 1159-64. doi:10.1038/nn.2353

Fowler, N., Stein, J. (2005). Yellow Filters Can Improve Magnocellular Function: Motion Sensitivity, Convergence, Accommodation, and Reading. *Proceedings of the National Academy of Sciences*. *1039*, 283-293.

Frantz, C., Cuddy, A., Burnett, M., Ray, H., Hart, A. (2004). A Threat in the Computer: The Race Implicit Association Test as a Stereotype Threat Experience. *Personality and Social Psychology Bulletin*. *30* (12), 1611-1624. doi: 10.1177/0146167204266650

Franz, V., Gegenfurtner, K., Bulthoff, H., Fahle, M. (2000). No Evidence for a Dissociation Between Perception and Action. *Psychological Science*. *11*(1), 20-25.

Galaburda A, Livingstone M. (1993). Evidence for a magnocellular defect in developmental dyslexia. *Annals of The New York Academy of Sciences*. *14*(682), 70-82.

Gawronski, B., Balas, R., Creighton, L. (in press). Can the Formation of Conditioned Attitudes Be Intentionally Controlled? *Personality and Social Psychology Bulletin*. doi 10.1177/0146167213513907

Gawronski, B., Cesario, J. (2013). Of Mice and Men: What Animal Research Can Tell Us About Context Effects on Automatic Responses in Humans. *Personality and Social Psychology Review*. *17*(2), 187–215. DOI: 10.1177/1088868313480096

Gawronski, B., Creighton, L. (2013). Dual Process Theories. In D. E. Carlston (Ed.) *The Oxford handbook of social cognition*. Oxford University Press.

Gawronski, B., De Houwer, J. (in press). Implicit Measures in Social and Personality Psychology. In Reis, H. & Judd, C. (Eds.), *Handbook of research methods in social and personality psychology* (2nd edition). New York : Cambridge University Press.

Gawronski, B., Deutsch, R., Seidel, O. (2005). Contextual Influences on Implicit Evaluation: A Test of Additive Versus Contrastive Effects of Evaluative Context Stimuli in Affective Priming. *Personality and Social Psychology Bulletin. 31*(9), 1226 - 1236. Doi: 10.1177/0146167205274689

Gawronski1, B., Galdi, S. (in press). What Can Political Psychology Learn from Implicit Measures? Empirical Evidence and New Directions. *Political Psychology*. 1 - 17.

Gazzaniga, M.S. (2005). *Cognitive Neuroscience* (2nd ed.). New York: Norton.

Ghuman, A., Bar, M., Dobbins, I., Schnyer, D. (2008). The effects of priming on frontal-temporal communication. *The National Academy of Sciences USA*. *105*(24), 8405–8409.

Giesbrecht, B., Bischof, W. F., & Kingstone, A. (2003). Visual masking during the attentional blink: Tests of the object substitution hypothesis. *Journal of Experimental Psychology: Human Perception and Performance*. *29*(1), 238–258.

Giesbrecht, B., Bischof, W. F., & Kingstone, A. (2004). Seeing the light: Adapting luminance reveals low-levelvisual processes in the attentional blink. *Brain and Cognition*. *55*(2), 307–309.

Gómez, J. (1991). *Visual Behaviour as a Window for Reading the Mind of Others in Primates*. Oxford: Basil Blackwell.

Goodale, M., Milner, A. (1992). Separate visual pathways for perception and action. *Trends in Neurosciences*. 15 (1), 20–25.

Govan, C., Williams, K. (2004). Changing the affective valence of the stimulus items influences the IAT by re-defining the category. *Journal of Experimental Social Psychology*. *40*, 357–365.

Graf, P., & Schacter, D. (1985). Implicit and explicit memory for new associations in normal and amnesic subjects. *Journal of Experimental Psychology: Learning, Memory, and Cognition. 11*(3), 501-518.

Grandfield, T., Thompson, A., Turpin, G. (2005). An Attitudinal Study of Responses to a Range of Dermatological Conditions Using the Implicit Association Test. *Journal of Health Psychology*. *10*(6).
doi: 10.1177/1359105305057316.

Greenwald, A. (1992). New Look 3: Unconscious cognition reclaimed. *American Psychologist. 28*(3), 254-260.

Greenwald, A. (2001). What's Wrong with the Implicit Association Test? SESP Workshop, Spokane Washington. Retrieved from http://Faculty.washington.edu/agg/pdf/IAT.TopTenList.pdf

Greenwald, A. (2004). Revised Top Ten List of Things Wrong with the IAT. Attitudes Preconference SPSP - Austin, Texas January 29, 2004. Retrieved from http://faculty.washington.edu/agg/pdf/RevisedTop10.29Jan04.pdf

Greenwald, A., Banaji, M. (1995). Implicit Social Cognition: Attitudes, Self-Esteem, and Stereotypes. *Psychological Review*. *102*(1), 4-27.

Greenwald, A., Klinger, M., Schuh, E. (1995). Activation by Marginally Perceptible ("Subliminal") Stimuli: Dissociation of Unconscious From Conscious Cognition. *Journal of Experimental Psychology: General. 124*(1), 22-42.

Greenwald, A. McGee, D., Schwartz, J. (1998). Measuring Individual Differences in Implicit Cognition: The Implicit Association Test. *Journal of Personality and Social Psychology. 74*(6), 1464-1480.

Greenwald, A., Nosek, B. (2001). Health of the Implicit Association Test at Age 3. *Zeitschrift fUr Experimentelle Psychologie. 48*(2), 85-93.

Greenwald, A., Nosek, B. (2006). Attitudinal Dissociation (in Petty, R. E., Fazio, R. H., & Briñol, P. (*Eds*.) *Attitudes: Insights from the New Implicit Measures*. Hillsdale, NJ: Erlbaum.

Greenwald, A., Nosek, B., Banaji, M. (2003). Understanding and Using the Implicit Association Test: I. An Improved Scoring Algorithm. *Journal of Personality and Social Psychology. 85* (2), 197-216.
DOI: 10.1037/0022-3514.85.2.197

Greenwald, A., Gonzalez, R., Harris, R., Guthrie, D. (1997). Self-Knowledge and Self-Deception: Further Consideration. In Myslobodsky, M., (Ed.), *The Mythomanias: The Nature of Deception* (pp 51-62). New Jersey: Lawrence Erlbaum Associates.

Greschner, M., Shlens, J., Bakolitsa, C., Field, G., Gauthier, K., Jepson, L., Sher, A., (2010). Correlated firing among major ganglion cell types in primate retina. *The Journal of Physiology. 589*(1), 75-86. doi: 10.1113/jphysiol.2010.193888

Gunnar, M., Cheatham, C. (2003). Brain and Behavior Interface: Stress and the Developing Brain. *Infant Mental Health Journal. 24*(3), 195-211.
doi: 10.1002/imhj.10052

Haggard, P. (2005). Conscious intention and motor cognition. *Trends in Cognitive Sciences*. *9*(6), 290 -295. doi:10.1016/j.tics.2005.04.012

Haines, E., Summer, K. (2006). Implicit Measurement of Attitudes, Stereotypes, and Self-Concepts in Organizations Teaching Old Dogmas New Tricks. *Organizational Research Methods. 9*(4), 536-553.

Hart, A., Whalen, P., Shin, L., McInerney, S., Fischer, H., & Rauch, S. (2000). Differential response in the human amygdale to racial outgroup vs ingroup face stimuli. *NeuroReport. 22*, 2351-2355.

Hauk, O., Johnsrude, I., Pulvermüller, F. (2004). Somatotopic Representation of Action Words in Human Motor and Premotor Cortex. *Neuron. 41*(2), 301–307.

Haxby, J., Hoffman, E., Gobbini, M. (2000). The distributed human neural system for face perception. *Trends in Cognitive Sciences*. *4*(6), 223-233.

Hirai, M., Saunders, D., Troje, N. (2011). Allocation of attention to biological motion: Local motion dominates global shape. *Journal of Vision. 11*(3), 1–11. doi:10.1167/11.3.4.

Hochstein, S., Shapley, R. (1976). Quantitative Analysis of Retinal Ganglion Cell Classifications. *The Journal of Physiology. 262*, 237-264.

Hofmann, W., Friese, M., Strack, F. (2009). Impulse and Self-Control From a Dual-Systems Perspective. *Perspectives on Psychological Science. 4*(2), 162-176.

Hofmann W., Gawronski B., Gschwendner T., Le H., Schmitt M. (2005). A meta-analysis on the correlation between the implicit association test and explicit self-report measures. *Personality and Social Psychology Bulletin. 31*(10), 1369-1385. doi:10.1177/0146167205275613

Holmes, A., Winston, J., Eimer, M. (2005). The role of spatial frequency information for ERP components sensitive to faces and emotional facial. *Cognitive Brain Research. 25* (2), 508-520.

Hopfinger, J.B., Buonocore,M.H., Mangun,G.R. (2000). The neural mechanisms of top-down attentional control. *Nature Neuroscience. 3*, 284-291.

Howell, D. (1989). Fundamental Statistics for the Social Sciences. (2nd Ed.). Massachusetts: PWS-KENT Publishing.

Hubel, D., Livingstone, M. (1990). Color and Contrast Sensitivity in the Lateral Geniculate Body and Primary Visual Cortex of the Macaque. *The Journal of Neuroscience. 10* (7), 2223-2237.

Hubel, D., Wiesel, T. (1959). Receptive Fields of Single Neurones in the Cat's Striate Cortex. *The Journal of Physiology. 148*, 574-591.

Hubel, D., Wiesel, T. (1962). Receptive Fields, Binocular Interaction and Functional Architecture in the Cat's Visual Cortex. *The Journal of Physiology. 160*, 106-154.

Hume (2004). An Enquiry Concerning Human Understanding. Retrieved from http://etext.library.adelaide.

Humphrey, N. (1976). The Social Function of Intellect. In Bateson, P., Hinde, R. (Eds.), *Growing Points in Ethology*. Cambridge: Cambridge University Press.

Ingle D.J., Goodale M.A., Mansfield R.J.W. (2002). *Analysis of visual behavior*. Cambridge, MA: The MIT Press.

Inhelder, B., Piaget J. (1953). *The Growth of Logical Thinking from Childhood to Adolescence: An essay on the construction of formal operational structures.* Translated by A. Parsons and S. Milgam (1972). London: Routledge and Kegan Paul.

Inoue, K., Nadaoka, T., Oiji, A., Morioka, Y., Totsuka, S., Kanbayashi, Y. & Hukui, T. (1998). Clinical Evaluation of Attention-Deficit Hyperactivity Disorder by Objective Quantitative Measures. *Child Psychiatry and Human Development. 28*(3), 179-188.

Isoluminant. (2009). In *The Dictionary of Optometry and Visual Science* (*7th ed.*). Retrieved from http://medical-dictionary.thefreedictionary.com/isoluminant

Itti, L., Koch, C., Niebur, E. (1998). *A Model of Saliency-Based Visual Attention for Rapid Scene Analysis.* IEEE Transactions on Pattern Analysis and Machine Intelligence. *20* (11), 1254-1259.

Jacoby, R., Wiechmann, A., Amara, S., Leighton, B., Marshak, D. (2000). Diffuse bipolar Cells provide input to OFF parasol ganglion cells in the macaque retina. *The Journal of Comparative Neurolog*y. *416*(1), 6-18.

Jacoby, R., Marshak, D. (2000). Synaptic Connections of DB3 Diffuse Bipolar Cell Axons in Macaque Retina. *The Journal of Comparative Neurology*. *416*(1), 19 - 29.

Jacoby, R., Stafford, D., Kouyama, N. & Marshak, D. (1996). Synaptic Inputs to ON Parasol Ganglion Cells in the Primate Retina. *The Journal of Neuroscience. 16*(24), 8041–8056.

Jerison, H. (1973). Evolution of the brain and intelligence. New York: Academic Press.

Jetha, M., Zheng, X., Schmidt, L., Segalowitz, S. (2012). Shyness and the first 100 ms of emotional face processing. *Social Neuroscience. 7*(1), 74-89. doi: 10.1080/17470919.2011.581539

Jost, J., Nam, H., Amodio, D. Van Bavel, J. (2014). Political Neuroscience: The Beginning of a Beautiful Friendship. *Advances in Political Psychology. 35* (1), 3-42. doi: 10.1111/pops.12162

Kaplan, E., Shapley, R. (1982). X and Y cells in the lateral geniculate nucleus of macaque monkeys. *Journal of Physiology*. *330*, 125–143.

Kaplan, E., Shapley, R. (1986). The primate retina contains two types of ganglion cells, with high and low contrast sensitivity. *Proceedings of the National Academy of Sciences USA. 83*(8), 2755–2757.

Kelk, B. (2003). UK English Wordlist With Frequency Classification. Retrieved 23 November 2009 from www.bckelk.ukfsn.org.

Kirkpatrick, C. (2007). Tactical Deception and the Great Apes: Insight Into the Question of Theory of Mind. Totem: *The Berkeley Electronic Press. 15*(1), 4, 31-37.

Kitaoka, A. (2004). *On motion illusion in a stationary image*. Retrieved 30 March 2010 from http://www.psy.ritsumei.ac.jp/~akitaoka/TsukubaCOEsympo2005.html

Klistorner, A., Crewther, D., Crewther, S. (1997). Separate Magnocellular and Parvocellular Contributions from Temporal Analysis of the Multifocal VEP. *Vision Research. 37*(15), 2161-2169.

Kohler, C., Turner, T., Brensinger, C., Siegel, S., Kanes, S., Gur, R. E., Gur, R.C. (2003). Facial Emotion Recognition in Schizophrenia: Intensity Effects and Error Pattern. *American Journal of Psychiatry. 160*(10), 1768–1774.

Kok, A. (1999). Varieties of inhibition: manifestations in cognition, event-related potentials and ageing. *Acta Psychologica. 101*(2-3), 129 - 158.

Kolb, B., & Whishaw, I. (1996). *Fundamentals of human neuropsychology*. New York: W. H. Freeman & Company.

Kristjánsson, A., Nakayama, K. (2002). The attentional blink in space and time. *Vision Research. 42*(17), 2039–2050.

Kveraga, K. Ghuman, A, Bar, M. (2007). Top-down predictions in the cognitive brain. *Brain & Cognition. 65*, 145–168. doi:10.1016/j.bandc.2007.06.007

Kveraga, K., Boshyan, J., Bar, M. (2007b). Magnocellular Projections as the Trigger of Top-Down Facilitation in Recognition. *The Journal of Neuroscience. 27*(48), 13232–13240. doi:10.1523/JNEUROSCI.3481-07.2007

Laycock, R., Crewthera, S, Crewthera, D. (2007). A role for the 'magnocellular advantage' in visual impairments in neurodevelopmental and psychiatric. *Neuroscience and Biobehavioral Reviews. 31*, 363–376. doi:10.1016/j.neubiorev.2006.10.003

LeDoux J.E. (1996). The Emotional Brain. New York: Simon & Schuster.

LeDoux, J. (2012). Rethinking the Emotional Brain. *Neuron*. *73*, 653 - 676. Doi: 10.1016/j.neuron.2012.02.004

Lee, B. (2011). Visual pathways and psychophysical channels in the primate. *The Journal of Physiology. 589*(1), 41-47. doi:10.1113/jphysiol.2010.192658

Lee, B., Sun, H. (2009). The chromatic input to cells of the magnocellular pathway of primates. *Journal of Vision. 9*(2), 1–18.

Levine, J. (1983). Materialism and qualia: The explanatory gap. *Pacific Philosophical Quarterly. 64*, 354–361.

Lezak, M. D., Howieson, D.B., & Loring, D. W. (2004). *Neuropsychological Assessment* (4th *Ed*.). New York: Oxford University Press, Inc.

Lieberman, M, (2011). Why symbolic processing of affect can disrupt negative affect: Social cognitive and affective neuroscience investigations. In Todorov, A., Fiske, S. T., & Prentice, D (Eds.). *Social Neuroscience: Toward Understanding the Underpinnings of the Social Mind*. Oxford Scholarship Online. doi:10.1093/acprof:oso/9780195316872.001.0001

Lieberman, M. (2006). Social Cognitive Neuroscience: A Review of Core Processes. *Annual Review of Psychology*. *58*, 259–89.

Lieberman, M., Eisenberger, N., Crockett, M., Tom, S., Pfeifer, J., Way, B. (2007). Putting Feelings Into Words Affect Labeling Disrupts Amygdala Activity in Response to Affective Stimuli. *Association for Psychological Science. 18*(5), 421-428.

Lieberman, M., Hariri, A., Jarcho, J., Eisenberger, S., Bookheimer, S. (2005). An fMRI Investigation of race-related amygdala activity in African-American and Caucasian-American individuals. *Nature Neuroscience. 10*(1038), 1-3.

Lin, Z., He, S. (2009). Seeing the invisible: The scope and limits of unconscious processing in binocular rivalry. *Progress in Neurobiology*. *87*, 195–211.

Lindström, S., Wróbel, A. (2011). Feedforward and recurrent inhibitory receptive fields of principal cells in the cat's dorsal lateral geniculate nucleus. *European Journal of Physiology. 461*, 277–294.

Lisberger, S.G., Morris, E.J., Tychsen, L. (2001). Visual Motion Processing and Sensory-Motor Integration for Smooth Pursuit Eye Movements. *Annual Review of Neuroscience. 10*, 97-129.

Liu, C., Bryan, R., Miki, A., Woo, J., Liu, G., Elliott, M. (2006). Magnocellular and Parvocellular Visual Pathways Have Different Blood Oxygen Level–Dependent Signal Time Courses in Human Primary Visual Cortex. *American Journal of Neuroradiology*. *27*, 1628-1634.

Livingstone, M, Rosen, G., Drislane, F., Galaburda, A. (1991). Physiological and anatomical evidence for a magnocellular defect in developmental dyslexia. *Proceedings of the National Academy of Sciences USA. 88*, 7943-7947.

Livingstone, M. (1988). Art, Illusion and the Visual System. *Scientific American. 258*(1), 78-85.

Livingstone, M., Hubel, D. (2007b). Psychophysical evidence for separate channels for the perception of form, color, movement, and depth. *Journal of Neuroscience. 7*(11), 3416–3468.

Livingstone, M., Hubel, D. (1988). Segregation of Form, Color, Movement, and Depth: *Anatomy, Physiology, and Perception. Science. 240*(4853), 740-749. Retrieved from http://www.jstor.org/stable/1701543 on 05/08/2009 15:17.

Llinás, R., Ribary, U., Contreras, D., Pedroarena, C. (1998). The neuronal basis for consciousness. *Philosophical Transactions of The Royal Society Biological Sciences*. *353*(1377), 1841 - 1849.

Lu., Z., Lesmes, L., Sperling, G. (1999). The mechanism of isoluminant chromatic motion perception. *Proceedings of the National Academy of Science*. *96*(14), 8289-8294.

Maison, D., Greenwald, A., Bruin, R. (2001). The Implicit Association Test as a measure of implicit consumer attitudes. *Polish Psychological Bulletin. 32*(1), 1 - 9. doi://10.1066/S10012010002

Marshak, D., Yamada, E., Bordt, A., Perryman, W. (2002). Synaptic input to an ON parasol ganglion cell in the macaque retina: A serial section analysis. *Visual Neuroscience. 19* (3), 299-305.

Retrieved from http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3 on 7 June 2012

Martin P. (1998). Colour processing in the primate retina: recent progress. *The Journal of Physiology*. 513 (3), 631–638. doi: 10.1111/j.1469-7793.1998.631ba.x

Martin P., Lee B., White A., Solomon S., Rüttiger L. (2001). Chromatic sensitivity of ganglion cells in the peripheral primate retina. *Nature. 410*(6831), 933-936.

Maunsell, J., Ghose, J., Assad, J., Mcadams, C., Boudreau, C., Noerager, B. (1999). Visual response latencies of magnocellular and parvocellular LGN neurons in macaque monkeys. *Visual Neuroscience. 16*, 1 - 14.

Maunsell, J., Nealey, T., DePriest, D. (1990). Magnocellular and Parvocellular Contributions to Responses in the Middle Temporal Visual Area (MT) of the Macaque Monkey. *The Journal of Neuroscience. 10*(10), 3323-3334.

May C., Hasher L. (1998). Synchrony effects in inhibitory control over thought and action. *Journal of Experimental Psychology: Human Perception and Performance. 24*(2), 363–379.

Meltzoff, N., Moore, M. (1977). Imitation of Facial and Manual Gestures by Human Neonates. *Science. 198*(4312), 75-78.

Mermillod, M., Vuilleumier, P., Guyader, N., Alleysson, D., Marendaz, C. (2005). How Diagnostic are Spatial Frequencies for Fear Recognition? Paper presented at 4th Annual Summer Interdisciplinary Conference (ASIC 2005), Briançon: France (2005) Retrieved from http://csjarchive.cogsci.rpi.edu/proceedings/2005/docs/p1501.pdf on 18 September 2008, 06:07:18 PM.

Mishkin, M., Ungerleider, L., Macko, K. (1983). Object vision and spatial vision: two cortical pathways. *Trends in Neurosciences*. *6*, 414-417.

Morris, J., deBonis, M., Dolan, R. (2002). Human amygdala responses to fearful eyes. *NeuroImage. 17*(1), 214-222.

Munoz, D., Everling, S. (2004). Look Away: the Anti-saccade Task and the Voluntary Control of Eye Movement. *Nature Reviews Neuroscience. 5*(3), 218-228.

Mutalik, P. (2003). Editorial Review: Crick and Koch: A new "Framework for Consciousness". *Nature Neuroscience. 6* (2), 119 - 126.

Nagel, T. (1974). What is it like to be a Bat. *Philosophical Review. 83*, 435-456.

Nassi, J., Callaway, E (2009). Parallel processing strategies of the primate visual system. *Nature Reviews Neuroscience. 10*(10), 360-372.

Nealey, T., Maunsell, J. (1994). Magnocellular and Parvocellular Contributions to the Responses of Neurons in Macaque Striate Cortex. *The Journal of Neuroscience. 74*(4), 2069-2079.

Nier, J (2005). How Dissociated Are Implicit and Explicit Racial Attitudes? A Bogus Pipeline Approach. *Group Processes & Intergroup Relations*. *8*(1), 39-52. doi: 10.1177/1368430205048615

Nieuwenhuis, S., Jepma, M., La Fors, S., Olivers, C. (2008). The role of the magnocellular and parvocellular pathways in the attentional blink. *Brain and Cognition. 68*(1), 42–48. doi: 10.1016/j.bandc.2008.02.119

Nosek, B., Greenwald, A., Banaji, M. (2005). Understanding and Using the Implicit Association Test: II. Method Variables and Construct Validity. *Society for Personality and Social Psychology, Inc. 31* (2), 166-180. doi: 10.1177/0146167204271418

Omtzigt, D., Hendriks, A., Kolk, H. (2002). Evidence for magnocellular involvement in the identification of flanked letters. *Neuropsychologia. 40*(12), 1881–1890.

Orne, M. T. (1962). On the social psychology of the psychological experiment: With particular reference to demand characteristics. *American Psychologist*. *17*(11), 776-783.

Osterhout, L; Bersick, M.; McLaughlin, J. (1997). Brain Potentials reflect violations of Gender stereotypes. *Memory & Cognition. 25*(3), 273-285.

Otero-Millan, J., Macknik, S., & Martinez-Conde, S. (2012). Microsaccades and Blinks Trigger Illusory Rotation in the "Rotating Snakes" Illusion. *The Journal of Neuroscience.32* (17), 6043– 6051. doi:10.1523/JNEUROSCI.5823-11.2012

Pallet, P., Dobkins, K., (2013). Development of face discrimination abilities, and relationship to magnocellular pathway development. *Visual Neuroscience*. *30*, 251-262. doi:10.1017/S0952523813000217

Pessoa, L., Adolphs, R. (2010). Emotion processing and the amygdala: from a 'low road' to 'many roads' of evaluating biological significance. *Nature Reviews Neuroscience*. *11*, 773-782.

Petrusca, D., Grivich, M., Sher, A., Field, G., Gauthier, J., Greschner, M., Shlens, J., C (2007).Identification and Characterization of a Y-Like Primate Retinal Ganglion Cell Type. *The Journal of Neuroscience. 27*(41), 1019 –11027. doi: 10.1523/JNEUROSCI.2836-07.2007

Phelps, E. (2001). Faces and races in the brain. *Nature Neuroscience*. *4*(8), 775-776.

Phelps, E., LeDoux, J. (2005). Contributions of the Amygdala to Emotion Processing: From Animal Models to Human Behavior. *Neuron. 48*, 175-187. doi 10.1016/j.neuron.2005.09.025

Phelps, E., Thomas, L. (2004). Race, Behavior, and the Brain: The Role of Neuroimaging in Understanding Complex Social Behaviors. *Political Psychology. 24*(4), 747 - 758.

Piaget, J. (1947). *The Psychology of Intelligence*. New York: Harcourt Brace.

Piaget, J. (1953). *Logic and Psychology*. Manchester: Manchester University Press.

Piaget, J. (1967). *An Essay on the Relations Between Organic Regulations and Cognitive Processes*. Chicago: University of Chicago Press.

Poehlman, T., Uhlmann, E., Greenwald, A., Banaji, M. (ND). Understanding and Using the Implicit Association Test: III. Meta-analysis of Predictive Validity. Unpublished Draft (30 Oct 2008).

Poehlman, T., Uhlmann, E., Greenwald, A., Banaji, M. (2009). Understanding and Using the Implicit Association Test: III. Meta-analysis of Predictive Validity. *Journal of Personality and Social Psychology. 97*(1), 17-41.

Purpura, K. P., Schiff, N. D. (1997). The thalamic intralaminar nuclei: a role in visual awareness. *Neuroscientist. 3*, 8-15.

Ratner, K., Dotsh, R., Wigboldus, D., van Knippenberg, A., Amodio, D. (2014). Visualizing Minimal Ingroup and Outgroup Faces. *Journal of Personality and Social Psychology*. 106 (6), 897-911. DOI: 10.1037/a0036498

Ravall (2012). Lateral Inhibition, Vision and Optical Illusion's Explained. Retrieved from http://www.ravall.com/2011/05/30/lateral-inhibition-optical-illusions-explained/ on 02/09/2011.

Rizk-Jackson, A., Acevedo, S., Inman, D., Howieson, D., Benice, T., Raber, K. (2006). Effects of sex on object recognition and spatial navigation in humans. *Behavioural Brain Research. 173*, 181–190. doi: 10.1016/j.bbr.2006.06.029

Rizzolatti, G. (2005). The mirror neuron system and its function in humans. *Anatomy and embryology*. 210 (5-6), 419–421.

Rizzolatti, G., Fadiga, L., Gallese, V., Fogassi, L. (1996). Premotor cortex and the recognition of motor actions. *Cognitive Brain Research. 3*(2), 131–141.

Rowe, M. (2002). Trichromatic Color Vision in Primates. *Physiology*. *17*(3), 93-98.

Satpute, A., Lieberman. M. (2006). Integrating automatic and controlled processes into neurocognitive models of social cognition. *Brain Research. 1079*, 86 -97.

Schacter, D. (1987). Implicit Memory: History and Current Status. *Journal of Experimental Psychology. 13*(3), 501-518.

Schacter, D. L., & Graf, P. (1986a). Effects of elaborative processing on implicit and explicit memory for new associations. *Journal of Experimental Psychology: Learning, Memory, and Cognition. 12*(3), 432-444.

Schmid, M.C., Mrowka, S.W., Turchi, J., Saunders, R.C., Wilke, M., Peters, A.J. Ye, F.Q., (2010). Blindsight depends on the lateral geniculate nucleus. *Nature. 466*, 373–377. doi:10.1038/nature09179
Retrieved 2010/07/12 from http://www.nature.com/nature/journal/vaop/ncurrent

Segalowitz, S., Zheng, X. (2009). An ERP study of category priming: Evidence of early lexical semantic access. *Biological Psychology. 80*(1), 122-129. doi: 10.1016/j.biopsycho.2008.04.009.

Shapiro, K., Arnell, K., Raymond, J. (1997). The Attentional Blink. *Trends in Cognitive Science. 1*(8), 291-296.

Shapley, R., Kaplan, E., Soodak, R, (1981). Spatial summation and contrast sensitivity of X And Y cells in the lateral geniculate nucleus of the. *Nature. 292*, 543-545. doi:10.1038/292543a0

Shiffrin, R. M., & Schneider, W. (1977). Controlled and automatic human information processing: II. Perceptual learning, automatic attending. *Psychological Review. 84*(2), 127-190. doi: 10.1037/0033-295X.84.2.127

Simmons A. (2014). Factors Affecting PC Monitor Responsiveness. Retrieved 15/09/2014 from https://pcmonitors.info/articles/factors-affecting-pc-monitor-responsiveness/.

Singer, W. (1993). Synchronization of cortical activity and its putative role in information processing and learning. *Annual Review of Physiology. 55*, 349–374.

Skottun, B. (2005). Magnocellular reading and dyslexia. *Vision Research. 45*(1), 133–134.

Solomon, S., Lee, B., White, A., Rüttiger L.., Martin, P. (2005). Chromatic Organization of Ganglion Cell Receptive Fields in the Peripheral Retina. *The Journal of Neuroscience. 25* (18), 4527– 4539. doi: 10.1523/JNEUROSCI.3921-04.2005

Spering, M., Gegenfurtner, K. (2008). Contextual effects on motion perception and smooth pursuit eye movement. *Brain Research. 1225*, 76–85. doi:10.1016/j.brainres.2008.04.061

Sriram, N., Greenwald, A. (2009). The Brief Implicit Association Test. *Experimental Psychology. 56*(4), 283-294.

Starn, H. (2007). Disambiguation, Binding, and the Unity of Visual Consciousness. *Theory & Psychology. 17*(6), 747-777. doi: 10.1177/0959354307083492

Stein, J. (2001). The Magnocellular Theory of Developmental Dyslexia. *Dyslexia. 7*(1), 12-36. doi: 10.1002/dys.186

Steinman, B., Steinman, S., Lehmkuhle, S. (1996). Transient Visual Attention is Dominated by the Magnocellular Stream. *Vision Research. 37*(1), 17-23.

Stoesz, B., Jakobson, L. (2013). A sex difference in interference between identity and expression judgments with static but not dynamic faces. *Journal of Vision. 13* (5), 1-14. doi: 10.1167/13.5.26

Strawson, P. (1959). Individuals: an essay in Descriptive Metaphysics. New York: Routledge.

Tabibnia, G., Lieberman, M., Craske, M. (2008). The Lasting Effect of Words on Feelings: Words May Facilitate Exposure Effects to Threatening Images. *Emotion*. *8*(3), 307–317.

Tamietto, M., Cauda, F., Corazzini, L., Savazzi, S., Marzi, C., Goebel, R., Weiskrantz, L. (2009). Collicular Vision Guides Nonconscious Behavior. *Journal of Cognitive Neuroscience. 22* (5), 888-902.

Todorov, A., Fiske, S., Prentice, D. (2011). Social Neuroscience: Toward Understanding the Underpinnings of the Social Mind. *Oxford Scholarship Online*. doi:10.1093/acprof:oso/9780195316872.001.000

Tononi, G., Edelman, G. (1998). Consciousness and Complexity. *Science. 282*, 1846-1851.

Trevarthen, C., Reddy, V. In M. Velman & S. Schneider (2006). *Consciousness in infants.* Oxford: Blackwells.

Ungerleider, L., Mishkin, M. (1982). *Two cortical visual systems*. In D. J. Ingle, M.A. Goodale, and R. J. W. Mansfield, (Eds.). Cambridge, MA: MIT Press.

Uno, H., Tarara, R., Else, J.G., Suleman, M.A., Sapolsky, R.M. (1989). Hippocampal damage associated with prolonged and fatal stress in primates. *The Journal of Neuroscience. 9* (5), 1705-1711.

Vedantam, S. (Sunday Jan 23 2005; Page W12). See no Bias. *Washington Post*.

Verwey, C., Quayle, M. (2012). Whiteness, Racism, and Afrikaner Identity in Post-apartheid South Africa. *African Affairs*. *111*(445), 551-575.

Vizioli, L., Foreman, K., Rousselet, G., Caldara, R. (2010). Inverting faces elicits sensitivity to race on the N170 component: A cross-cultural study. *Journal of Vision*. 10 (15), 1-23. doi: 10.1167/10.1.15

Vohs, K., Baumeister, R., Cirarocco, N. (2005). Self-Regulation and Self-Presentation: Regulatory Resource Depletion Impairs Impression Management and Effortful Self-Presentation. *Journal of Personality and Social Psychology*. 88 (4), 632-657. doi: 10.1037/0022-3514.88.4.632

Vuilleumier, P. (2005). How brains beware: neural mechanisms of emotional attention. *Trends in Cognitive Sciences. 9*(12), 585-594.

Vuilleumier, P., Armony, J., Dolan, R. (2003). Distinct spatial frequency sensitivities for processing faces and emotional expressions. *Nature Neuroscience. 6*(6), 624 - 631.

Weiskrantz, L. (1986). Blindsight: *A Case Study and Implications*. Oxford: Oxford University Press.

White, B., Boehnke, E., Marino, R., Itti, L. & Munoz, D., (2009). Color-Related Signals in The Primate Superior Colliculus. *The Journal of Neuroscience. 29*(30), 12159 – 12166. doi:10.1523/JNEUROSCI.1986-09.2009

Whiten, A. (Ed.) (1991). *Natural Theories of Mind*. Oxford: Basil Blackwell.

Whiten, A., Perner, J. (1991). Fundamental issues in the Multidisciplinary study of Mindreading. In Whiten, A. (Ed.) *Natural Theories of Mind*, (pp 1-17). Oxford: Basil Blackwell.

Williams, M., Morris, A., McGlone, F., Abbott, D., Mattingley, J. (2004). Amygdala responses to fearful and happy facial expressions under conditions of binocular suppression. *Journal of Neuroscience. 24*, 2898– 2904.

Wilson, R., Keil, F. (Eds.) (1999). *The MIT Encyclopedia of the Cognitive Sciences*. Cambridge, Massachusetts: The MIT Press.

Wilson, T., Bar-Anan, Y. (2008). The Unseen Mind. *Science*. *321*, 1046 - 1047. doi: 10.1126/science.1163029

Wittenbrink, B., Judd, C., Park, B. (1997). Evidence for Racial Prejudice at the Implicit Level and Its Relationship With Questionnaire Measures. *Journal of Personality and Social Psychology*. *72*(2), 262 - 274.

Wood, J. (2003). Social Cognition and the Prefrontal Cortex. Behavioral and Cognitive *Neuroscience Reviews. 2*(2), 97-114. doi: 10.1177/1534582303002002002

Worden, R.P. (1996). Primate Social Intelligence. *Cognitive Science. 20*(4), 579-616.

Yang, P., Chung, L., Chen, C. & Chen, C (2004). Rapid improvement in academic grades following methylphenidate treatment in attention-deficit hyperactivity. *Psychiatry and Clinical Neurosciences. 58*, 37–41.

Yin, H., Knowlton, B. (2006). The role of the basal ganglia in habit formation. *Nature Reviews Neuroscience*. *7* (June 2006), 464-476. doi:10.1038/nrn1919

Zeki, S. (2003). The disunity of consciousness. *Trends in Cognitive Sciences. 7*(5), 214-218. doi:10.1016/S1364-6613(03)00081-0

Appendix 1

See media disk for this appendix. This appendix is in a sub-folder named ‘Appendix 1’ under ‘Appendices Applications and materials’ on the media disk and in the Dropbox folder. The document is entitled ‘Appendix 1.pdf’. Alternatively, this material may be accessed from a Dropbox location which will be shared on request to Prof. Colin Tredoux (colin.tredoux@uct.ac.za).

Appendix 2

See media disk for this appendix. This appendix is in a .pdf document in a sub-folder named ‘Appendix 2’ under ‘Appendices Applications and materials’ on the media disk and in the Dropbox folder. This material may be accessed from a Dropbox location which will be shared on request to Prof. Colin Tredoux (colin.tredoux@uct.ac.za).

Appendix 3

Delphi code for calibration procedure (Experiment 2)

```
Procedure Calibrate(var Timer2: TTimer; Gc: Double; SC: Double; Va: Double);
Begin
  if (Counter > 1) and (Pract[GlobalCounter].Value = 0) then
    if (((Sc / Va) > 0.75) and (Timer2.Interval > 30)) then
      Timer2.Interval := Timer2.Interval - 5;
  if ((Counter < 2) and (Pract[GlobalCounter].Value = 0) and (Valid > 20)) then
    If Timer2.Interval<300 Then Timer2.Interval := Timer2.Interval + 10;
  if (Counter > 1) and (Pract[GlobalCounter].Value = 0) then
    if (Sc / Va < 0.70) then
      if (Timer2.Interval < 150) then
        Timer2.Interval := Timer2.Interval + 5;
end;
```

Delphi code to display practice targets and call calibration procedure in Replication Experiment

```
// DISPLAY THE PRACTICE TARGETS
Procedure DisplayP(VAR Letter: String;
Field,Mask,Validnumber,errornumber,RTime,ETime :Tlabel; Latency,ELatency:
TAdvSmoothLedLabel; Timer1,Timer2: Ttimer;  VAR Last: String; KeyCounter : TLabel;
BtnStartTest,BtnChoose,BtnBeginPractice:   TadvSmoothButton;  ProgressBar1 :
TAdvSmoothProgressBar; Time2: TLabel; Bevel1: TadvSmoothPanel;  CaptionTargets,
CaptionError, CaptionScore : TLabel);
Var Va,SC,Gc : Double;
BEGIN
  Mask.Visible:=False;
  GlobalCounter:=GlobalCounter+1;
  Progressbar1.Next;
  Va:=Valid;
  Sc:=Counter;
  Gc:=GlobalCounter;
 If Cal then Calibrate(Timer2, Gc, SC, Va);
  Time2.Caption:=IntToStr(Timer2.Interval);
  If Timer2.Interval<LowestValue then LowestValue:=Timer2.Interval;
  IF GlobalCounter<=(PracticeT) THEN
   BEGIN
    If Pract[GlobalCounter].Value=0 Then
      Begin
        Valid:=Valid+1;
        Validnumber.Caption:=IntToStr(valid);
      End;
      If Pract[GlobalCounter].Value=1 Then
        Begin
          PracticeNonTargets:=PracticeNonTargets+1;
        End;
      Pract[GlobalCounter].Timing:=Timer2.Interval;
      Field.Caption:=Pract[GlobalCounter].Word;
      Ts:=DateTimeToTimeStamp(now);
      Pressed:=False;
      StartClock;
      Timer2.Enabled:=true;
    end;
IF GLobalCounter>PracticeT Then
  Begin
   EndupTime;
   Timer1.Enabled:=false;
   RTime.Caption:='Mean T.L. ';
   ETime.Caption:='Mean E.L. ';
   IF (Counter>0) Then P1Mean:=(P1Tot DIV Counter);
   If (Errors>0) Then P1EMean:=(P1ETot DIV Errors);
   Latency.Caption.Value:=P1Mean;
   ELatency.Caption.Value:=P1EMean;
```

```
   Pract[GlobalCounter].Timing:=Timer2.Interval;
   FileFunctions.EndPractice(PracticeD);
```

Delphi code to implement calibration values for non-practice phases

```
Procedure TForm1.StartAllTesting(Sender: TObject);
begin
  Progressbar1.Visible:=True;
  ProgressBar1.Minimum:=0;
  If (Phase = 'P') Then
    Begin
      LowestValue:=T2;
      FileFunctions.ChecklistP;
      ProgressBar1.Maximum:=PracticeT;
      ProgressBar1.Step:=1;
      Timer1.Interval:=T1; //900
      Timer2.Interval:=T2;  // 200 actual stimulus exposure time
      Timer3.Interval:=T3;  // 300 mask exposure time
    End;
  If (Phase <>'P') Then
  Begin
    Progressbar1.Maximum:=MainT;
    If Overide Then T2:=Sub
      Else T2:=50+(LowestValue DIV 10);
    Timer1.Interval:=T1;
    Timer2.Interval:=T2;
    Timer3.Interval:=T3;
  End;
PanelOff(Bevel1,CaptionTargets,CaptionError,CaptionScore,RTime,ETime,ValidNumber,Er
rorNumber,KeyCounter,Latency,ELatency);
  GlobalCounter:=0;
  Arraylength:=LSensibleW+LNonWord;
  Setlength(Both,ArrayLength+1);
  Sorting.SortAll(LSensibleW,LNonWord,SensibleW,NonW,Both);
  ProgressBar1.Position:=0;
  Mask.Caption:=MaskCh;
  Banner1.Caption:='Testing... PRESS "S" to end';
  Field.Visible:=true;
  Field.Caption:='X';
  If Phase='M' Then
    Begin
      FileFunctions.Checklist1;
    End;
  If Phase='M2' Then
    Begin
      FileFunctions.Checklist2;
    End;
  StartupTime;
  Timer1.Enabled:=true;
  BtnChoose.Visible:=false;
```

```
  Form1.KeyPreview:=true;
end;
```

Delphi code to select isoluminance colour pairs and compute values

```
Procedure TForm1.CalcPAverages;
Var Ix,Highest,SolutionIdx,LocalIx,SolutionDx : Integer;
    SL1: Array[1..5] Of Integer;
    SL2: Array[1..5] Of Integer;
    SL3: Array[1..5] Of Integer;
    SL4: Array[1..5] Of Integer;
    SL5: Array[1..5] Of Integer;
    C1R : Array Of Integer;
    C1G : Array Of Integer;
    C1B : Array Of Integer;
    C2R : Array Of Integer;
    C2G : Array Of Integer;
    C2B : Array Of Integer;
    Overall : Array[1..5] Of Integer;
Begin
  Solution1H:=0;
  Solution1Av:=0;
  Solution2H:=0;
  Solution2Av:=0;
  Solution3H:=0;
  Solution3Av:=0;
  Solution4H:=0;
  Solution4Av:=0;
  Solution5H:=0;
  Solution5Av:=0;
  FinalSolution:=0;
  SL1[1]:=StrToInt(T20S1.Caption);
  SL1[2]:=StrToInt(T18S1.Caption);
  SL1[3]:=StrToInt(T16S1.Caption);
  SL1[4]:=StrToInt(T14S1.Caption);
  SL1[5]:=StrToInt(T12S1.Caption);
  Solution1H:=MaxIntValue(SL1);
    if Solution1H>0 then
    Begin
      if Solution1H >=5 then
        Solution1Av:=(SumInt(SL1) Div 5);
      Solution1T:=SumInt(SL1);
      Overall[1]:=Solution1T;
    End;

  SL2[1]:=StrToInt(T20S2.Caption);
  SL2[2]:=StrToInt(T18S2.Caption);
  SL2[3]:=StrToInt(T16S2.Caption);
  SL2[4]:=StrToInt(T14S2.Caption);
  SL2[5]:=StrToInt(T12S2.Caption);
```

```
Solution2H:=MaxIntValue(SL2);
if Solution2H>0 then
  Begin
    if Solution2H >=5 then
      Solution2Av:=(SumInt(SL2) Div 5);
    Solution2T:=SumInt(SL2);
    Overall[2]:=Solution2T;
  End;

SL3[1]:=StrToInt(T20S3.Caption);
SL3[2]:=StrToInt(T18S3.Caption);
SL3[3]:=StrToInt(T16S3.Caption);
SL3[4]:=StrToInt(T14S3.Caption);
SL3[5]:=StrToInt(T12S3.Caption);
Solution3H:=MaxIntValue(SL3);
if Solution3H>0 then
  Begin
    if Solution3H>=5 then
      Solution3Av:=(SumInt(SL3) Div 5);
    Solution3T:=SumInt(SL3);
    Overall[3]:=Solution3T;
  End;

SL4[1]:=StrToInt(T20S4.Caption);
SL4[2]:=StrToInt(T18S4.Caption);
SL4[3]:=StrToInt(T16S4.Caption);
SL4[4]:=StrToInt(T14S4.Caption);
SL4[5]:=StrToInt(T12S4.Caption);
Solution4H:=MaxIntValue(SL4);
if SOlution4H>0 then
  Begin
    if Solution4H>=5 then
      Solution4Av:=(SumInt(SL4) Div 5);
    Solution4T:=SumInt(SL4);
  Overall[4]:=Solution4T;
  End;

SL5[1]:=StrToInt(T20S5.Caption);
SL5[2]:=StrToInt(T18S5.Caption);
SL5[3]:=StrToInt(T16S5.Caption);
SL5[4]:=StrToInt(T14S5.Caption);
SL5[5]:=StrToInt(T12S5.Caption);
Solution5H:=MaxIntValue(SL5);
if SOlution5H>0  then
  Begin
    if Solution5H>=5 then
      Solution5Av:=(SumInt(SL5) Div 5);
    Solution5T:=SumInt(SL5);
    Overall[5]:=Solution5T;
  End;
```

```
if (MaxIntValue(Overall)>0)  then
Begin
  if Overall[1]=MaxIntValue(Overall) then FinalSolution:=1;
  if Overall[2]=MaxIntValue(Overall) then FinalSolution:=2;
  if Overall[3]=MaxIntValue(Overall) then FinalSolution:=3;
  if Overall[4]=MaxIntValue(Overall) then FinalSolution:=4;
  if Overall[5]=MaxIntValue(Overall) then FinalSolution:=5;
  if FinalSolution=1 then Highest:=Solution1H;
  if FinalSolution=2 then Highest:=Solution2H;
  if FinalSolution=3 then Highest:=Solution3H;
  if FinalSolution=4 then Highest:=Solution4H;
  if FinalSolution=5 then Highest:=Solution5H;
End
Else
  FinalSolution:=0;

ChkS1.Visible:=true;
ChkS2.Visible:=true;
ChkS3.Visible:=true;
ChkS4.Visible:=true;
ChkS5.Visible:=true;

if FinalSolution=1 then ChkS1.Checked:=true;
if FinalSolution=2 then ChkS2.Checked:=true;
if FinalSolution=3 then ChkS3.Checked:=true;
if FinalSolution=4 then ChkS4.Checked:=true;
if FinalSolution=5 then ChkS5.Checked:=true;

 //Calculates all colour solutions
for SolutionDx := 1 to 5 do
  Begin
    SolutionIdx:=0;
    for Ix := 1 to PDataIndx do
      Begin
        if PCalData[Ix].ColourSolution=SolutionDx then
          Begin
            SolutionIdx:=SolutionIdx+1;
          End;
      End;
      if SolutionIdx>0 then
      Begin
        SetLength(C1R,SolutionIdx);
        SetLength(C1G,SolutionIdx);
        SetLength(C1B,SolutionIdx);
        SetLength(C2R,SolutionIdx);
        SetLength(C2G,SolutionIdx);
        SetLength(C2B,SolutionIdx);
        LocalIx:=0;
        for Ix := 1 to PDataIndx do
```

```
    Begin
      if PCalData[Ix].ColourSolution=SolutionDx then
        Begin
          Inc(LocalIx);
          C1R[LocalIx-1]:=PCalData[Ix].C1R;
          C1G[LocalIx-1]:=PCalData[Ix].C1G;
          C1B[LocalIx-1]:=PCalData[Ix].C1B;
          C2R[LocalIx-1]:=PCalData[Ix].C2R;
          C2G[LocalIx-1]:=PCalData[Ix].C2G;
          C2B[LocalIx-1]:=PCalData[Ix].C2B;
        End;
    End;
// background destination colour
    SOl1[SolutionDx].SR1:=(SumInt(C1R) Div SolutionIdx);   //FDR
    SOl1[SolutionDx].SG1:=(SumInt(C1G) Div SolutionIdx);   //FDG
    SOl1[SolutionDx].SB1:=(SumInt(C1B) Div SolutionIdx);   //FDB

    //ForeGround destination colour
    SOl2[SolutionDx].SR2:=(SumInt(C2R) Div SolutionIdx);    //DR
    SOl2[SolutionDx].SG2:=(SumInt(C2G) Div SolutionIdx);    //DG
    SOl2[SolutionDx].SB2:=(SumInt(C2B) Div SolutionIdx);    //DB
End
Else
  Begin
    if SolutionDx=1 then
      Begin
        SOl1[SolutionDx].SR1:=S1C1R;
        SOl1[SolutionDx].SG1:=S1C1G;
        SOl1[SolutionDx].SB1:=S1C1B;

        SOl2[SolutionDx].SR2:=S1C2R;
        SOl2[SolutionDx].SG2:=S1C2G;
        SOl2[SolutionDx].SB2:=S1C2B;
      End;
    if SolutionDx=2 then
      Begin
        SOl1[SolutionDx].SR1:=S2C1R;
        SOl1[SolutionDx].SG1:=S2C1G;
        SOl1[SolutionDx].SB1:=S2C1B;

        SOl2[SolutionDx].SR2:=S2C2R;
        SOl2[SolutionDx].SG2:=S2C2G;
        SOl2[SolutionDx].SB2:=S2C2B;
      End;
    if SolutionDx=3 then
      Begin
        SOl1[SolutionDx].SR1:=S3C1R;
        SOl1[SolutionDx].SG1:=S3C1G;
        SOl1[SolutionDx].SB1:=S3C1B;
```

```
            SOl2[SolutionDx].SR2:=S3C2R;
            SOl2[SolutionDx].SG2:=S3C2G;
            SOl2[SolutionDx].SB2:=S3C2B;
          End;
        if SolutionDx=4 then
          Begin
            SOl1[SolutionDx].SR1:=S4C1R;
            SOl1[SolutionDx].SG1:=S4C1G;
            SOl1[SolutionDx].SB1:=S4C1B;

            SOl2[SolutionDx].SR2:=S4C2R;
            SOl2[SolutionDx].SG2:=S4C2G;
            SOl2[SolutionDx].SB2:=S4C2B;
          End;
        if SolutionDx=5 then
          Begin
            SOl1[SolutionDx].SR1:=S5C1R;
            SOl1[SolutionDx].SG1:=S5C1G;
            SOl1[SolutionDx].SB1:=S5C1B;

            SOl2[SolutionDx].SR2:=S5C2R;
            SOl2[SolutionDx].SG2:=S5C2G;
            SOl2[SolutionDx].SB2:=S5C2B;
          End;
      End;
  End;



SolutionIdx:=0;
 for Ix := 1 to PDataIndx do
 Begin
  if PCalData[Ix].ColourSolution=FinalSolution then
  Begin
    SolutionIdx:=SolutionIdx+1;
  End;
 End;

 if SolutionIdx>0 then
 Begin
  SetLength(C1R,SolutionIdx);
  SetLength(C1G,SolutionIdx);
  SetLength(C1B,SolutionIdx);
  SetLength(C2R,SolutionIdx);
  SetLength(C2G,SolutionIdx);
  SetLength(C2B,SolutionIdx);
 End;

 //Calculate averages by putting the FinalSolution values into the temporary array and
// calculates
```

```
LocalIx:=0;
for Ix := 1 to PDataIndx do
Begin
 if PCalData[Ix].ColourSolution=FinalSolution then
 Begin
   Inc(LocalIx);
   C1R[LocalIx-1]:=PCalData[Ix].C1R;
   C1G[LocalIx-1]:=PCalData[Ix].C1G;
   C1B[LocalIx-1]:=PCalData[Ix].C1B;
   C2R[LocalIx-1]:=PCalData[Ix].C2R;
   C2G[LocalIx-1]:=PCalData[Ix].C2G;
   C2B[LocalIx-1]:=PCalData[Ix].C2B;
 End;
End;

 // background from Colour  <=
 SR:=MSR;  //122;
 SG:=MSG;  //122;
 SB:=MSB;  //122;

 if FinalSolution =0 then
   Prescribed:=True;

 if FinalSolution<>0 then
 Begin
   // background destination colour
   Cal1R:=(SumInt(C1R) Div SolutionIdx);   //FDR
   Cal1G:=(SumInt(C1G) Div SolutionIdx);   //FDG
   Cal1B:=(SumInt(C1B) Div SolutionIdx);   //FDB

   //ForeGround destination colour
   Cal2R:=(SumInt(C2R) Div SolutionIdx);    //DR
   Cal2G:=(SumInt(C2G) Div SolutionIdx);    //DG
   Cal2B:=(SumInt(C2B) Div SolutionIdx);    //DB
 End;

 if  Prescribed then
 Begin
   DR:=PC1R;
   DG:=PC1G;
   DB:=PC1B;
   FDR:=PC2R;
   FDG:=PC2G;
   FDB:=PC2B;
 End
   Else
     Begin
      // background destination colour
      DR:=(SumInt(C1R) Div SolutionIdx);   //FDR
      DG:=(SumInt(C1G) Div SolutionIdx);   //FDG
```

```
      DB:=(SumInt(C1B) Div SolutionIdx);   //FDB

      //ForeGround destination colour
      FDR:=(SumInt(C2R) Div SolutionIdx);    //DR
      FDG:=(SumInt(C2G) Div SolutionIdx);    //DG
      FDB:=(SumInt(C2B) Div SolutionIdx);    //DB
    End;

//Foreground threshold Colou >=
FSR:=MFSR;  //122;
FSG:=MFSG;  //122;
FSB:=MFSB; //122;

If UserCal then
  ConfirmColour
  Else
    ProcessParvoPics;
  //update the summary table
 ADOSummaryParvo.TableName:='ParvoCalibration';
 ADOSummaryParvo.Open;
 ADOSummaryParvo.Active:=True;
 With ADOSummaryParvo Do
 Begin
   Append;
   Fields.FieldByName('ID').Value:=Login.VNumber;
   Fields.FieldByName('Solution 1Hz').Value:=(1000 Div(Ptiming[1]));
   Fields.FieldByName('Solution 2Hz').Value:=(1000 Div(Ptiming[2]));
   Fields.FieldByName('Solution 3Hz').Value:=(1000 Div(Ptiming[3]));
   Fields.FieldByName('Solution 4Hz').Value:=(1000 Div(Ptiming[4]));
   Fields.FieldByName('Solution 5Hz').Value:=(1000 Div(Ptiming[5]));

   Fields.FieldByName('S1 Hz1 F').Value:=SL1[1];
   Fields.FieldByName('S1 Hz2 F').Value:=SL1[2];
   Fields.FieldByName('S1 Hz3 F').Value:=SL1[3];
   Fields.FieldByName('S1 Hz4 F').Value:=SL1[4];
   Fields.FieldByName('S1 Hz5 F').Value:=SL1[5];

   Fields.FieldByName('S2 Hz1 F').Value:=SL2[1];
   Fields.FieldByName('S2 Hz2 F').Value:=SL2[2];
   Fields.FieldByName('S2 Hz3 F').Value:=SL2[3];
   Fields.FieldByName('S2 Hz4 F').Value:=SL2[4];
   Fields.FieldByName('S2 Hz5 F').Value:=SL2[5];

   Fields.FieldByName('S3 Hz1 F').Value:=SL3[1];
   Fields.FieldByName('S3 Hz2 F').Value:=SL3[2];
   Fields.FieldByName('S3 Hz3 F').Value:=SL3[3];
   Fields.FieldByName('S3 Hz4 F').Value:=SL3[4];
   Fields.FieldByName('S3 Hz5 F').Value:=SL3[5];

   Fields.FieldByName('S4 Hz1 F').Value:=SL4[1];
```

```
    Fields.FieldByName('S4 Hz2 F').Value:=SL4[2];
    Fields.FieldByName('S4 Hz3 F').Value:=SL4[3];
    Fields.FieldByName('S4 Hz4 F').Value:=SL4[4];
    Fields.FieldByName('S4 Hz5 F').Value:=SL4[5];

    Fields.FieldByName('S5 Hz1 F').Value:=SL5[1];
    Fields.FieldByName('S5 Hz2 F').Value:=SL5[2];
    Fields.FieldByName('S5 Hz3 F').Value:=SL5[3];
    Fields.FieldByName('S5 Hz4 F').Value:=SL5[4];
    Fields.FieldByName('S5 Hz5 F').Value:=SL5[5];

    Fields.FieldByName('S1 High').Value:=Solution1H;
    Fields.FieldByName('S1 Avg').Value:=Solution1Av;
    Fields.FieldByName('S2 High').Value:=Solution2H;
    Fields.FieldByName('S2 Avg').Value:=Solution2Av;
    Fields.FieldByName('S3 High').Value:=Solution3H;
    Fields.FieldByName('S3 Avg').Value:=Solution3Av;
    Fields.FieldByName('S4 High').Value:=Solution4H;
    Fields.FieldByName('S4 Avg').Value:=Solution4Av;
    Fields.FieldByName('S5 High').Value:=Solution5H;
    Fields.FieldByName('S5 Avg').Value:=Solution5Av;
    Fields.FieldByName('Final Solution').Value:=FinalSolution;
    Fields.FieldByName('BG R').Value:=DR;
    Fields.FieldByName('BG G').Value:=DG;
    Fields.FieldByName('BG B').Value:=DB;
    Fields.FieldByName('FG R').Value:=FDR;
    Fields.FieldByName('FG G').Value:=FDG;
    Fields.FieldByName('FG B').Value:=FDB;
    if UserCal then
      Fields.FieldByName('User Calibration').Value:=1
        Else
          Fields.FieldByName('User Calibration').Value:=0;
    if Prescribed then
      Fields.FieldByName('Prescribed Solution').Value:=1
        Else
          Fields.FieldByName('Prescribed Solution').Value:=0;
    if Bypass then
      Fields.FieldByName('Bypass Calibration').Value:=1
        Else
          Fields.FieldByName('Bypass Calibration').Value:=0;
    Fields.FieldByName('PC1R').Value:=PC1R;
    Fields.FieldByName('PC1G').Value:=PC1G;
    Fields.FieldByName('PC1B').Value:=PC1B;
    Fields.FieldByName('PC2R').Value:=PC2R;
    Fields.FieldByName('PC2G').Value:=PC2G;
    Fields.FieldByName('PC2B').Value:=PC2B;
    UpdateRecord;
    Post;
  End;
End;
```

Delphi procedures to extract and randomly distribute picture files between different conditions

```
procedure ExtractPFiles;
Var
  ReturnCode :  Cardinal;
  Path       :  STRING;
  FileSpec   :  STRING;
  Line,Str   :  STRING;
  SearchRec  :  TSearchRec;
  IDX,NullS,NullL : Integer;
  FileList : TStringList;
  Smaller : TStringList;
  Bigger : TStringList;
begin
  FileList:=TStringList.Create;
  Smaller:=TStringList.Create;
  Bigger:=TStringList.Create;
FileSpec := '*.bmp';
Path:='C:\CPT\Files\ShoeBox\LumPics\';
ReturnCode := SysUtils.FindFirst(Path + Filespec, faAnyFile, SearchRec);
  WHILE ReturnCode = 0 DO
    Begin
      Line := SearchRec.Name;
      FileList.Add(Line);
      ReturnCode := SysUtils.FindNext(SearchRec);
    End;
   for IDX := 0 to FileList.Count - 1 do
  Begin
    Line:=FileList[IDX];
    Str:=MidStr(Line,1,1);
    if Str='S' then
    Begin
      Line:=AnsiReplaceText(Line,'.bmp','');
      Smaller.Add(Line);
    End;
    if Str='L' then
      Begin
      Line:=AnsiReplaceText(Line,'.bmp','');
      Bigger.Add(Line);
      End;
  End;
  NullS:=9;
  NullL:=11;
  if NullS>0 then
  Begin
    for IDX := 1 to NullS do
      Begin
        Smaller.Add('blank');
      End;
  End;
```

```
  if NullL>0 then
  Begin
    for IDX := 1 to NullL do
      Begin
        Bigger.Add('blank');
      End;
  End;
  Smaller.SaveToFile('C:\CPT\StFiles\SmallerCM.txt');
  Bigger.SaveToFile('C:\CPT\StFiles\BiggerCM.txt');
  FileList.SaveToFile('C:\CPT\StFiles\AllCM.txt');
  Smaller.Free;
  Bigger.Free;
  FileList.Clear;
  FileList.Free;
end;

Procedure OpenMainFiles(Var LTargets,LNon: Integer; Phase: String);
Var Counts : Integer;
Temp,FileNameML,FileNameMS : String;
Begin
  FileNameML:='C:\CPT\StFiles\Large'+Phase+'.txt';
  FileNameMS:='C:\CPT\StFiles\Small'+Phase+'.txt';
  Assign(InfileM,FileNameML);
  Reset(InfileM);
  Counts:=0;
    While NOT Eof(InfileM) DO
      Begin
        Readln(InfileM,Temp);
        Counts:=Counts+1;
      End;
    LTargets:=Counts;
    SetLength(PracTargets,LTargets+1);
    Reset(InfileM);
    For Counts:= 1 to LTargets DO
      Begin
        Readln(InfileM,PracTargets[Counts].TargetStr);
        PracTargets[Counts].TargetValue:=0;
      End;
    Reset(InfileM);
  Assign(InfileM,FileNameMS);
  Reset(InfileM);
  Counts:=0;
    While NOT Eof(InfileM) DO
      Begin
        Counts:=Counts+1;
        Readln(InfileM,Temp);
      End;
    LNon:=Counts;
    SetLength(PracNonTargets,LNon+1);
    Reset(InfileM);
```

```
  For Counts:=1 to LNon DO
  Begin
    Readln(InfileM,PracNonTargets[Counts].NonTargetString);
    PracNonTargets[Counts].NonTargetValue:=1;
  End;
  Close(InfileM);
End;

Procedure SortArray(Len : Integer);
var
  I, J, T: Integer;
  TS : String[10];
Begin
  for I := 0 To Len Do
    for J := Len-1 downto I do
      if ListArray[I].Index > ListArray[J].Index then
      begin
        T:=ListArray[I].Index;
        TS:=ListArray[I].Item;
        ListArray[I].Item:=ListArray[J].Item;
        ListArray[I].Index:=ListArray[J].Index;
        ListArray[J].Item:=TS;
        ListArray[J].Index:=T;
      end;
End;

Procedure RandomOrderList(Var FileList: TStringList);
Var
  RR : Integer;
  I : Integer;
  Len : Integer;
  Itm : String[20];
Begin
  Len:=FileList.Count;
  SetLength(ListArray,Len);
  for I := 0 to FileList.Count - 1 do
  Begin
   RR:=Random(100000);
   Itm:=FileList[I];
   Itm:=AnsiReplaceText(Itm,'.bmp','');
   ListArray[I].Item:=Itm;
   ListArray[I].Index:=RR;
  End;
  SortArray(Len);
  FileList.Clear;
  for I:=0 To Len-1 do
    Begin
      FileList.Add(ListArray[I].Item);
    End;
End;
```

```
procedure ExtractFileNames;
Var
  ReturnCode :  Cardinal;
  Path       :  STRING;
  FileSpec   :  STRING;
  Line,Str   :  STRING;
  SearchRec  :  TSearchRec;
  IDX,NullS,NullL,SmallerLen,BiggerLen,BiggerIDX,SmallerIDX : Integer;
  FileList : TStringList;
  Smaller,Smaller1,Smaller2,Smaller3,Smaller4 : TStringList;
  Bigger,Bigger1,Bigger2,Bigger3,Bigger4 : TStringList;
begin
  FileList:=TStringList.Create;
FileSpec := '*.bmp';
Path:='C:\CPT\Files\ShoeBox\All Pictures\';
ReturnCode := SysUtils.FindFirst(Path + Filespec, faAnyFile, SearchRec);
  WHILE ReturnCode = 0 DO
    Begin
      Line := SearchRec.Name;
      FileList.Add(Line);
      ReturnCode := SysUtils.FindNext(SearchRec);
    End;
  FileList.SaveToFile('C:\CPT\StFiles\BMPListALL.txt');
  RandomOrderList(FileList);
  Smaller:=TStringList.Create;
  Bigger:=TStringList.Create;
  for IDX := 0 to FileList.Count - 1 do
  Begin
    Line:=FileList[IDX];
    Str:=MidStr(Line,1,1);
    if Str='S' then
      Smaller.Add(Line);
    if Str='L' then
      Bigger.Add(Line);
  End;
  SmallerLen:=Smaller.Count Div 4;
  BiggerLen:=Bigger.Count Div 4;
  SmallerIDX:=SmallerLen;
  BiggerIDX:=BiggerLen;
  NullS:=SmallerLen Div 2;
  NullL:=BiggerLen Div 2;

  Bigger1:=TStringList.Create;
  for IDX:=0 to BiggerIDX- 1 do
    Begin
      Bigger1.Add(Bigger[IDX]);
    End;
    if NullL>0 then
```

```
Begin
  for IDX := 1 to NullL do
    Begin
      Bigger1.Add('Null');
    End;
End;
Bigger1.SaveToFile('C:\CPT\StFiles\Large1.txt');
Bigger1.Free;

Bigger2:=TStringList.Create;
for IDX:= BiggerIDX to (BiggerIDX+BiggerLen)-1 do
  Begin
    Bigger2.Add(Bigger[IDX]);
  End;
  if NullL>0 then
Begin
  for IDX := 1 to NullL do
    Begin
      Bigger2.Add('Null');
    End;
End;
Bigger2.SaveToFile('C:\CPT\StFiles\Large2.txt');
Bigger2.Free;

BiggerIDX:=BiggerIDX+BiggerLen;
Bigger3:=TStringList.Create;
for IDX:= BiggerIDX to (BiggerIDX+BiggerLen)-1 do
  Begin
    Bigger3.Add(Bigger[IDX]);
  End;
  if NullL>0 then
Begin
  for IDX := 1 to NullL do
    Begin
      Bigger3.Add('Null');
    End;
End;
Bigger3.SaveToFile('C:\CPT\StFiles\Large3.txt');
Bigger3.Free;

BiggerIDX:=BiggerIDX+BiggerLen;
Bigger4:=TStringList.Create;
for IDX:= BiggerIDX to (BiggerIDX+BiggerLen)-1 do
  Begin
    Bigger4.Add(Bigger[IDX]);
  End;
  if NullL>0 then
Begin
  for IDX := 1 to NullL do
    Begin
```

```
      Bigger4.Add('Null');
    End;
End;
Bigger4.SaveToFile('C:\CPT\StFiles\Large4.txt');
Bigger4.Free;

Smaller1:=TStringList.Create;
for IDX:=0 to SmallerLen- 1 do
  Begin
    Smaller1.Add(Smaller[IDX]);
  End;
  if NullS>0 then
Begin
  for IDX := 1 to NullS do
    Begin
      Smaller1.Add('Null');
    End;
End;
Smaller1.SaveToFile('C:\CPT\StFiles\Small1.txt');
Smaller1.Free;

Smaller2:=TStringList.Create;
for IDX:= SmallerIDX to (SmallerIDX+SmallerLen)-1 do
  Begin
    Smaller2.Add(Smaller[IDX]);
  End;
  if NullS>0 then
Begin
  for IDX := 1 to NullS do
    Begin
      Smaller2.Add('Null');
    End;
End;
Smaller2.SaveToFile('C:\CPT\StFiles\Small2.txt');
Smaller2.Free;

SmallerIDX:=SmallerIDX+SmallerLen;
Smaller3:=TStringList.Create;
for IDX:= SmallerIDX to (SmallerIDX+SmallerLen)-1 do
  Begin
    Smaller3.Add(Smaller[IDX]);
  End;
  if NullS>0 then
Begin
  for IDX := 1 to NullS do
    Begin
      Smaller3.Add('Null');
    End;
End;
Smaller3.SaveToFile('C:\CPT\StFiles\Small3.txt');
```

```
  Smaller3.Free;

  SmallerIDX:=SmallerIDX+SmallerLen;
  Smaller4:=TStringList.Create;
  for IDX:= SmallerIDX to (SmallerIDX+SmallerLen)-1 do
    Begin
      Smaller4.Add(Smaller[IDX]);
    End;
    if NullS>0 then
  Begin
    for IDX := 1 to NullS do
      Begin
        Smaller4.Add('Null');
      End;
  End;
  Smaller4.SaveToFile('C:\CPT\StFiles\Small4.txt');
  Smaller4.Free;

  NullS:=Smaller.Count Div 2;
  NullL:=Bigger.Count Div 2;
  if NullS>0 then
  Begin
    for IDX := 1 to NullS do
      Begin
        Smaller.Add('Null');
      End;
  End;
  if NullL>0 then
  Begin
    for IDX := 1 to NullL do
      Begin
        Bigger.Add('Null');
      End;
  End;
  Smaller.SaveToFile('C:\CPT\StFiles\Small.txt');
  Bigger.SaveToFile('C:\CPT\StFiles\Large.txt');
  Smaller.Free;
  Bigger.Free;
  FileList.Clear;
  FileList.Free;
end;
Procedure GetFileNames;
Var InputFile : Text; TempS : String;
Begin
  Assign(InputFile,'C:\CPT\StFiles\StartFiles.txt');
  Reset(InputFile);
  Readln(InputFile,TempS);
  if TempS='short' then
    Begin
      FLarge:='C:\CPT\StFiles\Large1.txt';
```

```
      FSmall:='C:\CPT\StFiles\Small1.txt';
      FSmallerCM:='C:\CPT\StFiles\SmallerCMShort.txt';
      FBiggerCM:='C:\CPT\StFiles\BiggerCMShort.txt';
    End
      Else
        Begin
          FLarge:='C:\CPT\StFiles\Large1.txt';
          FSmall:='C:\CPT\StFiles\Small1.txt';
          FSmallerCM:='C:\CPT\StFiles\SmallerCM.txt';
          FBiggerCM:='C:\CPT\StFiles\BiggerCM.txt';
        End;
End;

unit Sorting;
...

interface
Uses Cpt,Sysutils;
Procedure TrueFalse(Var Flag: Boolean); External
'TF.dll';
Procedure SortTargetType(Var Both : TBoth; Arraylength : Integer);

implementation
Procedure SortTargetType(Var Both : TBoth; Arraylength : Integer);
Var HowLong,LoopIdx,MaxM,MaxP : Integer;
Flag,MComplete,PComplete : Boolean;
Begin
  HowLong:=Arraylength Div 2;
  // Initialise all to 6
  for LoopIdx:=1 To ArrayLength do
    Begin
      Both[LoopIdx].TargetType:=6;
    End;
  MaxM:=0;
  MaxP:=0;
  MComplete:=False;
  PComplete:=False;
  for LoopIdx := 1 To ArrayLength do
    Begin
      if MaxM>Howlong then
        MComplete:=True;
      if MaxP>Howlong then
        PComplete:=True;
      TrueFalse(Flag);
      if Flag=True then
      Begin
        if Not(MComplete) then
        Begin
          Both[LoopIdx].TargetType:=0;
          MaxM:=MaxM+1;
```

```
      End;
    End;
    if Flag=False then
    Begin
      if Not(PComplete) then
      Begin
        Both[LoopIdx].TargetType:=1;
        MaxP:=MaxP+1;
      End;
    End;
  End;
  for LoopIdx:=1 To ArrayLength do
    Begin
      if Both[LoopIdx].TargetType=6 then
        Begin
          if MComplete then Both[LoopIdx].TargetType:=1;
          if PComplete then Both[LoopIdx].TargetType:=0;
        End;
    End;
End;
```

Appendix 4

See media disk for this appendix. This appendix contains five folders in a sub-folder named ‘Appendix 4’ under ‘Appendices Applications and materials’ on the media disk and in the Dropbox folder. Each folder contains text files with the source code for the experiments. This material may also be accessed from a Dropbox location which will be shared on request to Prof. Colin Tredoux (colin.tredoux@uct.ac.za).

Appendix 5

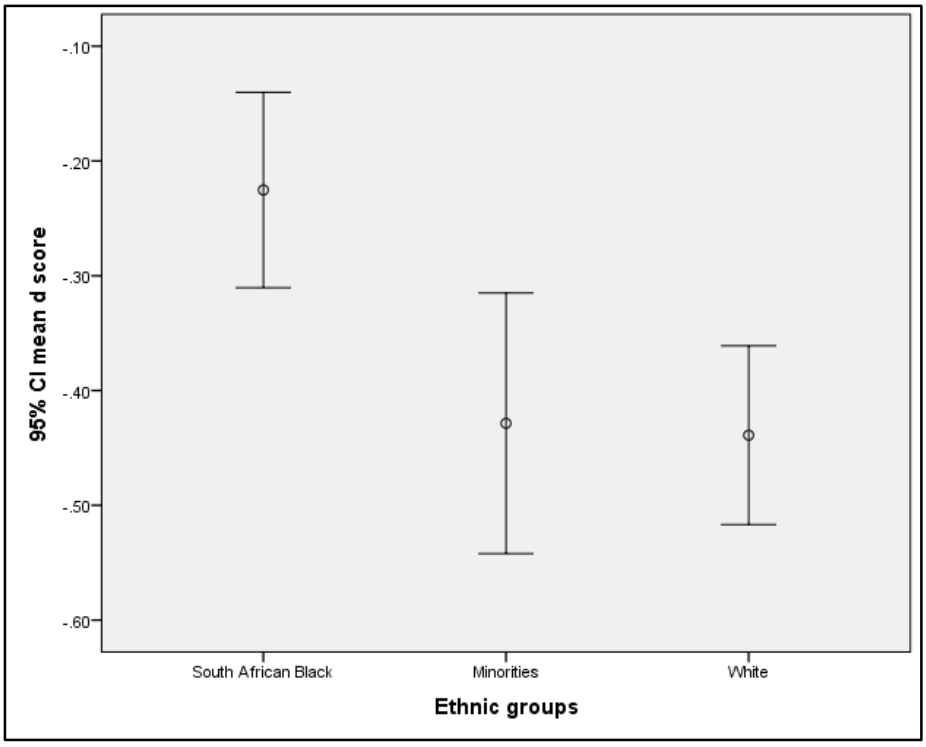


Figure 145. IAT Experiment 2, 95% CI of d means for ethnic groupings.

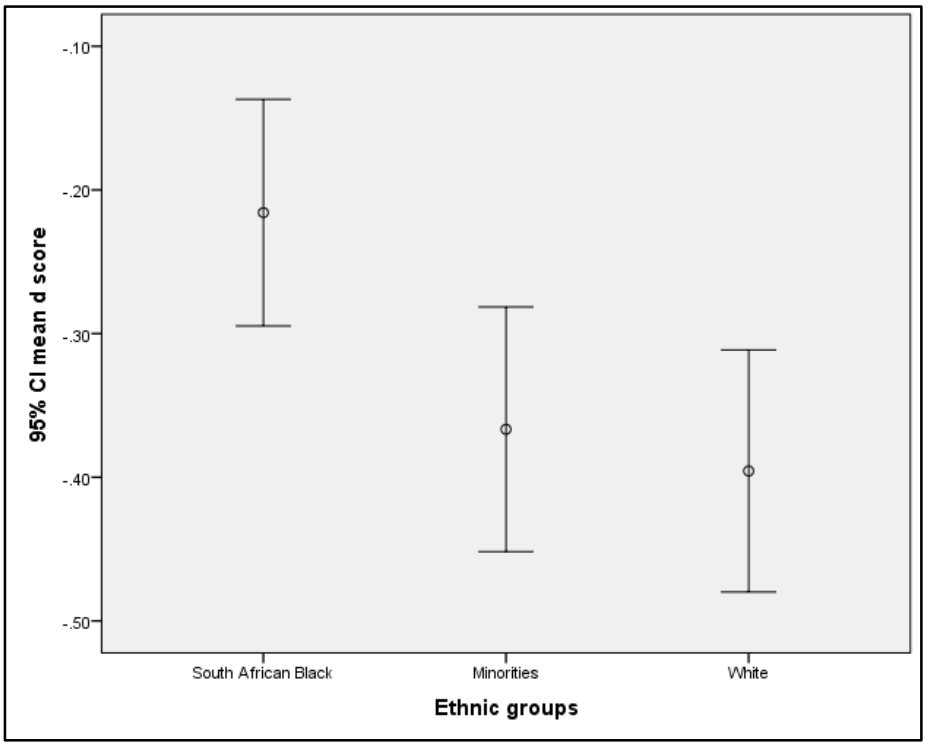


Figure 146. IAT Experiment 3, 95% CI of d means for ethnic groupings.

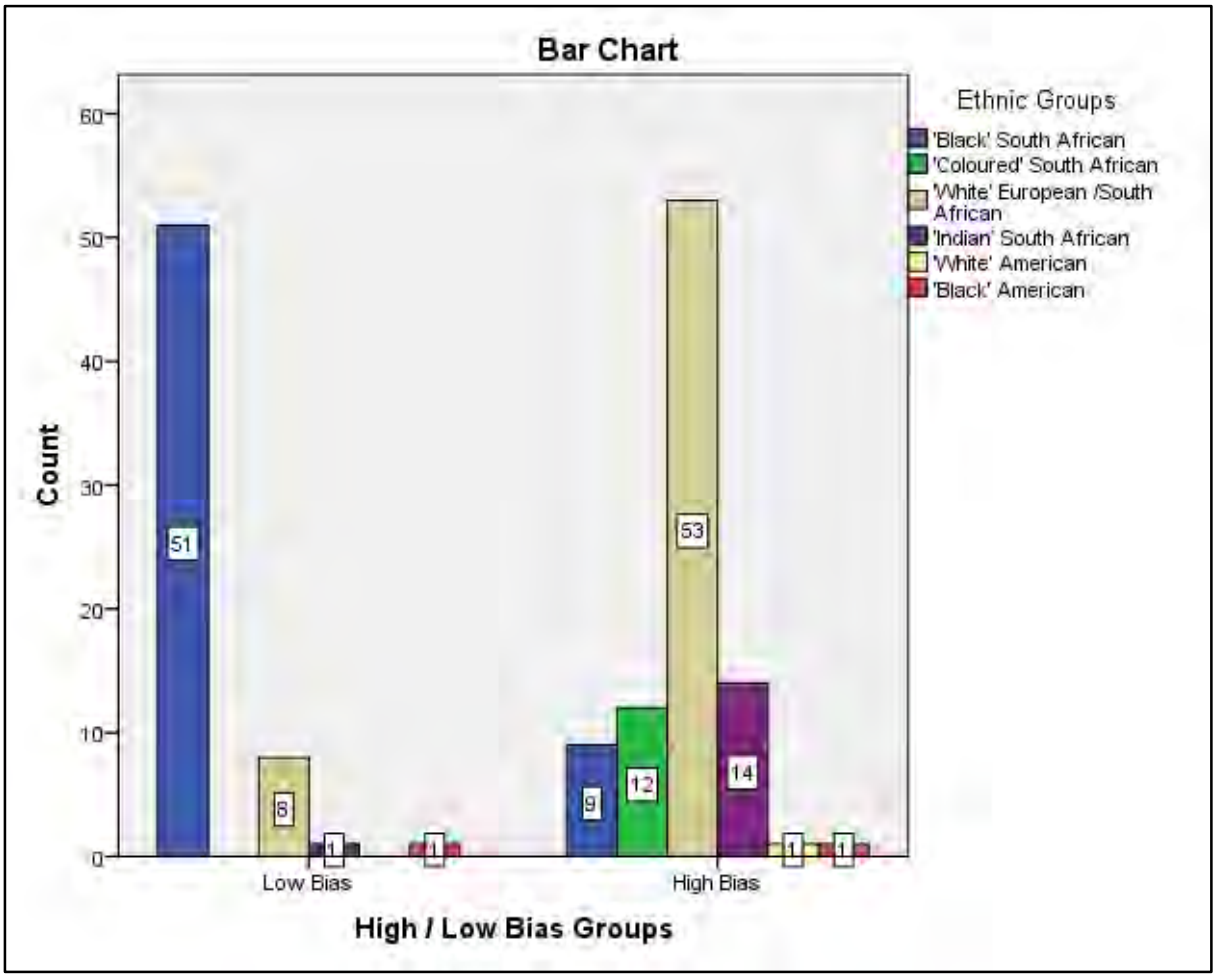


Figure 147 Ethnic composition of IAT High /Low bias groups.

Appendix 6

Explanation and demonstration of screen lumination concepts and 'RGB' convention for defining these concepts

A line drawing has values which range from 0 – 255 ($FF in hex, depending on the notation). The brightest colour that a computer screen can display is pure white (RGB 255, 255, 255). That is, each pixel is illuminated to its maximum value of 255. This is a decimal value that corresponds to $FF in hex, or 11111111 in binary. An LCD or CRT screen contains very small dots or pixels which vary in illumination from 0 to 255 ($FF). There are Red, Green and Blue coloured pixels which vary by the same amount (i.e. 0 – 255). The method of colour mixing for a computer screen is the same as that used in a television screen – namely, additive colour mixing. When only the Red pixels are illuminated, one sees pure Red on the screen – see the illustration below:

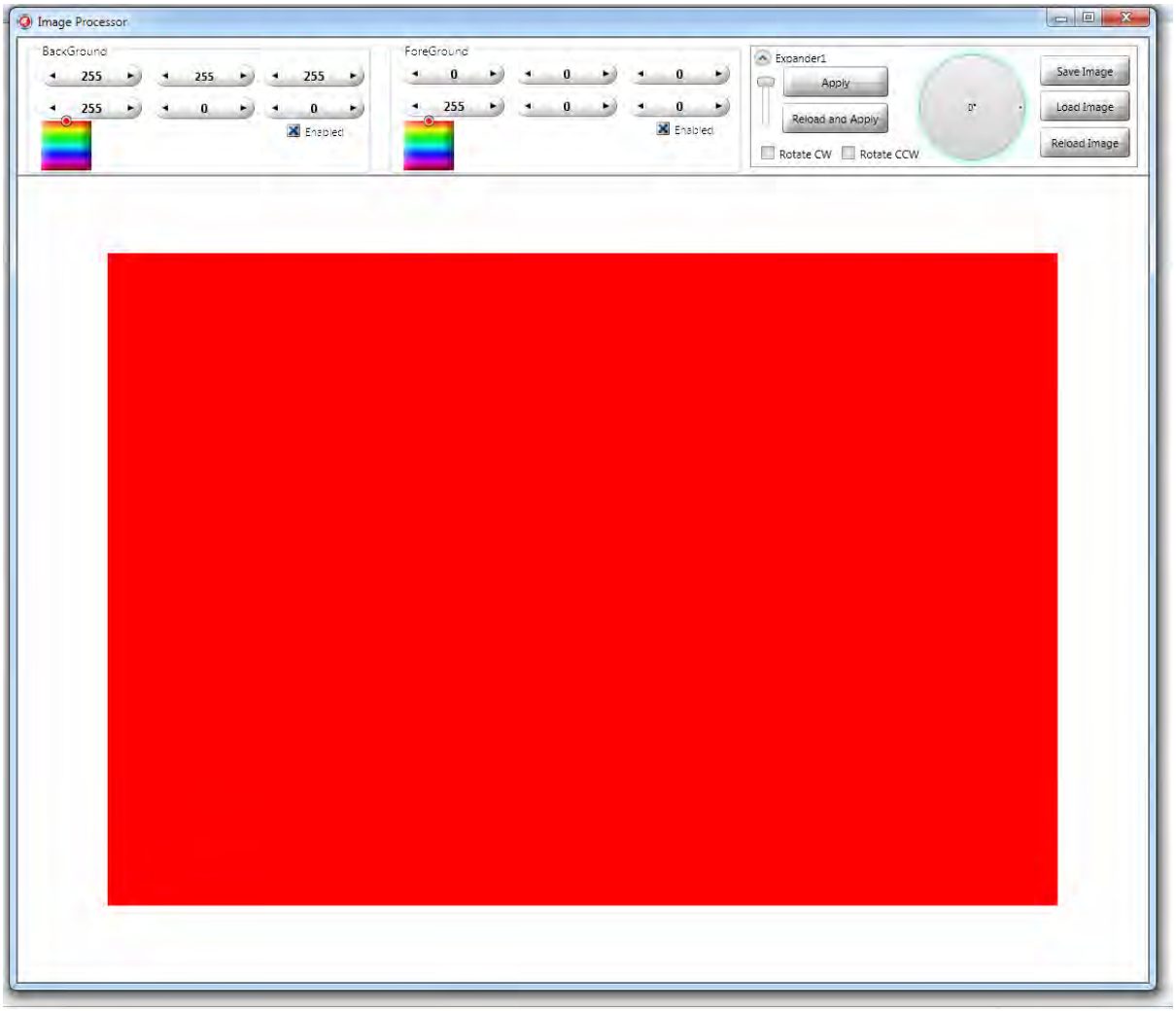


Figure 148. Line drawing with all values converted to RGB(255,255,255).

This drawing was converted from the following original:

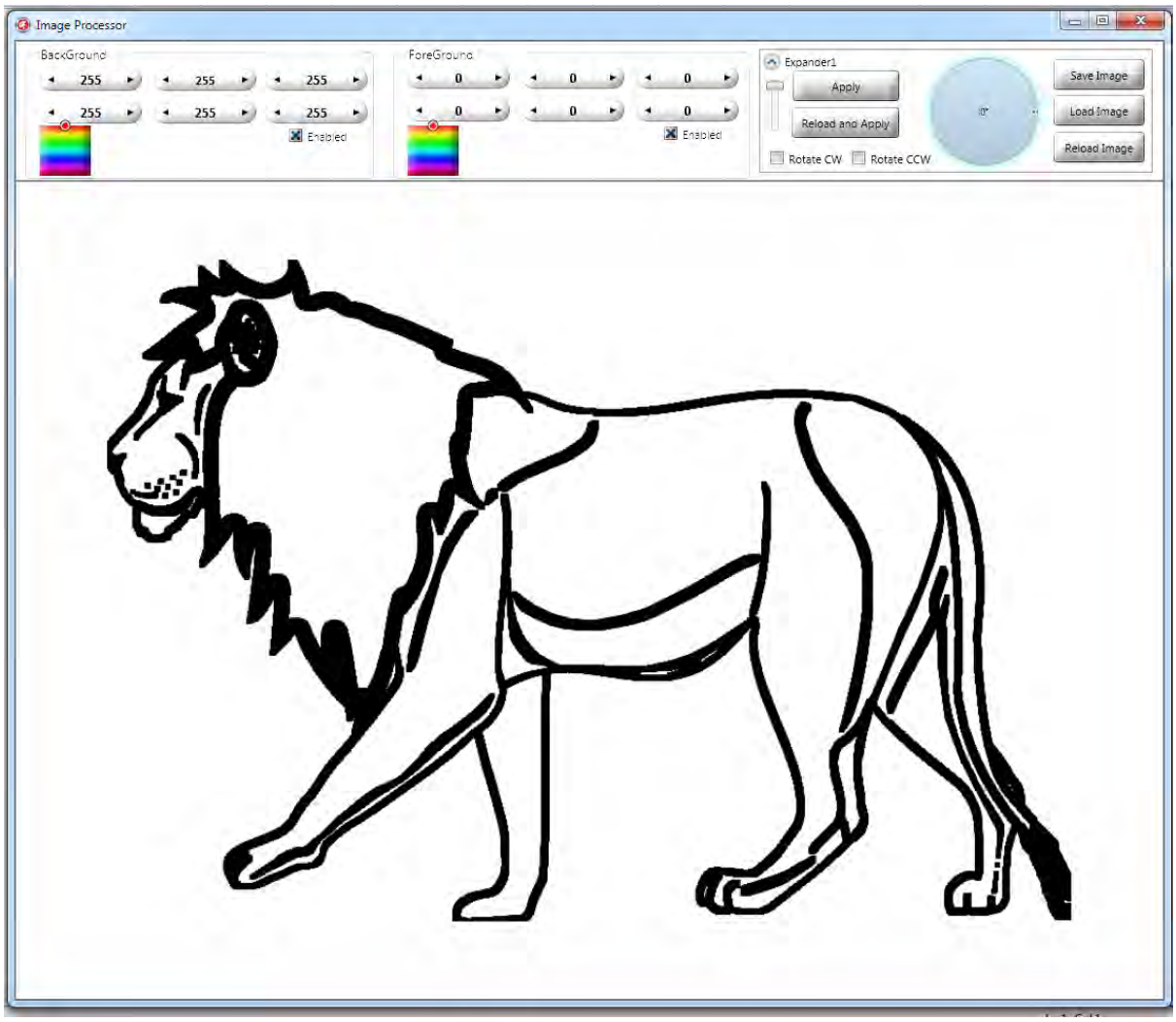


Figure 149. In the above drawing it may be seen that the so-called 'Background' values were converted from RGB(255,255,255) to RGB(255,255,255). That is, they remained the same (i.e. pure white). The soc-called 'Forground' values (pure black) were converted from RGB(0,0,0) to RGB(0,0,0). Again they remained exactly the same.

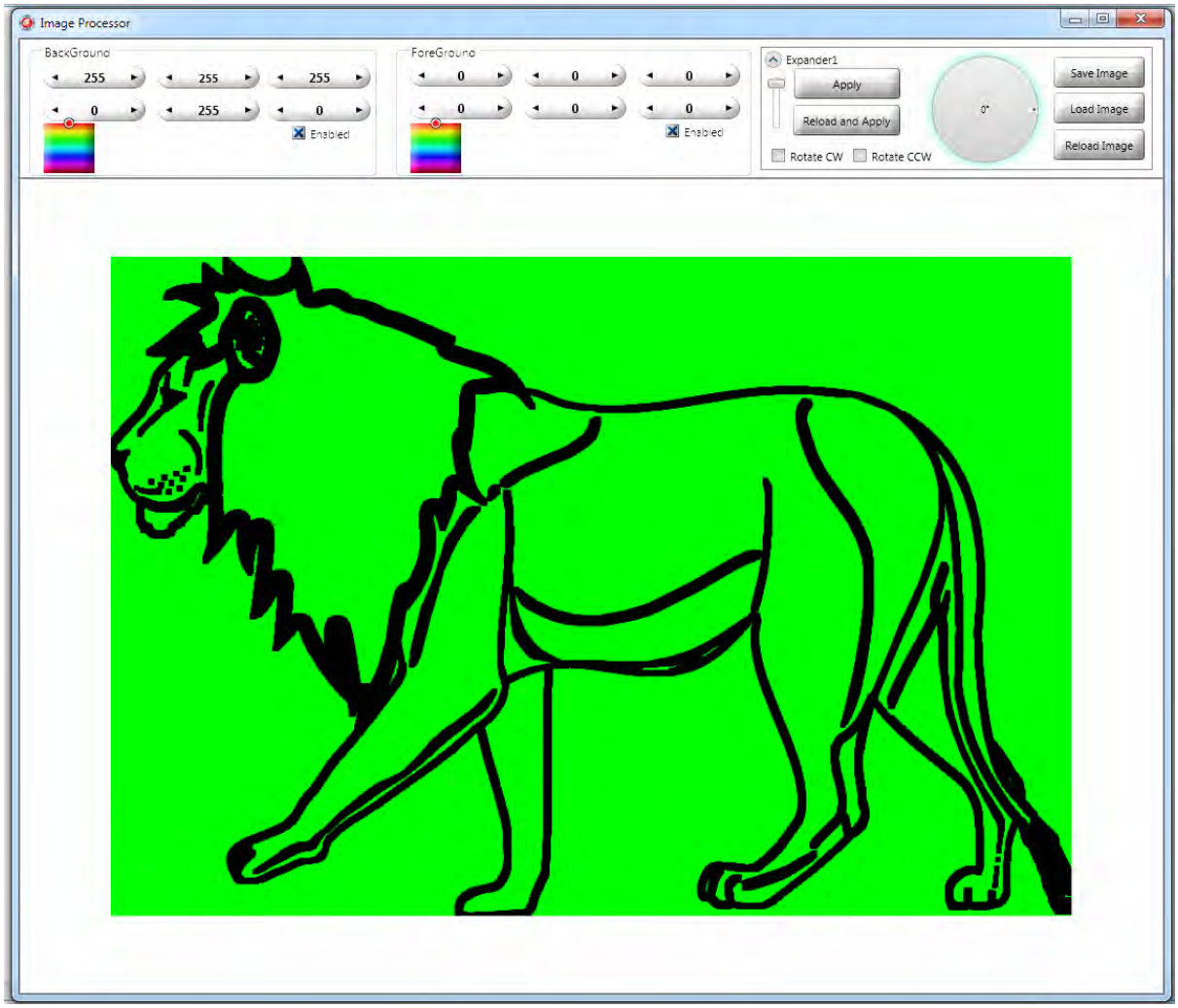


Figure 150. When the so-called 'background' values are converted from RGB(255,255,255) to RGB(255,0,255) and appears bright green. The so-called 'foreground' values are left unchanged at RGB(0,0,0), the black outline of the image remains black.

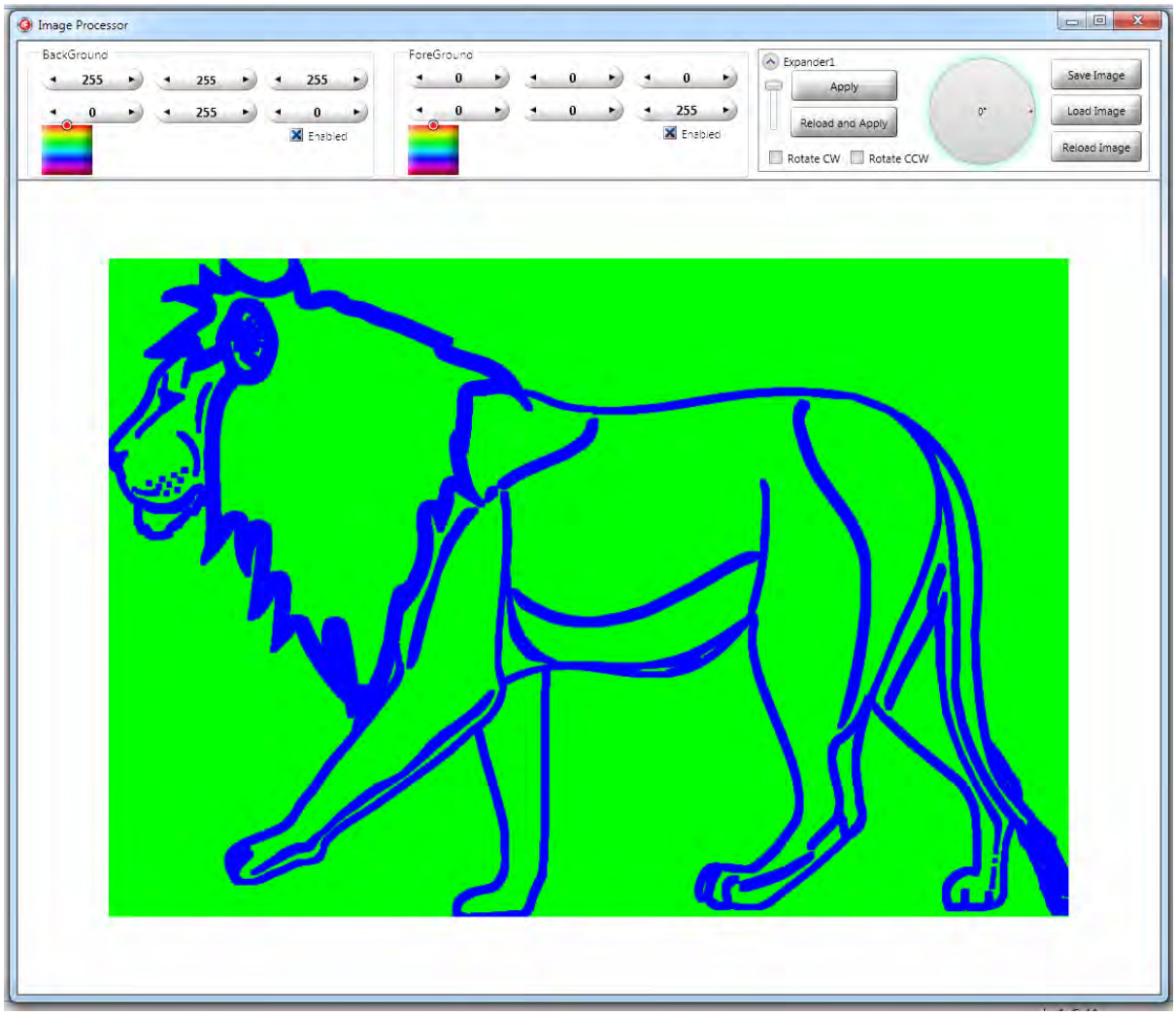


Figure 151. When the so-called 'background' colour is converted from RGB(255,255,255) to RGB(0,255,0) the colour is again bright green. When the so-called 'foreground' colour is converted from RGB(0,0,0) to RGB(0,0,255), the black outline appears bright blue.

In the Replication experiment, the original source line drawings appeared thus:

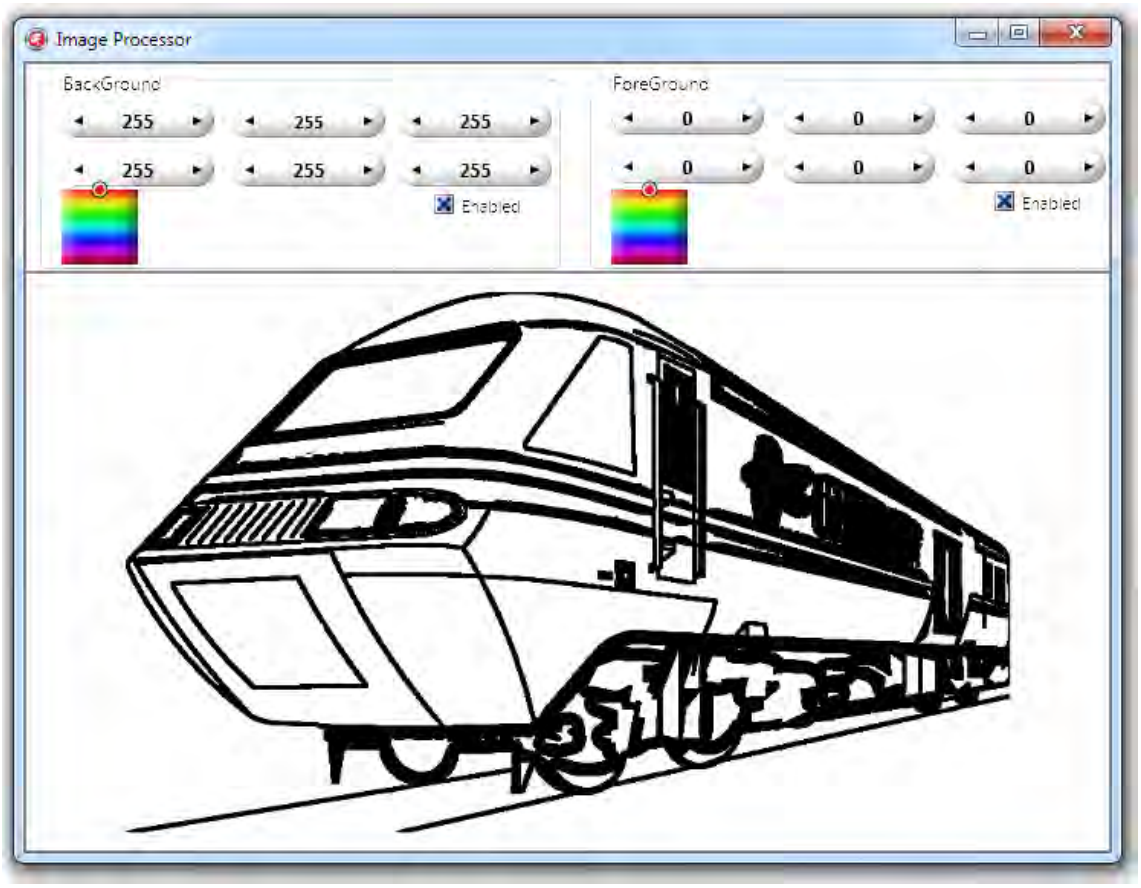


Figure 152. Original source line drawing for Replication study. The 'background' colour was pure white – RGB(255,255,255) and the 'forground' colour was pure black – RGB(0.0.0).

Compare this with the picture (Figure 153) where the 'background' was transformed from pure white – RGB(255,255,255) to RGB(31,31,31) and the 'foreground' was transformed to gray – RGB(42, 42,42) from pure black – RGB(0,0,0). This was the negative extreme towards which the various values would cycle.

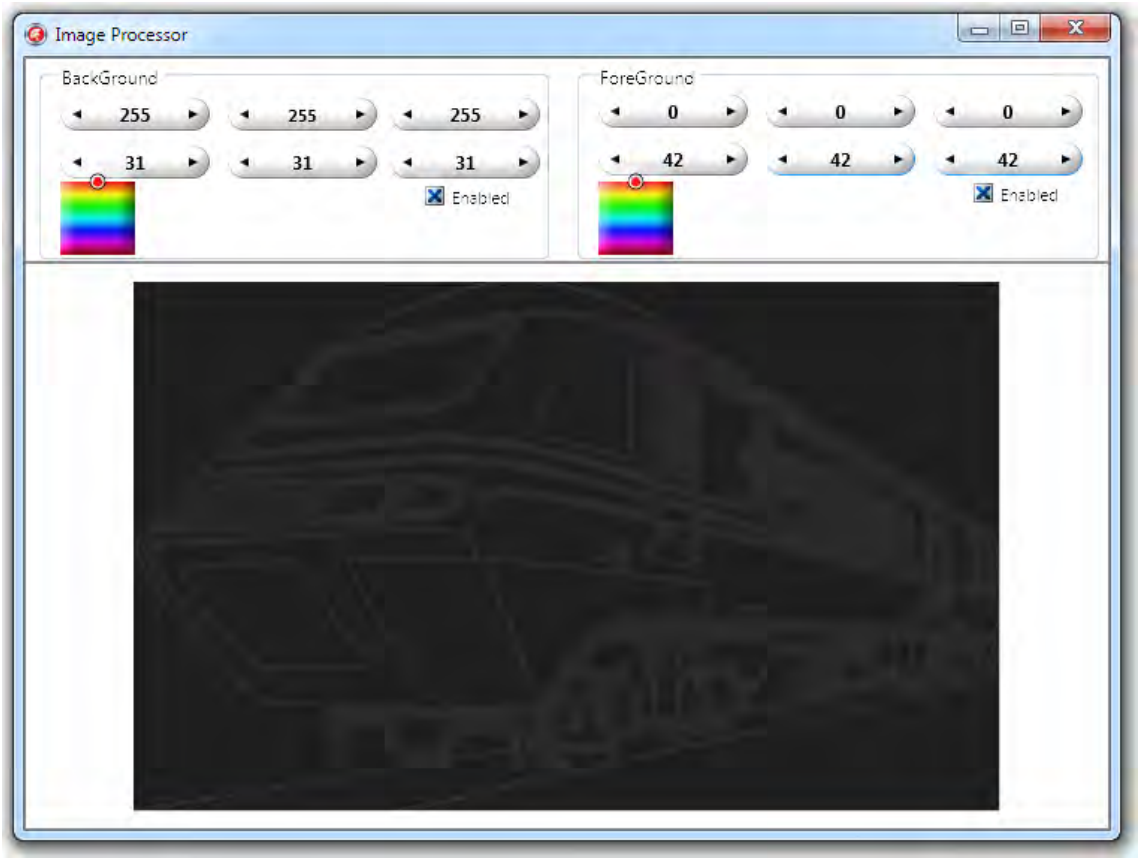


Figure 153. Image transformed to negative extreme value from the pure white 'background' colour of RGB(255,255,255) to RGB(31,31,31) and the 'foreground' was transformed to gray - RGB(42,42,42) from pure black = RGB(0,0,0). This was the most negative value to which the 'foreground' value would cycle to. The 'background' shade was held constant at RGB(31,31,31).

Compare this with the picture (Figure 154) where the 'background' was transformed from pure white – RGB(255,255,255) to RGB(31,31,31) and the 'foreground' was transformed to gray – RGB(20,20,20) from pure black – RGB(0,0,0). This was the positive extreme towards which the various values would cycle.

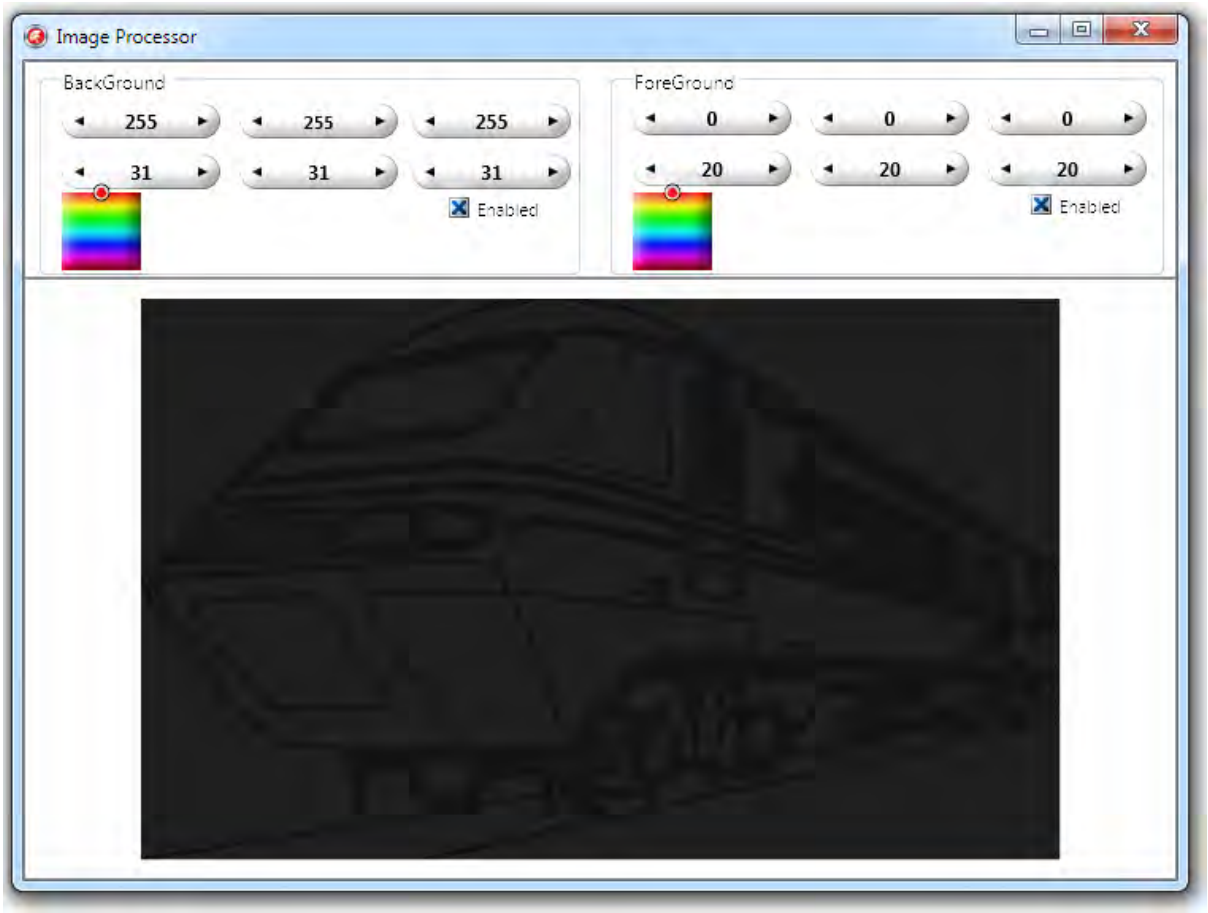


Figure 154. Image transformed to negative extreme value from the pure white 'background' colour of RGB(255,255,255) to RGB(31,31,31) and the 'foreground' was transformed to gray - RGB(20,20,20) from pure black = RGB(0,0,0). This was the most positive value to which the 'foreground' value would cycle to. The 'background' shade was held constant at RGB(31,31,31).

Naturally when the 'foreground' and 'background' values are transformed to the same values, the picture is effectively blank as the image foreground is at the same level as the image background – see Figure 154.

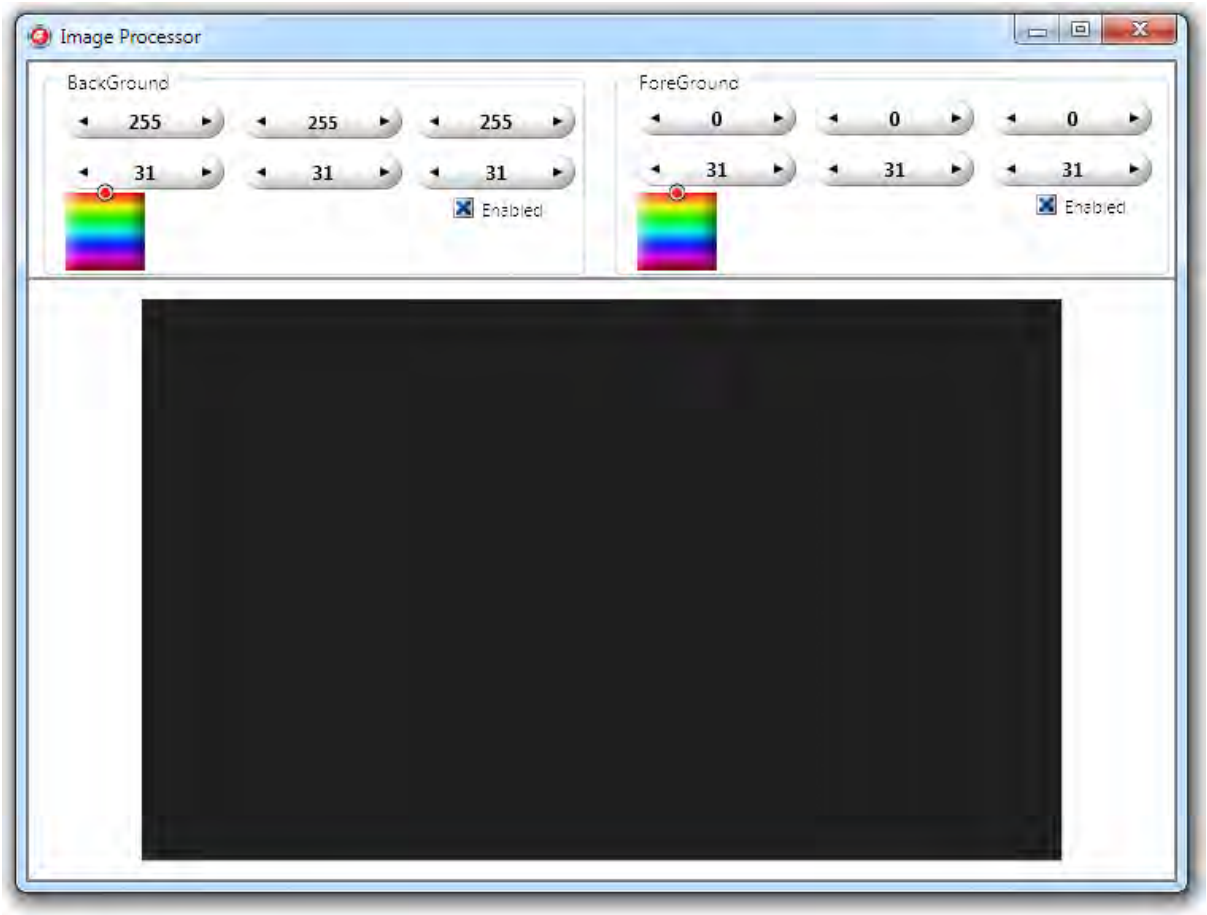


Figure 155. Image transformed to similar values: 'foreground' RGB(31,31,31), 'background' RGB(31,31,31). The image is blank because the 'foreground' and 'background' values are the same.

Figure 155 shows a blank image presentation where the foreground and background are the same shade. This was effectively a null presentation.

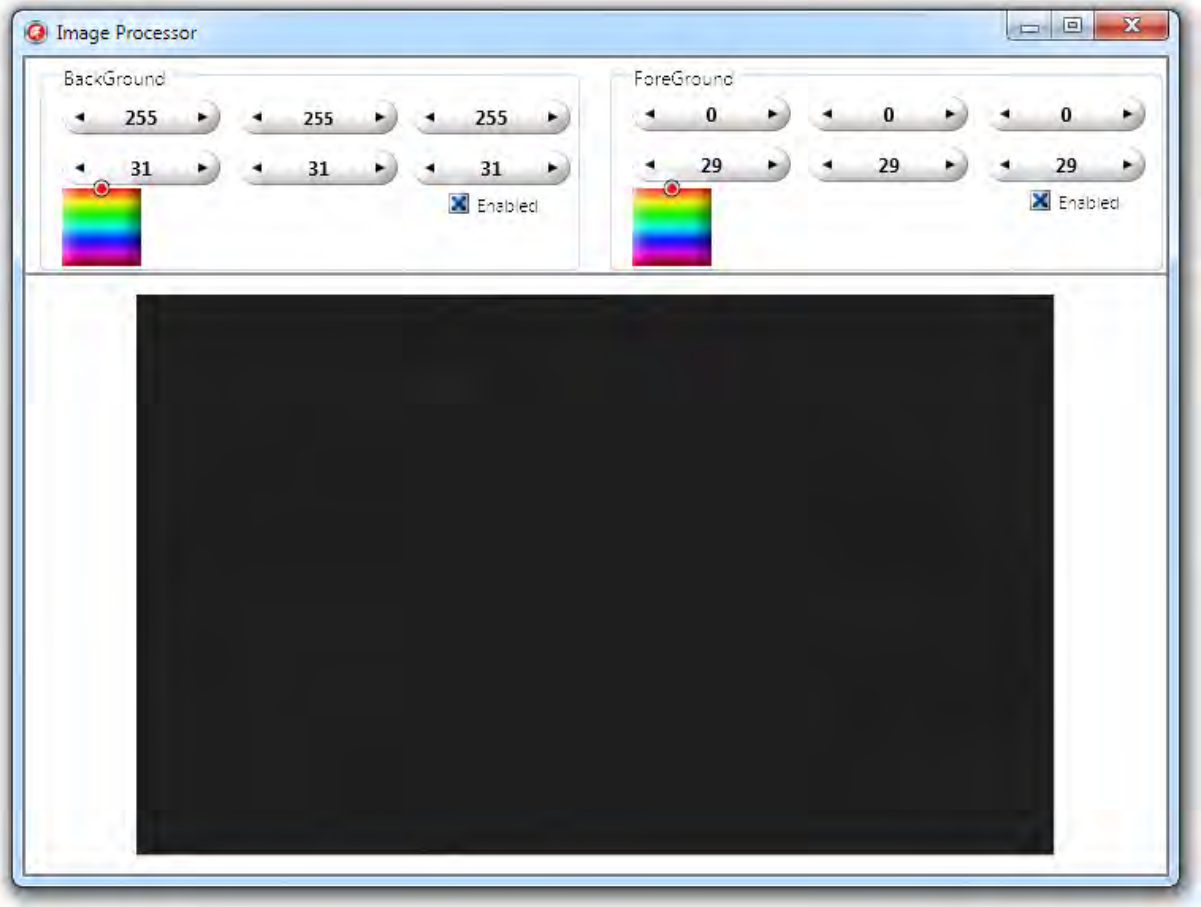


Figure 156. This shows an image presentation where the foreground and background values are within 4 points of each other. This was one of the most common contrast levels (negative) at which stimuli were recognised.

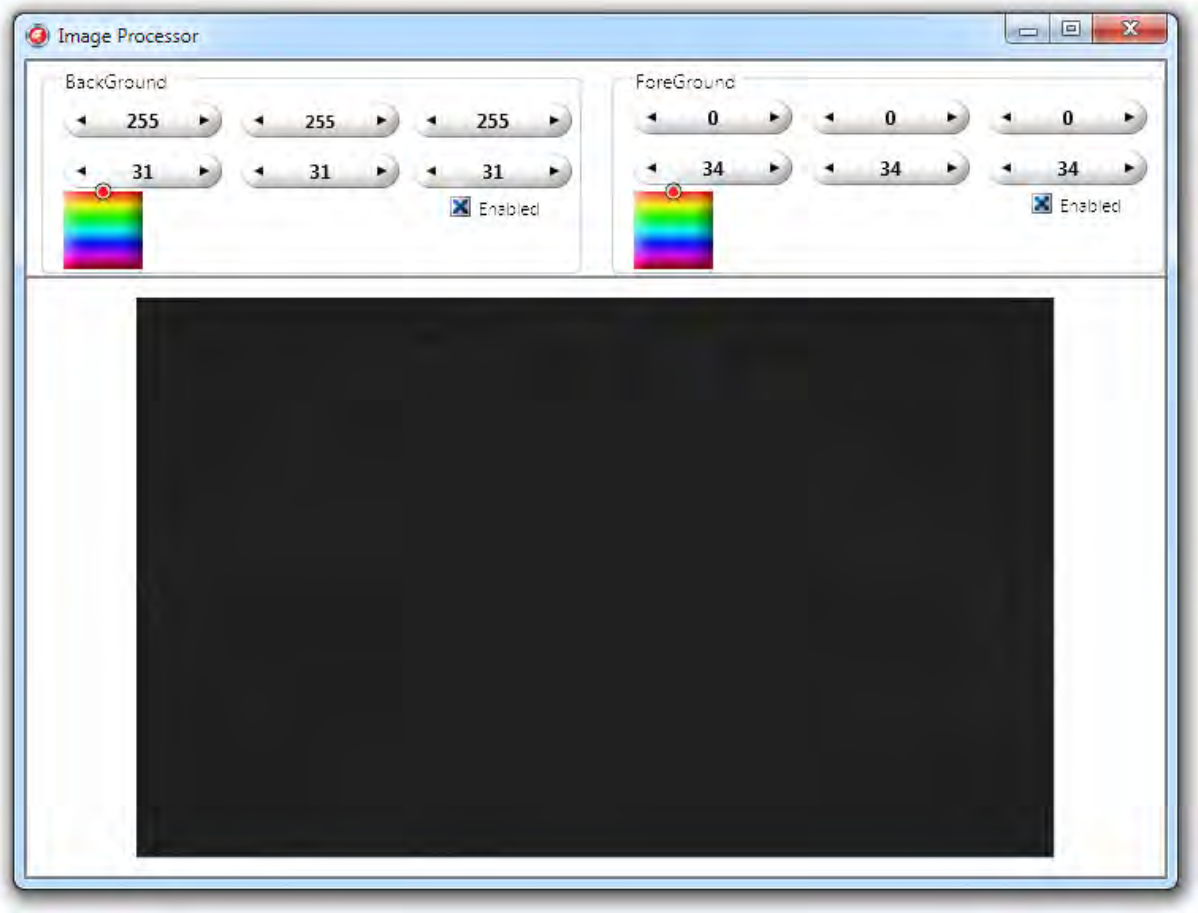


Figure 157. This shows an image presentation where the foreground and background values are within 3 points of each other. This was one of the most common contrast levels (positive) at which stimuli were recognised.

Figure 157 shows the most common positive contrast levels at which stimuli were visible. It is important to note that this will vary between different screens, since each screen has different characteristics which affect stimulus visibility.

Delphi Code for Image Processor Application

```
unit Unit1;
// Amended 13 May 2012
interface

uses
  System.SysUtils, System.Types, System.UITypes, System.Classes, System.Variants,
  StrUtils, FMX.Controls, FMX.Forms, FMX.Dialogs, FMX.Objects, FMX.Edit,
  FMX.Filter.Effects, FMX.Effects, FMX.Layouts, FMX.ExtCtrls, FMX.ListBox,
  FMX.Ani, FMX.Colors, FMX.Types;

type
  TMyPixelDescriptor = record
    Blue: Byte;
    Green: Byte;
    Red: Byte;
  end;

  PMyPixelArray = ^TMyPixelArray;
  TMyPixelArray = array[0..32767] of TMyPixelDescriptor;
  TMyPictureArray = array of TMyPixelArray;
  TForm1 = class(TForm)
    Image1: TImage;
    Button1: TButton;
    SR: TSpinBox;
    SG: TSpinBox;
    SB: TSpinBox;
    DR: TSpinBox;
    DG: TSpinBox;
    DB: TSpinBox;
    FSR: TSpinBox;
    FSG: TSpinBox;
    FSB: TSpinBox;
    FDR: TSpinBox;
    FDG: TSpinBox;
    FDB: TSpinBox;
    CheckBox1: TCheckBox;
    CheckBox2: TCheckBox;
    Background: TGroupBox;
    ForeGround: TGroupBox;
    BevelEffect1: TBevelEffect;
    BevelEffect2: TBevelEffect;
    BevelEffect3: TBevelEffect;
    Button2: TButton;
    BevelEffect4: TBevelEffect;
    OpenDialog1: TOpenDialog;
    Button3: TButton;
    BevelEffect5: TBevelEffect;
    ImageViewer1: TImageViewer;
```

```
Panel1: TPanel;
Button4: TButton;
BevelEffect6: TBevelEffect;
Expander1: TExpander;
Button5: TButton;
BevelEffect7: TBevelEffect;
SaveDialog1: TSaveDialog;
ArcDial1: TArcDial;
GlowEffect1: TGlowEffect;
CheckBox3: TCheckBox;
CheckBox4: TCheckBox;
Timer1: TTimer;
TrackBar1: TTrackBar;
ColorAnimation1: TColorAnimation;
ComboBox1: TComboBox;
ListBoxItem1: TListBoxItem;
ListBoxItem2: TListBoxItem;
ListBoxItem3: TListBoxItem;
ListBoxItem4: TListBoxItem;
ListBoxItem5: TListBoxItem;
ListBoxItem6: TListBoxItem;
ListBoxItem7: TListBoxItem;
ListBoxItem8: TListBoxItem;
ListBoxItem9: TListBoxItem;
ListBoxItem10: TListBoxItem;
ListBoxItem11: TListBoxItem;
ListBoxItem12: TListBoxItem;
Label1: TLabel;
ColorPicker1: TColorPicker;
ColorPicker2: TColorPicker;
Label2: TLabel;
procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Button5Click(Sender: TObject);
procedure Button4Click(Sender: TObject);
procedure ArcDial1Change(Sender: TObject);
procedure ArcDial1DblClick(Sender: TObject);
procedure CheckBox3Click(Sender: TObject);
procedure CheckBox4Click(Sender: TObject);
procedure Timer1Timer(Sender: TObject);
procedure TrackBar1Change(Sender: TObject);
procedure ComboBox1Change(Sender: TObject);
Procedure BlueGreen;
Procedure RedGreen;
Procedure Line;
Procedure MColours;
Procedure DBlueGreen;
Procedure DRedGreen;
Procedure Snake1;
```

```
    Procedure LRedGreen;
    Procedure YellowCyan;
    Procedure BrightCyanGreen;
    Procedure EscherBlue;
    Procedure EscherBlueY;
    Procedure ReloadAndChange;
    procedure ColorPicker1Click(Sender: TObject);
    procedure Expander1Resize(Sender: TObject);
    procedure ColorPicker2Click(Sender: TObject);
  private
    { Private declarations }
  public
     RotateA: Integer;
    { Public declarations }
  end;

var
  Form1: TForm1;
  Opened : Boolean;
  FNam : ShortString;
  Const Folder= 'C:\CPT\Files\ShoeBox\Apple2.bmp';

implementation
Uses Thread;


{$R *.fmx}

procedure TForm1.ArcDial1Change(Sender: TObject);
begin
  ImageViewer1.RotationAngle:=360-ArcDial1.Value;
end;

procedure TForm1.ArcDial1DblClick(Sender: TObject);
begin
  ArcDial1.Value:=0;
end;

procedure TForm1.Button1Click(Sender: TObject);
Var AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,AToBB:
Integer;
  ABox1,ABox2: Boolean;
begin
  AFR:=Trunc(Form1.Sr.Value);
  AFG:=Trunc(Form1.Sg.Value);
  AFB:=Trunc(Form1.Sb.Value);
  AToR:=Trunc(Form1.Dr.Value);
  AToG:=Trunc(Form1.Dg.Value);
  AToB:=Trunc(Form1.Db.Value);
  AFBR:=Trunc(Form1.FSR.Value);
```

```
  AFBG:=Trunc(Form1.FSG.Value);
  AFBB:=Trunc(Form1.FSB.Value);
  AToBR:=Trunc(Form1.FDR.Value);
  AToBG:=Trunc(Form1.FDG.Value);
  AToBB:=Trunc(Form1.FDB.Value);
  ABox1:=Form1.CheckBox1.IsChecked;
  ABox2:=Form1.CheckBox2.IsChecked;

ImageThread.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToB
G,
  AToBB,ABox1,ABox2,FNam);
//  OnTerminate
//  TFOrm3.Label2.Caption:='Done';
end;



procedure TForm1.Button2Click(Sender: TObject);
begin
  if Opendialog1.Execute then
    Image1.Bitmap.LoadFromFile(Opendialog1.FileName);
  Form1.ImageViewer1.Bitmap.Assign(Form1.Image1.Bitmap);
  Form1.Image1.Visible:=False;
  ComboBox1.Enabled:=True;
end;


procedure TForm1.Button3Click(Sender: TObject);
begin
  Image1.Bitmap.LoadFromFile(Opendialog1.FileName);
  Form1.ImageViewer1.Bitmap.Assign(Form1.Image1.Bitmap);
  Form1.Image1.Visible:=False;
end;

procedure TForm1.Button4Click(Sender: TObject);
begin
  Form1.Image1.Bitmap.Assign(Form1.ImageViewer1.Bitmap);
  if SaveDialog1.Execute then
    Form1.Image1.Bitmap.SaveToFile(SaveDialog1.FileName);
end;

Procedure TForm1.ReloadAndChange;
Var AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,AToBB:
Integer;
  ABox1,ABox2: Boolean;
begin
  Image1.Bitmap.LoadFromFile(Opendialog1.FileName);
  Form1.ImageViewer1.Bitmap.Assign(Form1.Image1.Bitmap);
  Form1.Image1.Visible:=False;
  AFR:=Trunc(Form1.Sr.Value);
```

```
  AFG:=Trunc(Form1.Sg.Value);
  AFB:=Trunc(Form1.Sb.Value);
  AToR:=Trunc(Form1.Dr.Value);
  AToG:=Trunc(Form1.Dg.Value);
  AToB:=Trunc(Form1.Db.Value);
  AFBR:=Trunc(Form1.FSR.Value);
  AFBG:=Trunc(Form1.FSG.Value);
  AFBB:=Trunc(Form1.FSB.Value);
  AToBR:=Trunc(Form1.FDR.Value);
  AToBG:=Trunc(Form1.FDG.Value);
  AToBB:=Trunc(Form1.FDB.Value);
  ABox1:=Form1.CheckBox1.IsChecked;
  ABox2:=Form1.CheckBox2.IsChecked;

ImageThread.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToB
G,
  AToBB,ABox1,ABox2,FNam);
end;

procedure TForm1.Button5Click(Sender: TObject);
Var AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,AToBB:
Integer;
  ABox1,ABox2: Boolean;
begin
  Image1.Bitmap.LoadFromFile(Opendialog1.FileName);
  Form1.ImageViewer1.Bitmap.Assign(Form1.Image1.Bitmap);
  Form1.Image1.Visible:=False;
  AFR:=Trunc(Form1.Sr.Value);
  AFG:=Trunc(Form1.Sg.Value);
  AFB:=Trunc(Form1.Sb.Value);
  AToR:=Trunc(Form1.Dr.Value);
  AToG:=Trunc(Form1.Dg.Value);
  AToB:=Trunc(Form1.Db.Value);
  AFBR:=Trunc(Form1.FSR.Value);
  AFBG:=Trunc(Form1.FSG.Value);
  AFBB:=Trunc(Form1.FSB.Value);
  AToBR:=Trunc(Form1.FDR.Value);
  AToBG:=Trunc(Form1.FDG.Value);
  AToBB:=Trunc(Form1.FDB.Value);
  ABox1:=Form1.CheckBox1.IsChecked;
  ABox2:=Form1.CheckBox2.IsChecked;

ImageThread.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToB
G,
  AToBB,ABox1,ABox2,FNam);
end;

procedure TForm1.CheckBox3Click(Sender: TObject);
begin
 if CheckBox4.IsChecked then
```

```
    CheckBox4.IsChecked:=False;
    RotateA:=Round(TrackBar1.Value);
    Timer1.Enabled:=True;
end;

procedure TForm1.CheckBox4Click(Sender: TObject);
begin
 if CheckBox3.IsChecked then
    CheckBox3.IsChecked:=False;
    RotateA:=Round(TrackBar1.Value);
    Timer1.Enabled:=True;
end;

Procedure TForm1.BlueGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=150;
 DB.Value:=0;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=0;
 FDG.Value:=150;
 FDB.Value:=70;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.RedGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=141;
 DB.Value:=0;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=150;
 FDG.Value:=0;
 FDB.Value:=0;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;
```

```
Procedure TForm1.Line;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=10;
 DG.Value:=10;
 DB.Value:=10;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=20;
 FDG.Value:=20;
 FDB.Value:=20;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.MColours;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=38;//119;
 DG.Value:=38;//119;
 DB.Value:=38;//119;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=31;//133;
 FDG.Value:=31;//133;
 FDB.Value:=31;//133;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.DBlueGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=200;
 DB.Value:=0;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
```

```
 FDR.Value:=0;
 FDG.Value:=200;
 FDB.Value:=100;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.DRedGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=10;
 DB.Value:=0;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=20;
 FDG.Value:=0;
 FDB.Value:=0;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.Snake1;
Begin
 FSR.Value:=190;
 FSG.Value:=190;
 FSB.Value:=190;
 FDR.Value:=215;
 FDG.Value:=215;
 FDB.Value:=0;
 SR.Value:=86;
 SG.Value:=86;
 SB.Value:=86;
 DR.Value:=0;
 DG.Value:=99;
 DB.Value:=255;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.LRedGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
```

```
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=200;
 DB.Value:=0;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=255;
 FDG.Value:=0;
 FDB.Value:=0;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.YellowCyan;
Begin
 SR.Value:=0;
 SG.Value:=0;
 SB.Value:=0;
 DR.Value:=0;
 DG.Value:=255;
 DB.Value:=255;
 FSR.Value:=255;
 FSG.Value:=255;
 FSB.Value:=255;
 FDR.Value:=255;
 FDG.Value:=255;
 FDB.Value:=0;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.BrightCyanGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=255;
 DB.Value:=0;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=0;
 FDG.Value:=254;
 FDB.Value:=255;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
```

```
 ReloadAndChange;
End;

Procedure TForm1.EscherBlue;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=100;
 DG.Value:=183;
 DB.Value:=230;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=95;
 FDG.Value:=121;
 FDB.Value:=185;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.EscherBlueY;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=90;
 DG.Value:=165;
 DB.Value:=220;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=115;
 FDG.Value:=165;
 FDB.Value:=137;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

procedure TForm1.Expander1Resize(Sender: TObject);
begin
  if Not(Expander1.IsExpanded) then
    ComboBox1.Visible:=True
    Else
      ComboBox1.Visible:=False;
end;

procedure TForm1.ColorPicker1Click(Sender: TObject);
```

```
Var NewColor: Integer;
   RR,GG,BB : Byte;
   CStr,RS,GS,BS : ShortString;
begin
   CheckBox1.IsChecked:=false;
   NewColor:=ColorPicker1.Color;
   CStr:=IntToHex(NewColor,8);
   BS:=MidStr(CStr,7,2);
   GS:=MidStr(CStr,5,2);
   RS:=MidStr(CStr,3,2);
   Label1.Text:='R '+RS+' G '+GS+' B '+BS;
   DR.Value:=StrToInt('$'+RS);
   DG.Value:=StrToInt('$'+GS);
   DB.Value:=StrToInt('$'+BS);
   CheckBox1.IsChecked:=true;
end;

procedure TForm1.ColorPicker2Click(Sender: TObject);
Var NewColor: Integer;
   RR,GG,BB : Byte;
   CStr,RS,GS,BS : ShortString;
begin
   CheckBox2.IsChecked:=false;
   NewColor:=ColorPicker2.Color;
   CStr:=IntToHex(NewColor,8);
   BS:=MidStr(CStr,7,2);
   GS:=MidStr(CStr,5,2);
   RS:=MidStr(CStr,3,2);
   Label2.Text:='R '+RS+' G '+GS+' B '+BS;
   FDR.Value:=StrToInt('$'+RS);
   FDG.Value:=StrToInt('$'+GS);
   FDB.Value:=StrToInt('$'+BS);
   CheckBox2.IsChecked:=true;
end;

procedure TForm1.ComboBox1Change(Sender: TObject);
begin
  CheckBox1.IsChecked:=false;
  CheckBox2.IsChecked:=false;
  if ComboBox1.ItemIndex=0 then  BlueGreen;
  if ComboBox1.ItemIndex=1 then  RedGreen;
  if ComboBox1.ItemIndex=2 then  Line;
  if ComboBox1.ItemIndex=3 then  MColours;
  if ComboBox1.ItemIndex=4 then  DBlueGreen;
  if ComboBox1.ItemIndex=5 then  DRedGreen;
  if ComboBox1.ItemIndex=6 then  Snake1;
  if ComboBox1.ItemIndex=7 then  LRedGreen;
  if ComboBox1.ItemIndex=8 then  YellowCyan;
  if ComboBox1.ItemIndex=9 then  BrightCyanGreen;
  if ComboBox1.ItemIndex=10 then  EscherBlue;
```

```
  if ComboBox1.ItemIndex=11 then  EscherBlueY;
end;



procedure TForm1.Timer1Timer(Sender: TObject);
begin
  if CheckBox3.IsChecked then
    Begin
      ArcDial1.Value:=ArcDial1.Value-RotateA;
    End;
  if CheckBox4.IsChecked then
    Begin
      ArcDial1.Value:=ArcDial1.Value+RotateA;
    End;
end;

procedure TForm1.TrackBar1Change(Sender: TObject);
begin
  RotateA:=Round(TrackBar1.Value);
end;

end.



unit Thread;

interface

uses
  Classes {$IFDEF MSWINDOWS}, Windows {$ENDIF}, Unit1, SysUtils,StrUtils,
FMX.Types;//, Vcl.Graphics;

type
  ImageThread = class(TThread)
  Private
  Var
  FR,FG,FB,ToR,ToG,ToB,FBR,FBG,FBB,ToBR,ToBG,ToBB : Integer;
  IsBlue,IsGreen : Boolean;
  Box1,Box2 : Boolean;
  FName : AnsiString;
  BitMap : TBitMap;
  I: Integer;
  T: Byte;
  PixelRecs: PAlphaColorRecArray;
  Procedure UpdatePicture;
  Procedure CallFile;
  procedure Execute; override;
  protected
```

```
  Public
  Constructor
Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean; Filename : ShortString);
  end;

type
  ImageThreadM = class(TThread)
  Private
  Var
  FR,FG,FB,ToR,ToG,ToB,FBR,FBG,FBB,ToBR,ToBG,ToBB : Integer;
  IsBlue,IsGreen : Boolean;
  Box1,Box2 : Boolean;
  FName : AnsiString;
  BitMap : TBitMap;
    type
      TMyPixelDescriptor = record
      Blue: Byte;
      Green: Byte;
      Red: Byte;
     end;
  PMyPixelArray = ^TMyPixelArray;
  TMyPixelArray = array[0..32767] of TMyPixelDescriptor;
  THMyPictureArray = array of TMyPixelArray;
  Procedure UpdatePictureM;
  Procedure CallFileM;
  procedure Execute; override;
  protected
  Public
  Constructor
Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean; Filename : ShortString);
  end;

implementation

Procedure ImageThreadM.CallFileM;
begin
  BitMap:=TBitMap.CreateFromFile(FName);
  BitMap.LoadFromFile(FName);
end;

Procedure ImageThread.CallFile;
begin
  BitMap:=TBitMap.CreateFromFile(FName);
  BitMap.LoadFromFile(FName);
end;
```

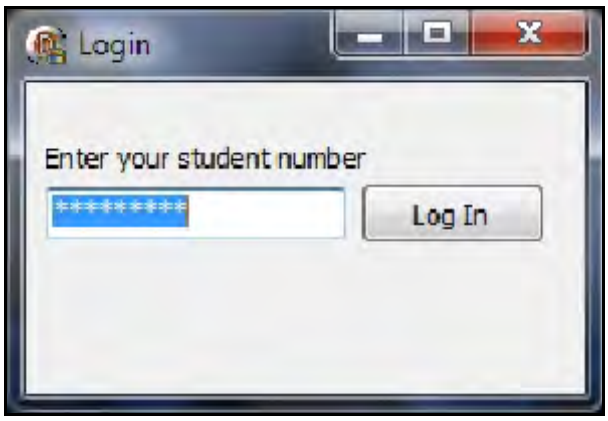
```
constructor
ImageThreadM.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,ATo
BG,
  AToBB : Integer; ABox1,ABox2 : Boolean;Filename : ShortString);
Begin
  FR:= AFR;
  FG:=AFG;
  FB:=AFB;
  ToR:=AToR;
  ToG:=AToG;
  ToB:=AToB;
  FBR:=AFBR;
  FBG:=AFBG;
  FBB:=AFBB;
  ToBR:=AToBR;
  ToBG:=AToBG;
  ToBB:=AToBB;
  Box1:=ABox1;
  Box2:=ABox2;
  FName := FileName;
  FreeOnTerminate:=True;
  Inherited Create(False);
End;

constructor
ImageThread.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToB
G,
  AToBB : Integer; ABox1,ABox2 : Boolean; Filename : ShortString);
Begin
  FR:= AFR;
  FG:=AFG;
  FB:=AFB;
  ToR:=AToR;
  ToG:=AToG;
  ToB:=AToB;
  FBR:=AFBR;
  FBG:=AFBG;
  FBB:=AFBB;
  ToBR:=AToBR;
  ToBG:=AToBG;
  ToBB:=AToBB;
  Box1:=ABox1;
  Box2:=ABox2;
  FName := FileName;
  FreeOnTerminate:=True;
  Inherited Create(False);
End;
```

```
Procedure ImageThreadM.UpdatePictureM;
begin
  Form1.Image1.Bitmap.Canvas.CreateFromBitmap(BitMap);
  AppendStr(FName,'1');
  Bitmap.Free;
end;

Procedure ImageThreadM.Execute;
Var
  PixelRecs: PAlphaColorRecArray;
  I : Integer;
  p : PMyPixelArray;
  x : Integer;
  y : Integer;
  T : Byte;
begin
  BitMap:=TBitMap.CreateFromFile(FName);
  BitMap.LoadFromFile(FName);
  PixelRecs := PAlphaColorRecArray(BitMap.StartLine);
for I := 0 to Bitmap.Width * Bitmap.Height - 1 do
  Begin
    PixelRecs := PAlphaColorRecArray(Bitmap.StartLine);
      with PixelRecs[I] do
        Begin
          if Box2  then
           if ((R<=FBR) AND (G<=FBG) AND (B<=FBB))then //is it Lighter?
            Begin
              R:=ToBR;
              G:=ToBG;
              B:=ToBB;
            End;
          if Box1 then
            if ((R<=FR) AND (G<=FG) AND (B<=FB)) then  //is it darker?
            Begin
              G:=ToG;
              R:=ToR;
              B:=ToB;
            End;
        End;
    End;
  if Terminated  then Exit;
  Synchronize(UpdatePictureM);
end;

Procedure ImageThread.Execute;
Var
  PixelRecs: PAlphaColorRecArray;
  SourceLine, TargetLine: PAlphaColorArray;
  I : Integer;
  p : PMyPixelArray;
```

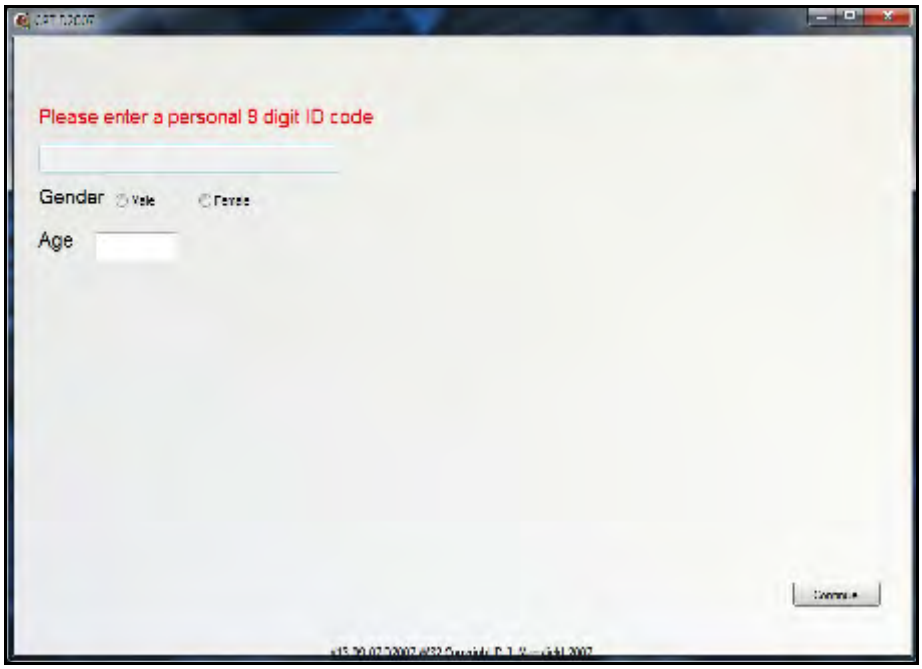
```
  x : Integer;
  y : Integer;
  R,G,B : Integer;
begin
  BitMap:=TBitMap.Create(Form1.Image1.Bitmap.Width,Form1.Image1.Bitmap.Height);
    Bitmap.Assign(Form1.Image1.Bitmap);
  for I := 0 to Bitmap.Width * Bitmap.Height - 1 do
  Begin
    PixelRecs := PAlphaColorRecArray(Bitmap.StartLine);
      with PixelRecs[I] do
       Begin
         if Box2  then
          if ((R<=FBR) AND (G>=FBG) AND (B>=FBB))then //is it blue
           Begin
             R:=ToBR;
             G:=ToBG;
             B:=ToBB;
           End;
         if Box1 then
           if ((R<=FR) AND (G>=FG) AND (B<=FB)) then  //is it green?
           Begin
             G:=ToG;
             R:=ToR;
             B:=ToB;
           End;
       End;
  End;
  if Terminated  then Exit;
  Synchronize(UpdatePicture);
end;

Procedure ImageThread.UpdatePicture;
begin
  Form1.Image1.Visible:=False;
  AppendStr(FName,'1');
  Form1.ImageViewer1.Bitmap:=Bitmap;
  Bitmap.Free;
end;

end.
```

Appendix 1.

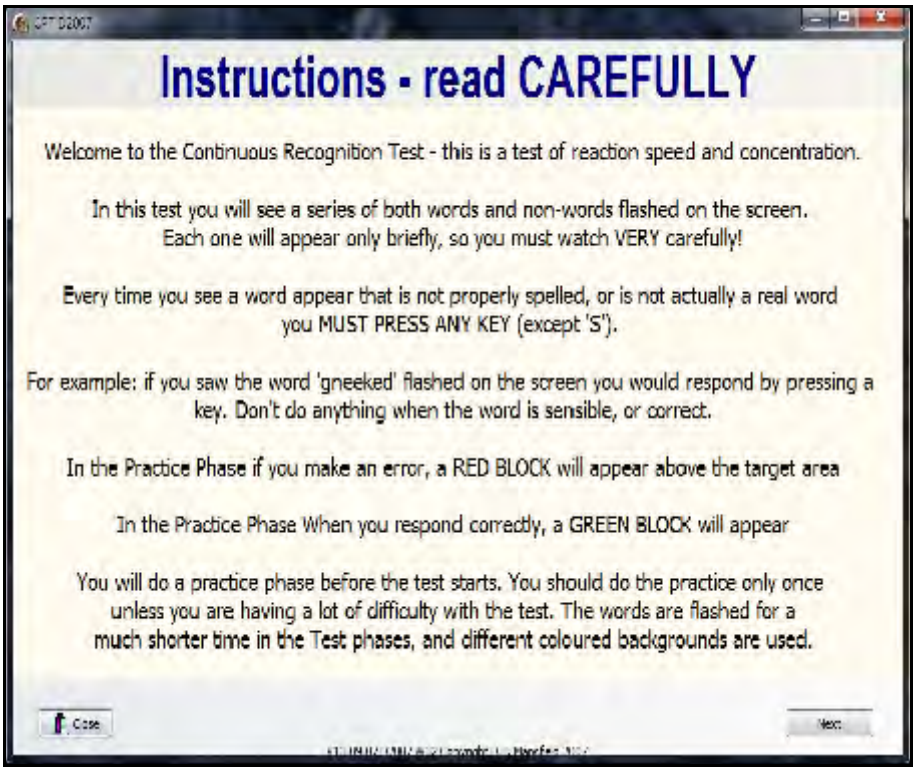
Experiment 1



Experiment 1. *Login Dialogue*



Experiment 1. *Random ID entry dialogue*

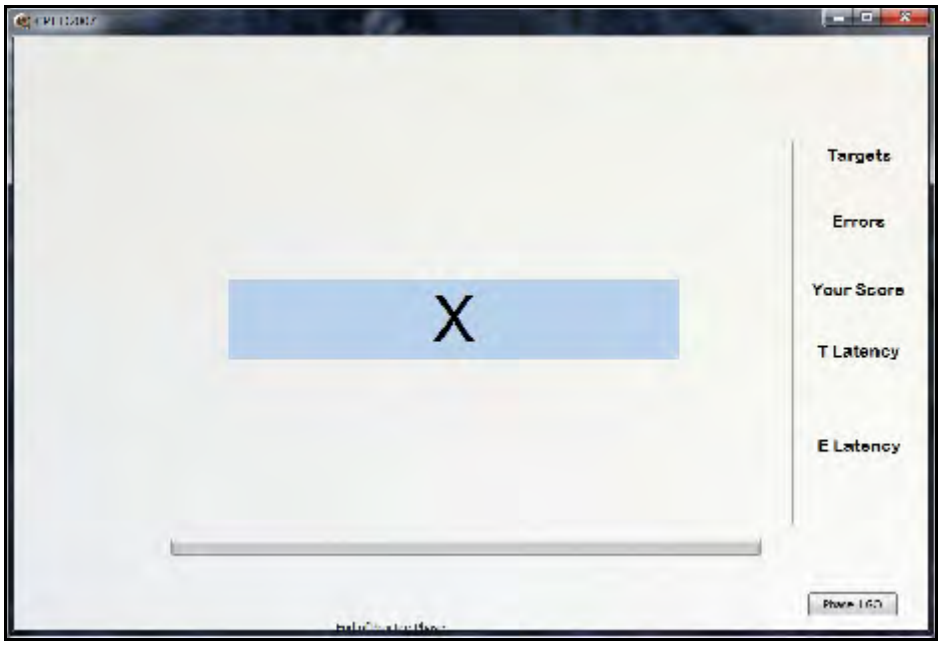


Experiment 1. *Instruction screen*

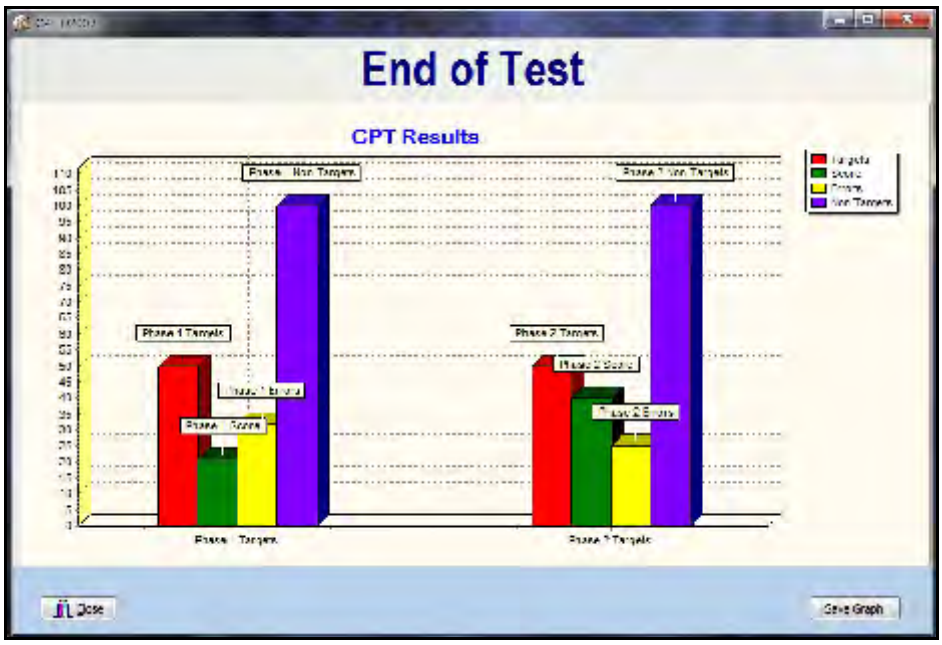
Experiment 1. *Practice phase non-target*

Experiment 1. *Practice phase mask*

Experiment 1. *Practice Phase fixation point.*



Experiment 1. *Practice Phase fixation point*

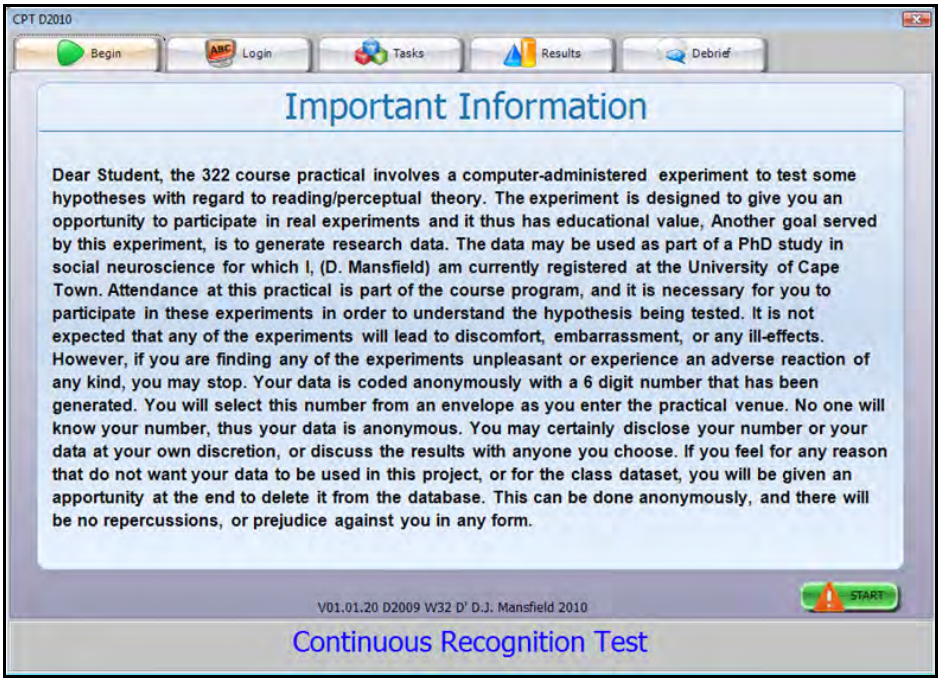


Experiment 1. *Performance Feedback Graphs.*

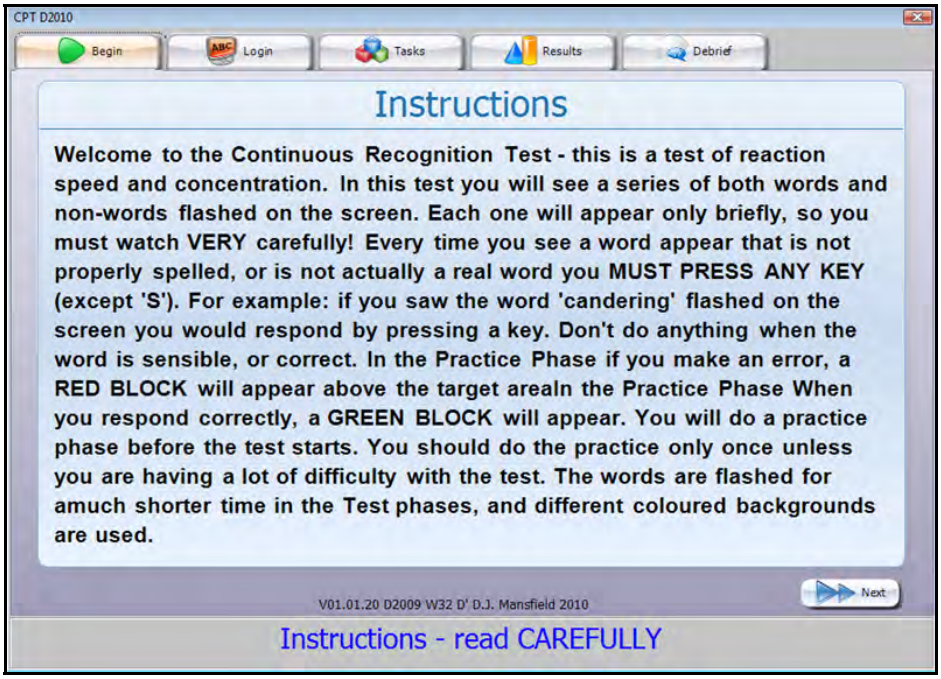
```
BEGIN TEST Phase 1  2007/09/13  03:18:47 PM
ID: 232123 Gender: F Age: 19
Score:  36   Targets: 50 Non Targets: 100  Errors: 12
Mean Target Latency: 532   Mean Error Latency: 403
END TEST PHASE 1   2007/09/13  03:21:22 PM
Actual Testing Time (Phase 1): 00:02:30
Colour: Red Mask used: XXXXXXXXXX Timer 1: 1000 Timer 2: 100 Timer 3: 100


BEGIN TEST phase 2  2007/09/13  03:21:28 PM
ID: 232123 Gender: F Age: 19
Score:  42   Targets: 50 Non Targets: 100  Errors: 5
Mean Target Latency: 534   Mean Error Latency: 427
END TEST PHASE 2   2007/09/13  03:24:01 PM
Actual Testing Time (Phase 2): 00:02:30
Colour: LBlue Mask used: XXXXXXXXXX Timer 1: 1000 Timer 2: 100 Timer 3: 100
```

Experiment 1 example of ASCII Result File

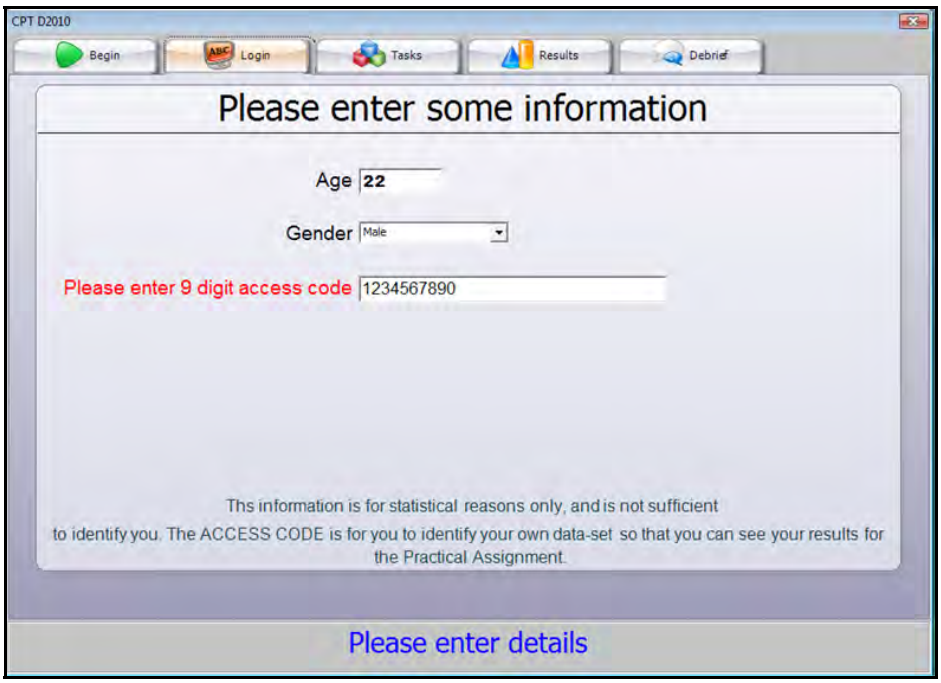
Experiment 2



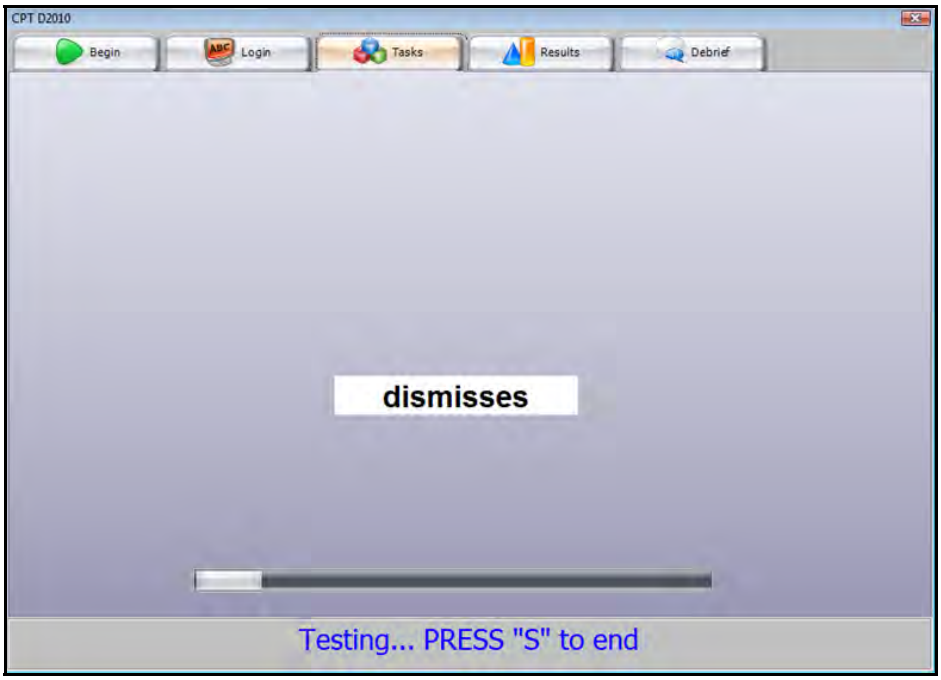
Information Screen: Experiment 2



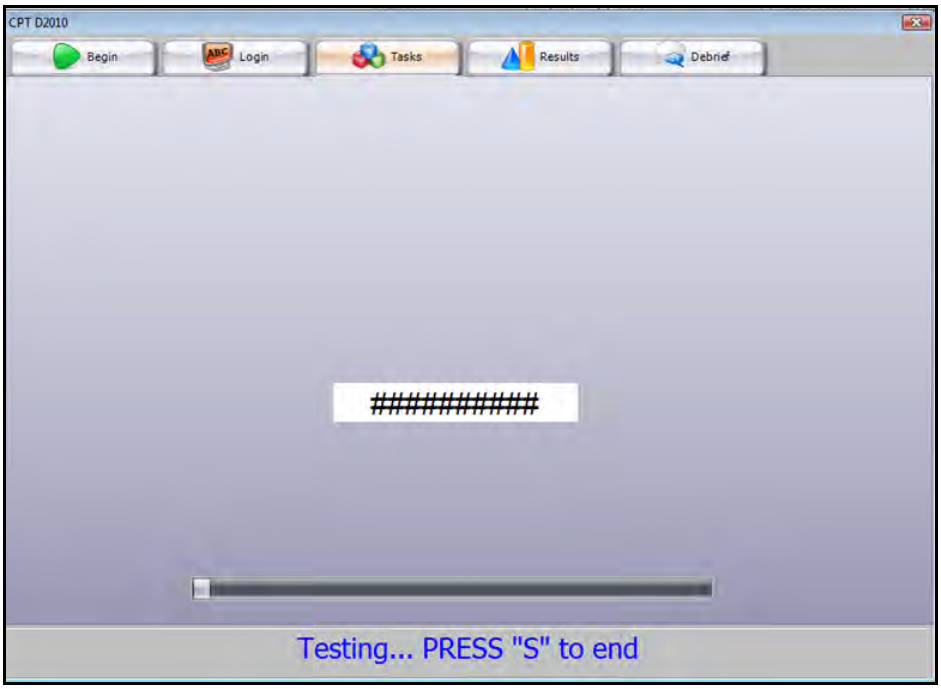
Instruction Screen: Experiment 2



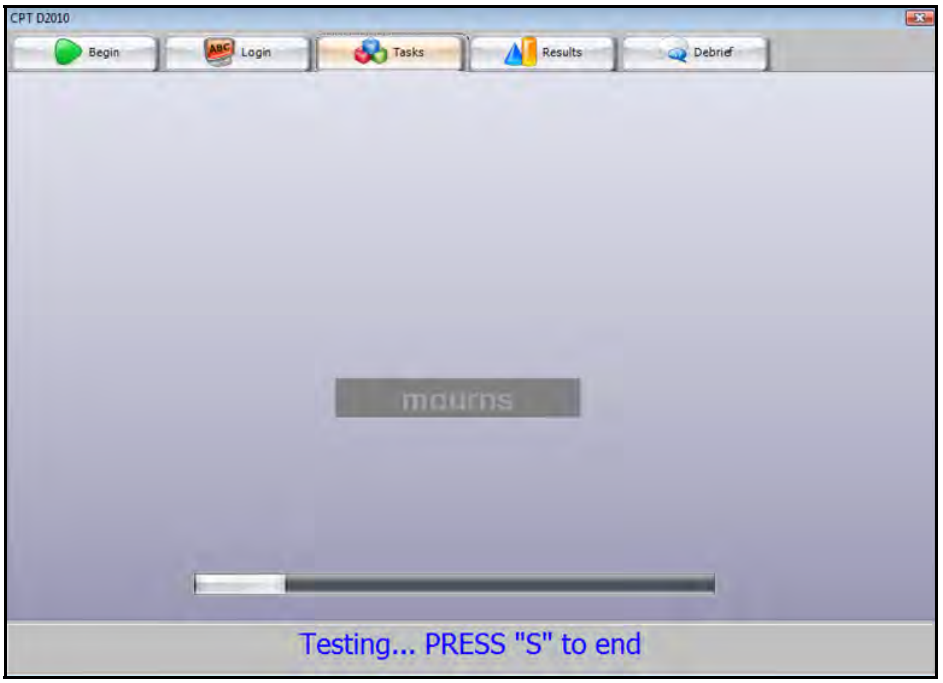
Information Entry Screen: Experiment 2



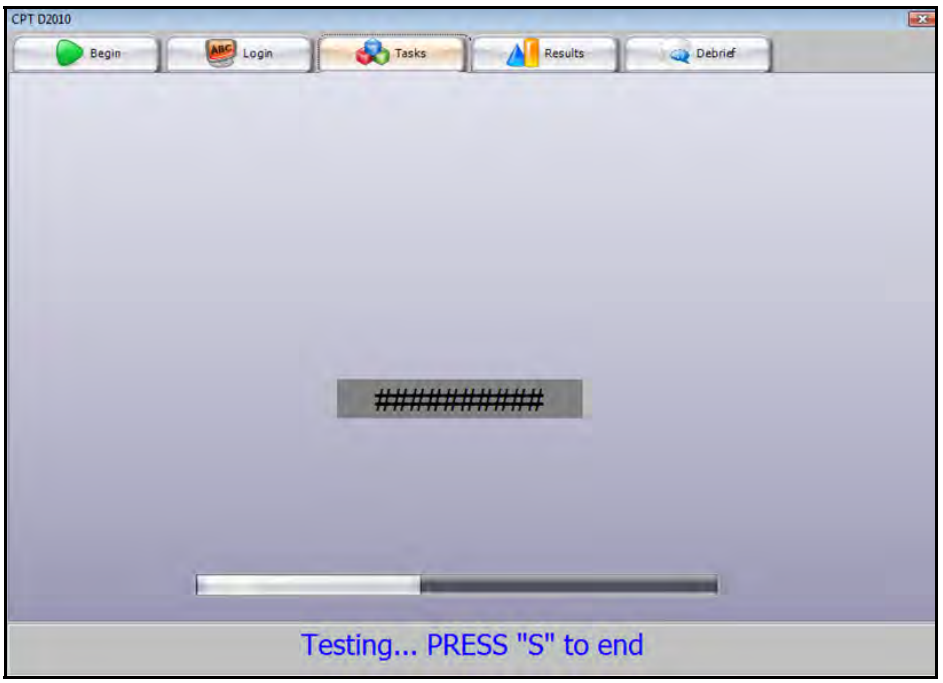
Practice Phase non-target: Experiment 2



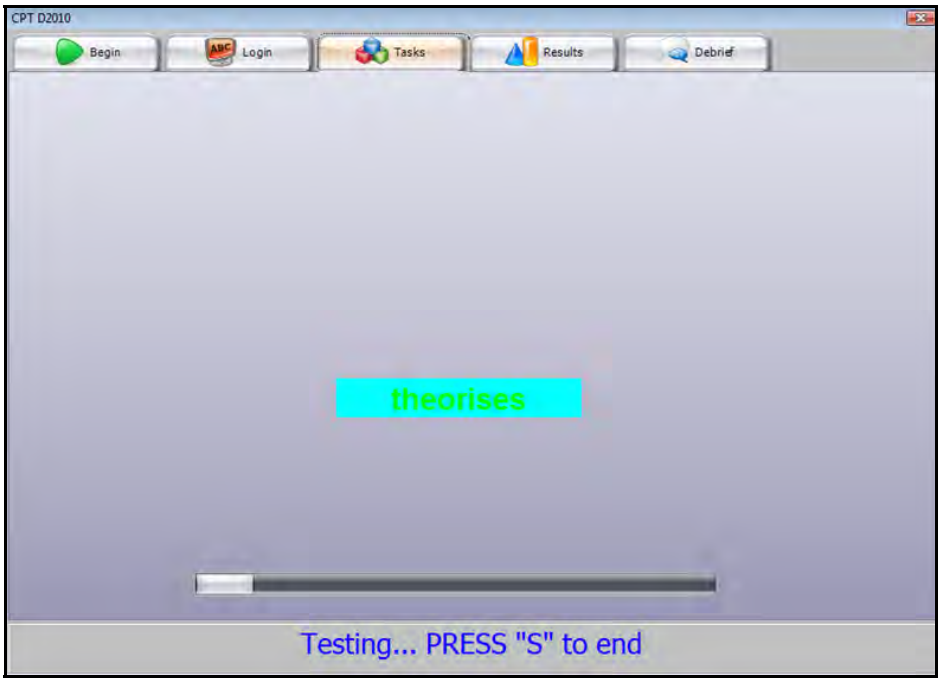
Practice Phase mask: Experiment 2



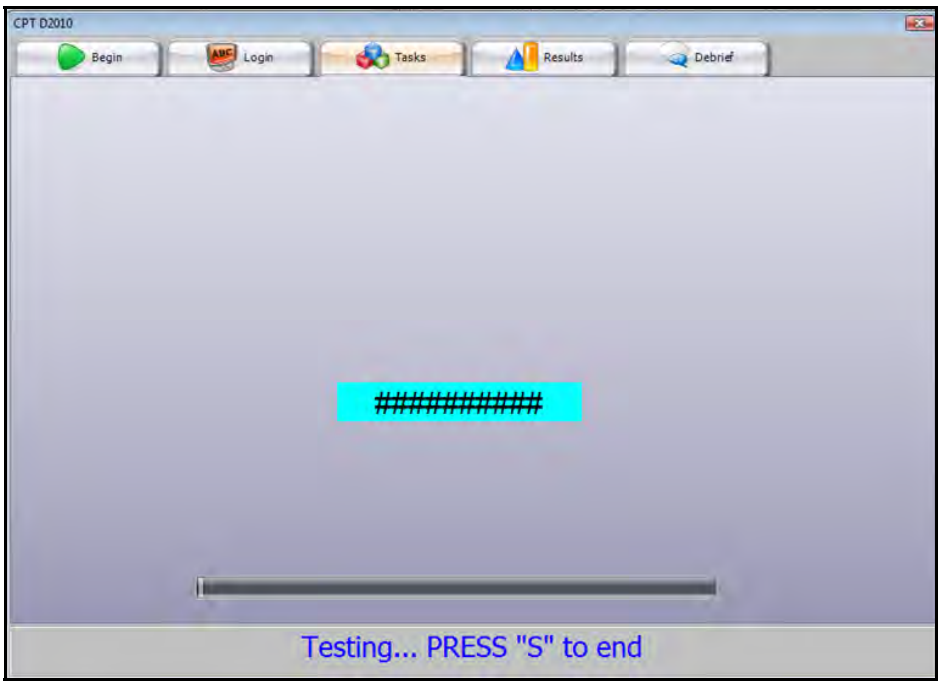
M Phase non-target: Experiment 2



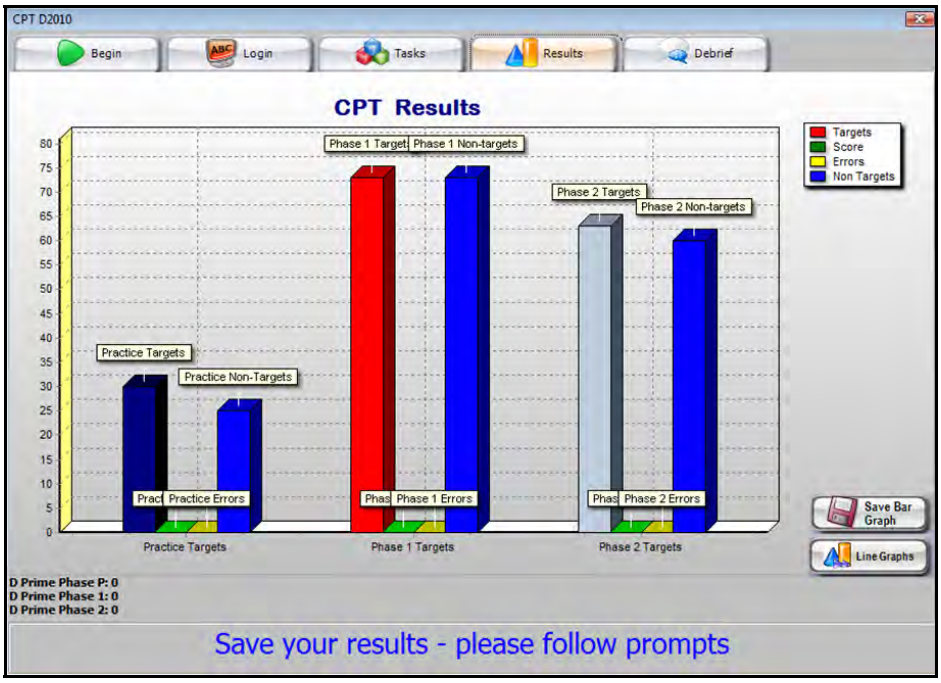
M Phase mask: Experiment 2



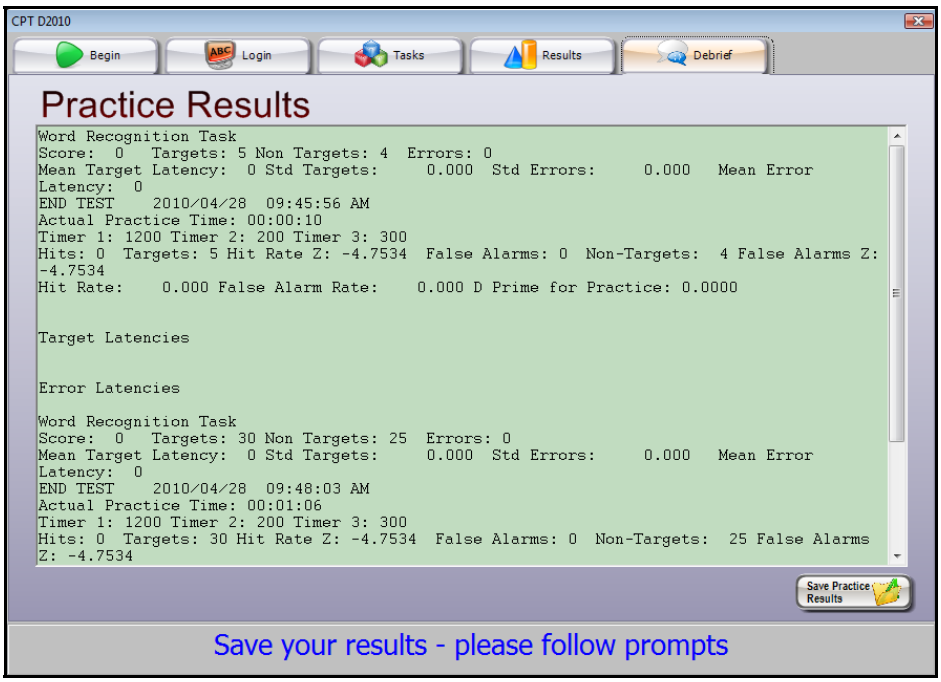
P Phase non-target: Experiment 2



P Phase mask: Experiment 2



Result graphs: Experiment 2



Feedback screen: Experiment 2

BEGIN TEST 2010/04/21 03:24:05 PM
Word Recognition Task
Score: 75 Targets: 100 Non Targets: 100 Errors: 60
Mean Target Latency: 375 Std Targets: 133.814 Std Errors: 174.224 Mean Error Latency: 360
END TEST 2010/04/21 03:28:15 PM
Actual Practice Time: 00:04:01
Timer 1: 1200 Timer 2: 200 Timer 3: 300
Hits: 75 Targets: 100 Hit Rate Z: 0.6745 False Alarms: 0 Non-Targets: 100 False Alarms Z: 0.2533
Hit Rate: 0.750 False Alarm Rate: 0.600 D Prime for Practice: 0.4211

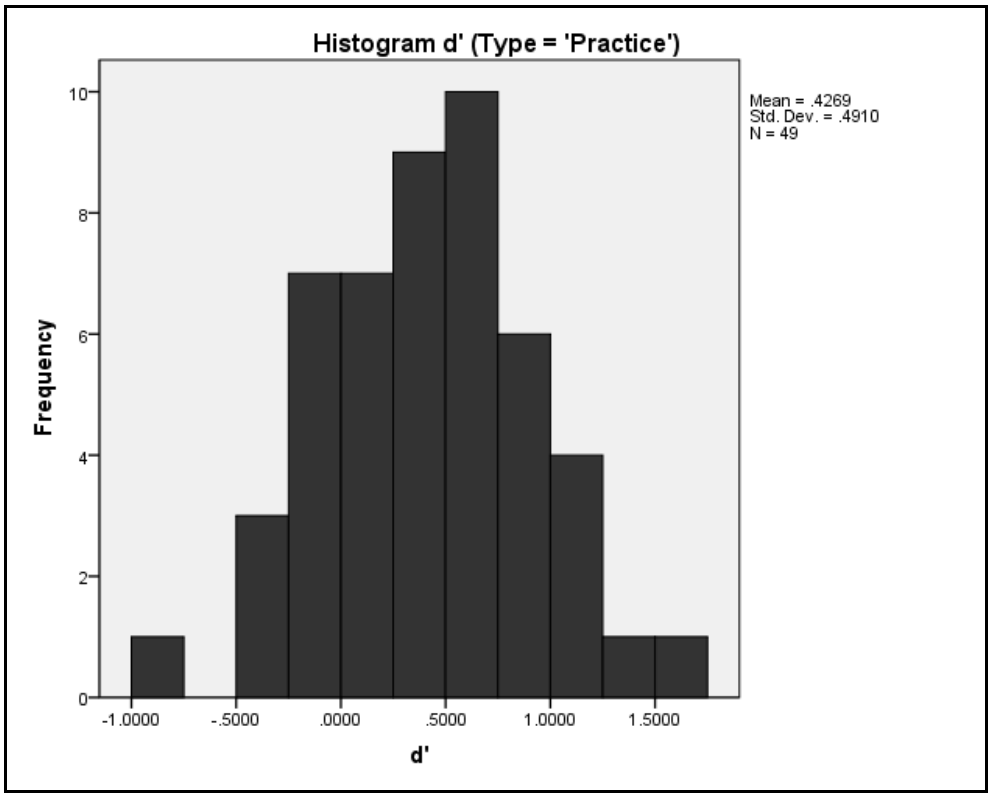
BEGIN TEST Phase 1 2010/04/21 03:28:29 PM
ID: 105955 Gender: F Age: 24
Word Recognition Task

Score: 81 Targets: 100 Non Targets: 100 Errors: 86
Mean Target Latency: 340 Std Targets: 217.652 Std Errors: 201.078 Mean Error Latency: 374
END TEST PHASE 1 2010/04/21 03:32:35 PM
Actual Testing Time (Phase 1): 00:04:00
Colour: Magno Mask used: ########## Timer 1: 1200 Timer 2: 53 Timer 3: 300
Hits: 81 Targets: 100 Hit Rate Z: 0.8779 False Alarms: 86 Non-Targets: 100 False Alarms Z: 1.0803
Hit Rate: 0.810 False Alarm Rate 0.860 D Prime for Phase 1: -0.2024

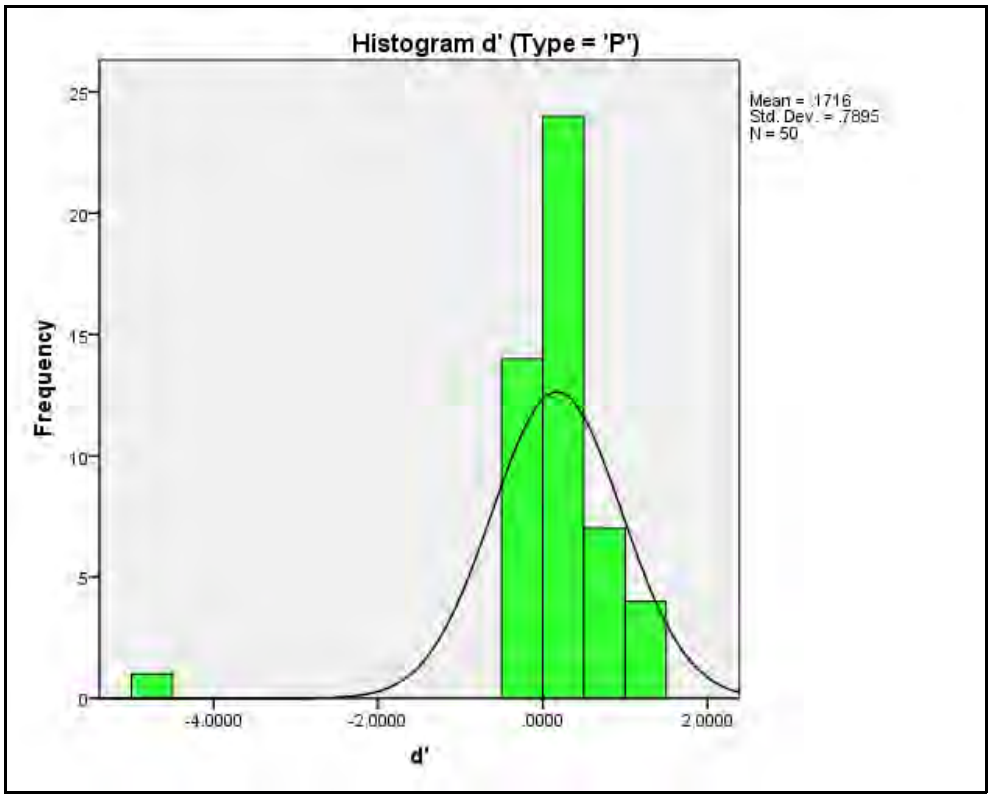
BEGIN TEST phase 2 2010/04/21 03:32:47 PM
ID: 105955 Gender: F Age: 24
Word Recognition Task

Score: 61 Targets: 100 Non Targets: 100 Errors: 66
Mean Target Latency: 480 Std Targets: 335.880 Std Errors: 309.063 Mean Error Latency: 456
END TEST PHASE 2 2010/04/21 03:36:54 PM
Actual Testing Time (Phase 2): 00:04:00
Colour: ParvoCG Mask used: ########## Timer 1: 1200 Timer 2: 53 Timer 3: 300
Hits: 61 Targets: 100 Hit Rate Z: 0.2793 False Alarms: 66 Non-Targets: 100 False Alarms Z: 0.4125
Hit Rate: 0.610 False Alarm Rate 0.660 D Prime for Phase 2: -0.1331

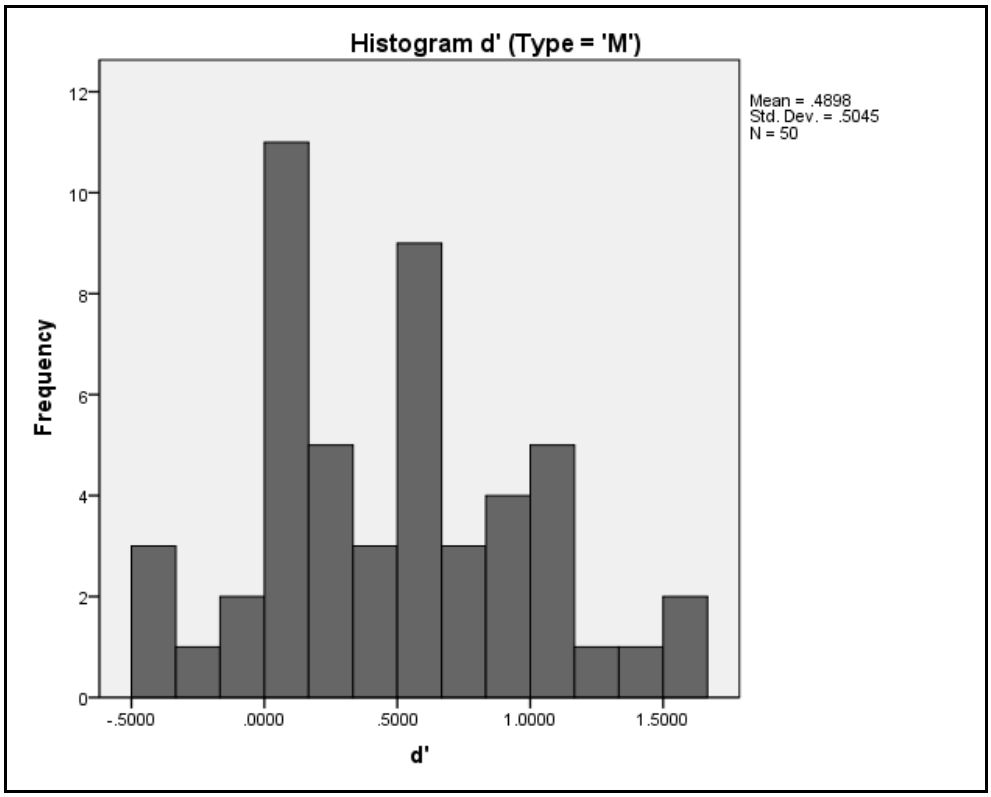
Example result file: Experiment 2



Distribution of d' for Practice Phase



Distribution of d' for 'P' Phase



Distribution of d' for 'M' Phase

Tests of Normality

		Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Type	Statistic	df	Sig.	Statistic	df	Sig.
Dprime	BW	.070	49	.200*	.992	49	.977
	M	.127	50	.041	.972	50	.290
	P	.261	50	.000	.584	50	.000

a. Lilliefors Significance Correction
*. This is a lower bound of the true significance.

Tests of normality of distribution

Experiment 3

10	Duration to initialise fixation '+', load and size pictures
200	Exposure interval during which stimuli were presented and scaled to max. 300 pixels.
500	Exposure window (Stimulus hidden after this period, although responses could be made for the remainder of the total trial, i.e. 1500 ms.)
enabled-	Mask disabled
Magno	M Exposure condition
ParvoCG	P Exposure condition
0,255,255	not used
255,0,0	not used
5	scaling interval (ms)
overide-	Overide of exposure time disabled
random	Random ordering of P and M phase enabled
enabled	Practice feedback (enabled)
enabled-	Main trials feedback (disabled)
calibrate-	Calibration disabled
Object Discrimination Task	Title
Graphics	Enables procedure which looks up graphics files
Panel-	Result panel disabled after trials

Runtime settings for Experiment 3

This is a test of concentration, speed and how fast you can recognise objects.

In this test you will see a series of pictures presented on the screen.
When each picture is presented, it will appear small, and then get bigger quite quickly.

You have to decide if the object you see is BIGGER or SMALLER than a SHOEBOX

So when you recognise that the object is BIGGER than a shoebox (in real life, of course)
you must respond by PRESSING A KEY (ANY key EXCEPT 'S').

For example: if you see a picture of a SHIP you must PRESS A KEY because a SHIP is bigger than a shoebox!

BUT, if you see an object that is smaller then a shoebox, like a PEN, don't do anything.

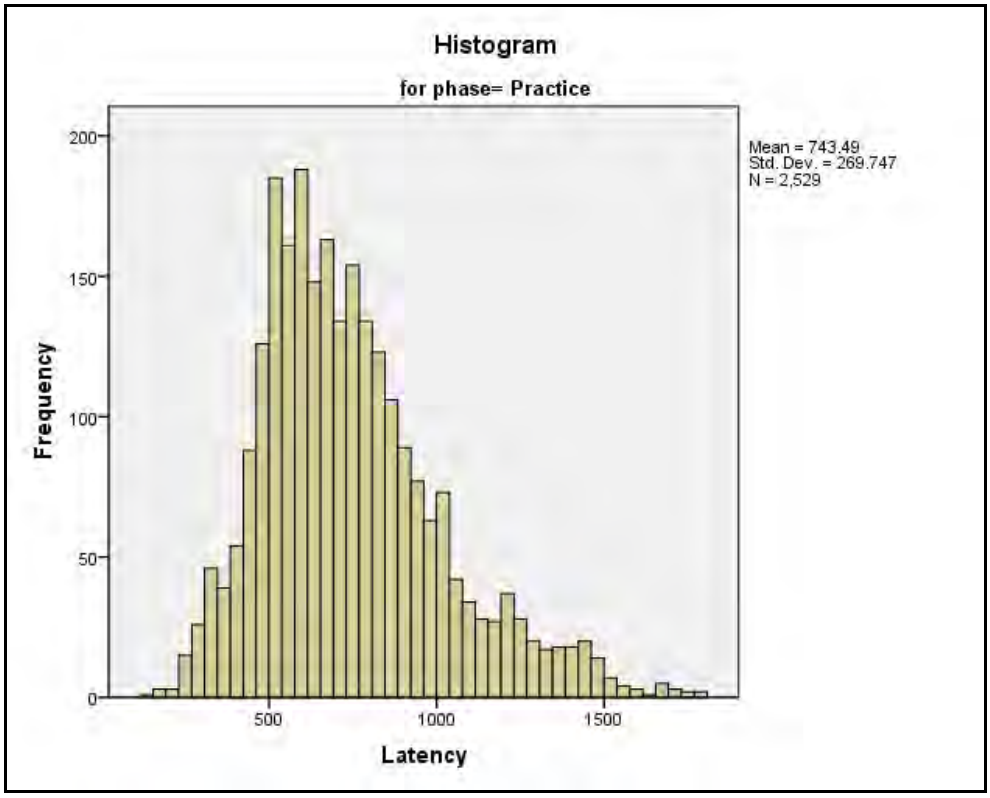
In the Practice Phase: if you make an mistake, a RED LIGHT will come on

When your response is correct, a GREEN LIGHT will appear

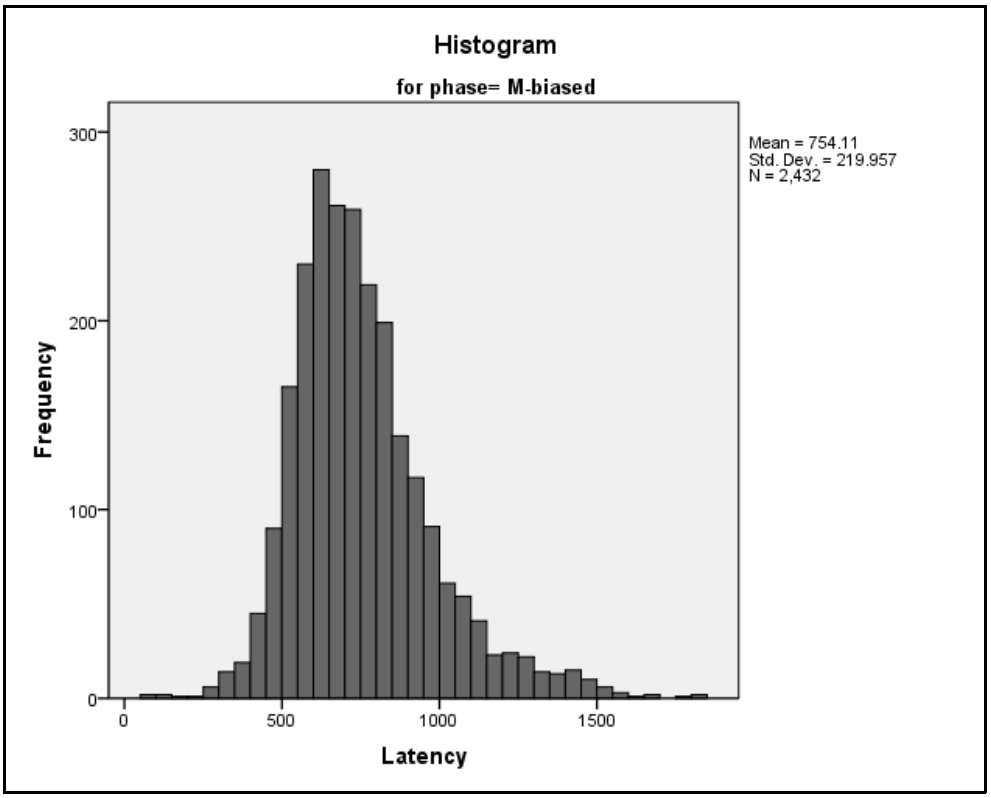
You will do a practice phase before the test starts.

THE MOST IMPORTANT THING IS TO RESPOND QUICKLY! AS SOON AS YOU RECOGNISE THE OBJECT, YOU MUST RESPOND.

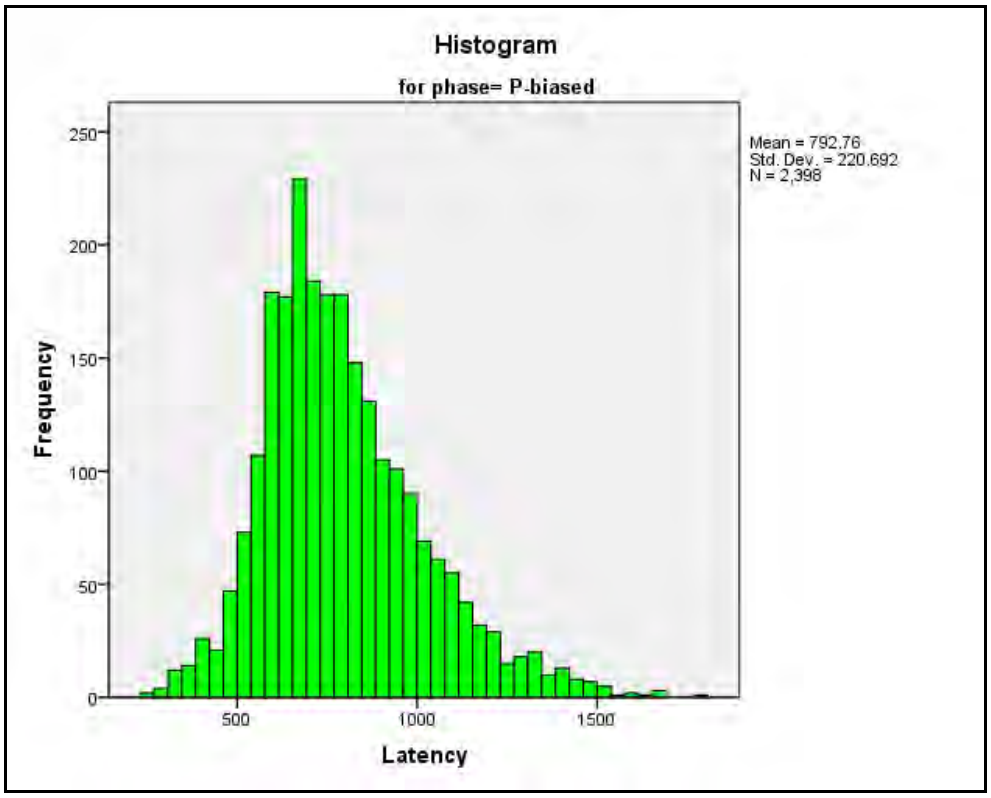
Runtime settings for Experiment 3



Experiment 3: Distribution of latency scores for Practice Phase.

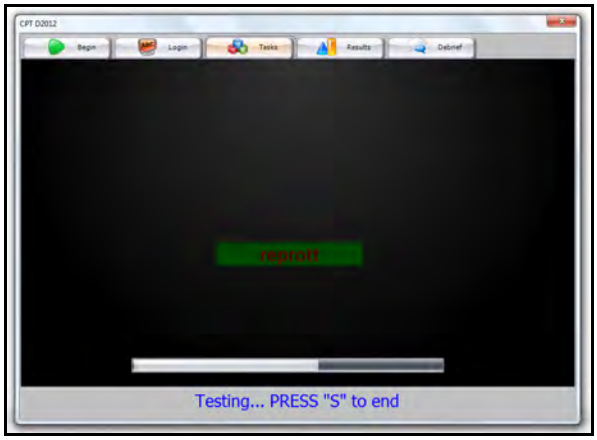


Experiment 3: Distribution of latency scores for M-biased Phase.

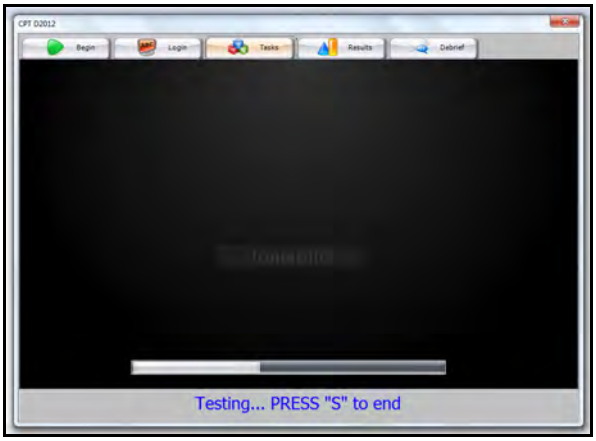


Experiment 3: Distribution of latency scores for P-biased Phase.

Experiment 4



Experiment 4: P Phase non-target



Experiment 4: M Phase non-target

BEGIN TEST 2011/05/03 01:16:19 PM
Word Recognition Task
Score: 69 Targets: 100 Non Targets: 100 Errors: 30
Mean Target Latency: 526 Std Targets: 152.502 Std Errors: 158.649 Mean Error Latency: 501
END TEST 2011/05/03 01:20:24 PM
Actual Practice Time: 00:04:01
Timer 1: 1200 Timer 2: 200 Timer 3: 300
Hits: 69 Targets: 100 Hit Rate Z: 0.4959 False Alarms: 0 Non-Targets: 100 False Alarms Z: -0.5244
Hit Rate: 0.690 False Alarm Rate: 0.300 D Prime for Practice: 1.0203

BEGIN TEST Phase 1 2011/05/03 01:20:32 PM
ID: 131622 Gender: F Age: 21
Word Recognition Task

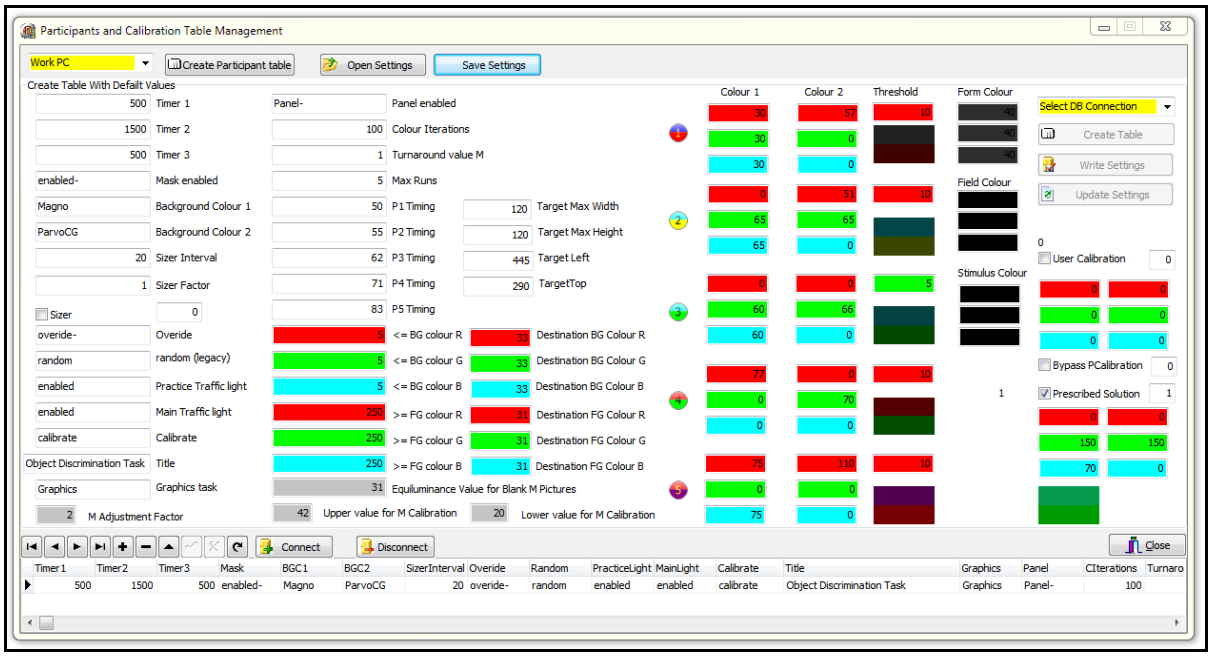
Score: 77 Targets: 100 Non Targets: 100 Errors: 24
Mean Target Latency: 614 Std Targets: 162.728 Std Errors: 181.958 Mean Error Latency: 607
END TEST PHASE 1 2011/05/03 01:24:36 PM
Actual Testing Time (Phase 1): 00:04:00
Colour: 0,70,0,77,0,0, Mask used: ########## Timer 1: 1200 Timer 2: 70 Timer 3: 300
Hits: 77 Targets: 100 Hit Rate Z: 0.7388 False Alarms: 24 Non-Targets: 100 False Alarms Z: -0.7063
Hit Rate: 0.770 False Alarm Rate 0.240 D Prime for Phase 1: 1.4451

BEGIN TEST phase 2 2011/05/03 01:24:39 PM
ID: 131622 Gender: F Age: 21
Word Recognition Task

Score: 92 Targets: 100 Non Targets: 100 Errors: 29
Mean Target Latency: 561 Std Targets: 116.975 Std Errors: 163.353 Mean Error Latency: 558
END TEST PHASE 2 2011/05/03 01:28:47 PM
Actual Testing Time (Phase 2): 00:04:00
Colour: 30,30,30,50,50,50, Mask used: ########## Timer 1: 1200 Timer 2: 70 Timer 3: 300
Hits: 92 Targets: 100 Hit Rate Z: 1.4051 False Alarms: 29 Non-Targets: 100 False Alarms Z: -0.5534
Hit Rate: 0.920 False Alarm Rate 0.290 D Prime for Phase 2: 1.9585

Experiment 4: Example Result File shown in colours which approximate the experiment

Replication



Replication experiment: *Remote calibration and setting application.*

The application creates the settings table on the remote server, and then writes the values to the table. This is analagous to the settings file from previous experiments. The Timer 1 – 3 fields set the timing of the experiment to run in the same way as that of Kveraga *et al.* The 'Practice' and 'Main Traffic Light' is the indicator which displays when the response is correct or not. The 'Panel enabled' field enables, or disables the display panel which shows constantly updated scores and performance. It was disabled to remove distractions, and its main purpose in the design of the program was to check scoring, and do run-time debugging. The number of iterations for each colour trial of the colour calibration phase is set in the field labeled 'Colour Iterations'. The 'Turnaround value M' field sets the initial calibration criterion for cycling the values of the luminance stimuli. 'Max Runs' is the number of colour solutions to be tested – the columns next to the dots and their colour definitions under 'Colour 1' and 'Colour 2'.

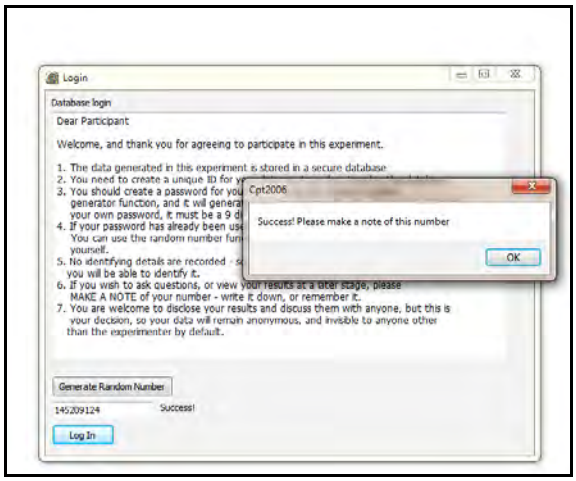
There are 5 fields 'P1-Timing...' - '...P5 Timing' which set the timer values to implement the frequency ranges for the flicker trials. 'P1 Timing' is set to 50 ms, because a timing component is set to trigger every 50 ms (1000 DIV 50) = 20 Hz. These setting correspond respectively to 20, 18, 16, 14, 12 Hz. The 9 panels below set the source and destination values for the luminance transformations. These are implemented in the algorithm snippet in Table 6 (LVM is the Lower Value for M; UVM is the Upper Value for M).

```
Function CheckFloor: Boolean;
Begin
  if (DR<LVM) AND (DG<LVM) AND
(DB<LVM) then
  Result:=True
    Else
      Result:=False;
End;

Function CheckRoof: Boolean;
Begin
  if (DR>UVM) AND (DG>UVM) AND
(DB>UVM) then
  Result:=True
    Else
      Result:=False;
```

Luminance calibration upper and lower check functions

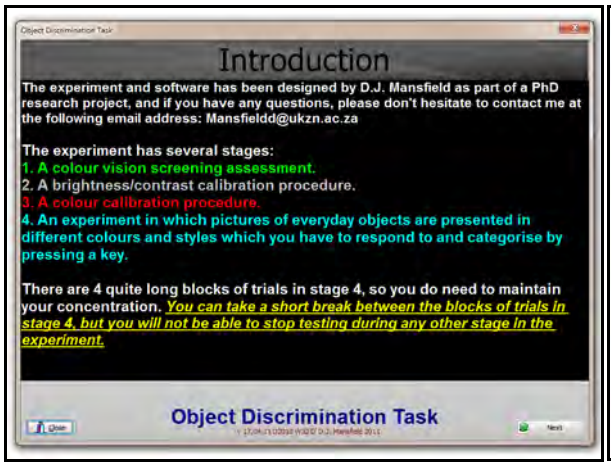
The 'Colour 1' and 'Colour 2' columns are fields which can be altered to set the different colour solution pairs. The 'Threshold' field shows the range in which the defined colour will cycle. In the top row, the threshold is set at 10, and the red value for 'Colour 2' is set to 57. 'Colour 1' is held constant, and the display flickers between 'Colour 1' and 'Colour 2' +/- 10.

There are fields for adjusting the 'Form Colour', back-ground 'Field Colour' for the icon surround, and setting the 'Stimulus Colour' to a constant. The 'User Calibration' when enabled, allows the user to review and fine-tune their calibration settings after the entire calibration procedure is complete. They would be presented with their averaged solutions, with the flickering enabled, and they could be adjusted with a high degree of precision. However, due to the sheer complexity of the procedure, this was disabled. If 'Bypass PCalibration' was checked, the entire colour calibration procedure would be disabled, and the default values would be enabled by checking 'Prescribed Solution'. After a lot of initial experimenting, I decided to enable the colour calibration procedure, but disable the 'User Calibration' procedure, and prescribe a colour solution. See Figure 29 for the settings that were used.

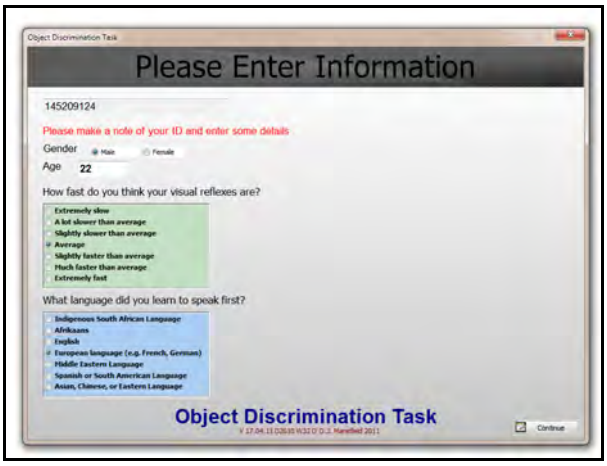


Successful Login.

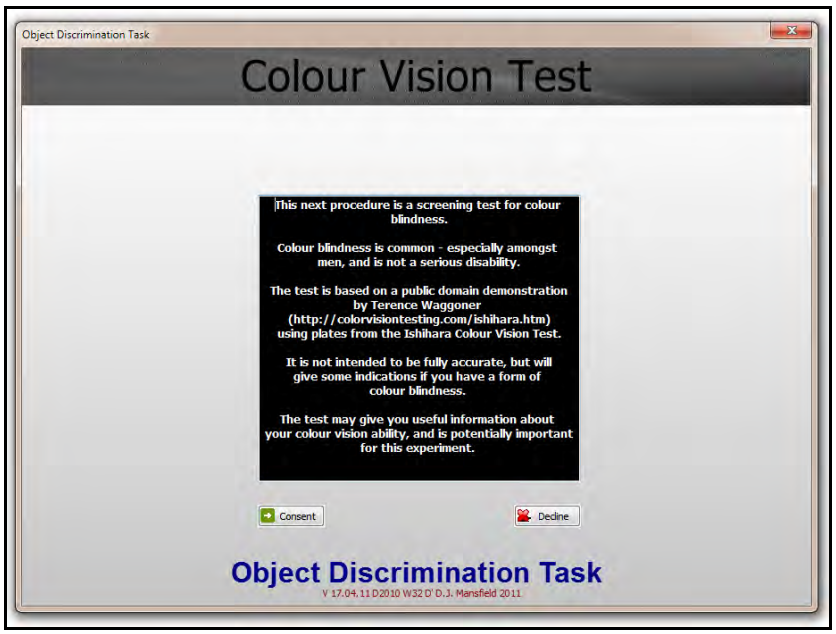
Welcome screen.



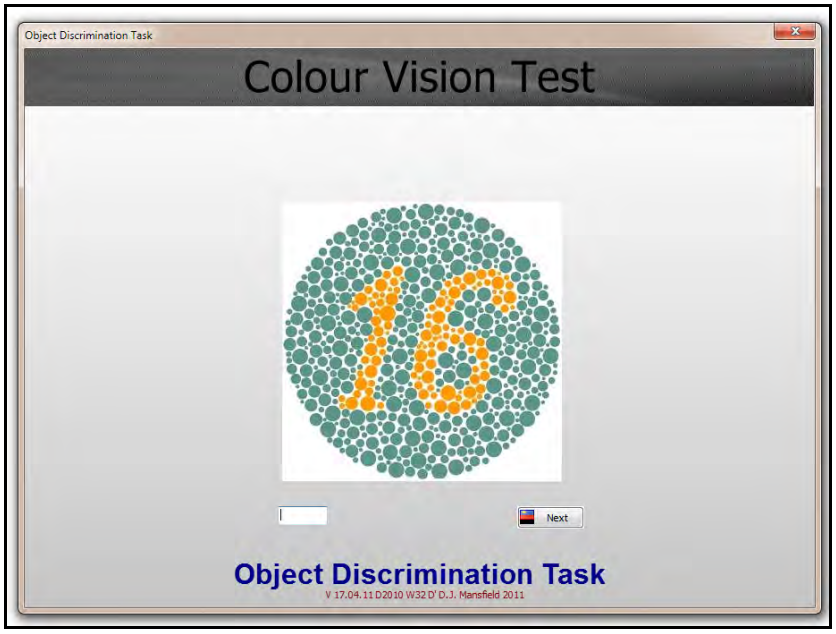
Introduction screen.



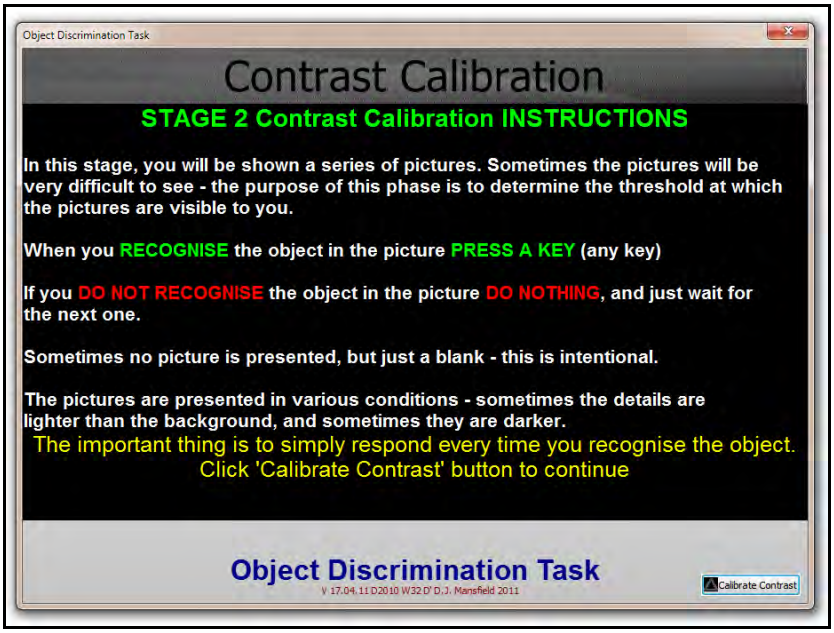
Information screen.



Introduction to Colour Vision Test - consent/decline screen



Colour Vision Test Item.

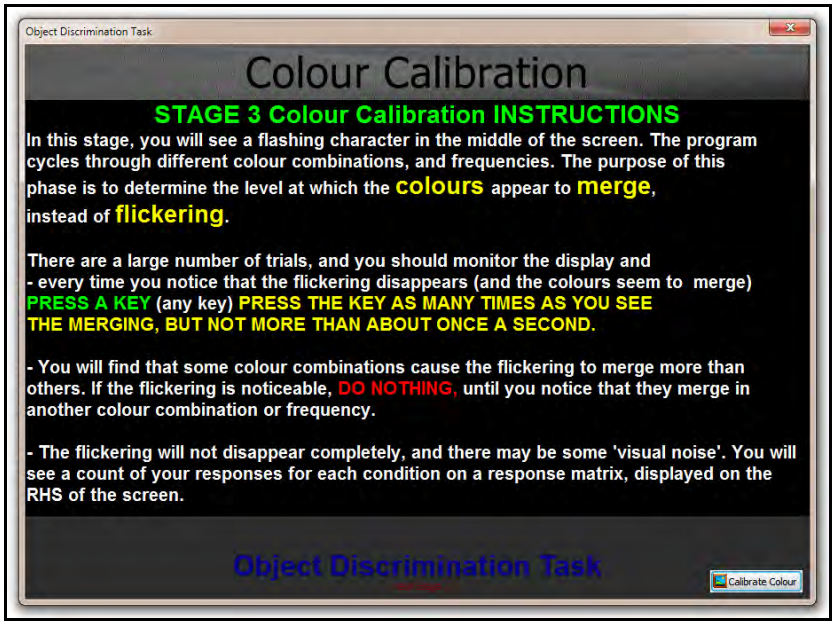


Contrast Calibration Instruction Screen.

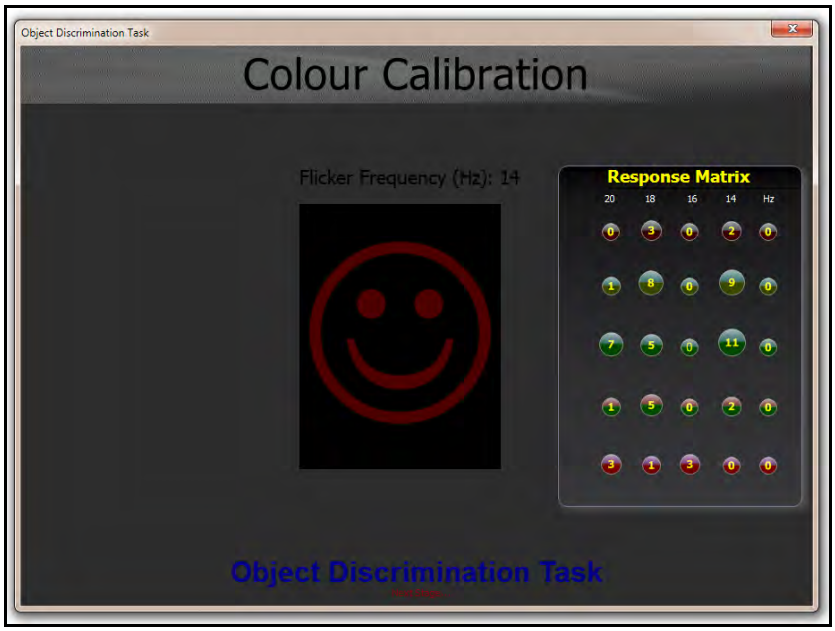
Contrast calibration trial example.

Feedback after correct response.

Feedback after incorrect response.



Colour Calibration instruction screen.

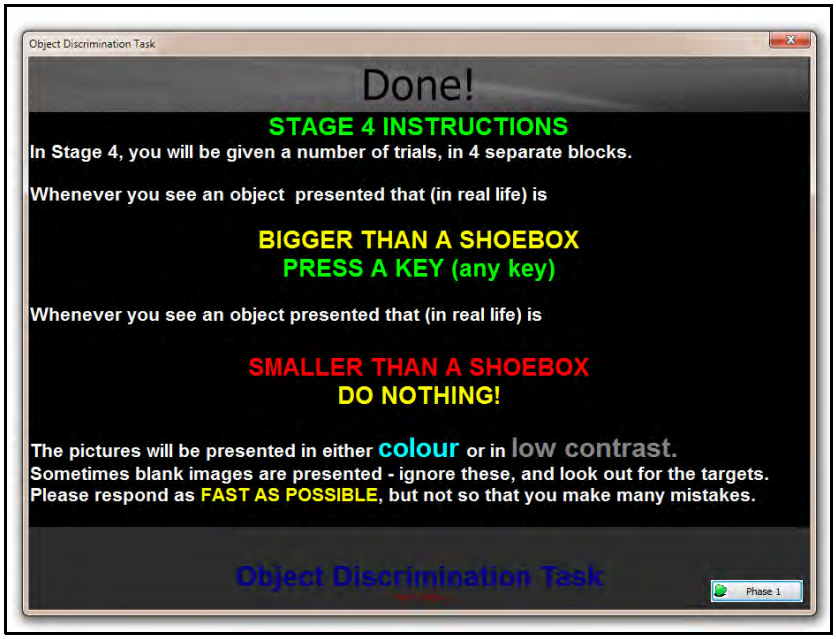


Colour calibration at 14 Hz, colour solution 4.

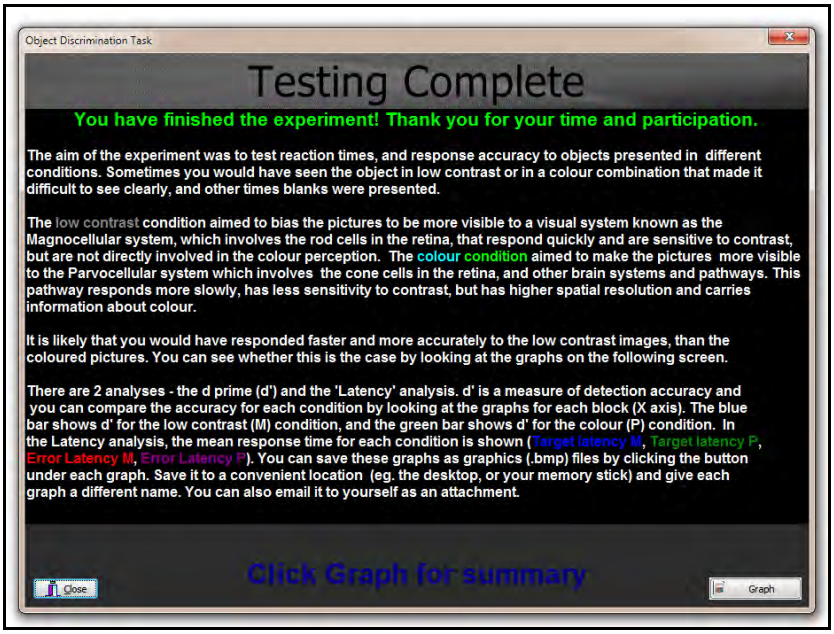
```
Word list 1  2012/08/11  05:36:53 PM
ID: 145209124 Gender: M Age: 22

Ladder      0  1
Scorn       1  0
Safety      1  0
SSciss      1  0
Null        1  0
SKey        1  1
Null        1  1
Lion1BW     0  0
LShtl       0  1
SWirebd     1  1
LDog07      0  0
LZebra      0  1
Null        1  1
SMuffins    1  1
SBTop       1  1
Null        1  1
LGuitar     0  0
SKeys       1  0
LHeli1      0  0
Null        1  0
Null        1  1
SSpoon      1  1
LPiano_2    0  0
Null        1  1
Null        1  1
LPram       0  1
Null        1  0
SPolices    1  1
Null        1  1
SAcorn      1  0
Null        1  1
Scart       1  0
Null        1  0
Null        1  1
Null        1  1
Null        1  0
Lplane1     0  0
SMaskT      1  0
LPiano      0  1
Null        1  1
Null        1  0
SGrout      1  0
SHolly      1  0
LMntbike    0  0
LTudor      0  1
Null        1  1
SPasha      1  0
LBikeG      0  1
LHeli2      0  1
LWkbench    0  1
Null        1  0
SSpan       1  0
LCarriage   0  0
Swatch      1  0
SBroc       1  1
SaltP       1  1
LCow        0  0
Lifeboat    0  1
LSewing     0  0
Null        1  1
```

Replication *Example item list.*

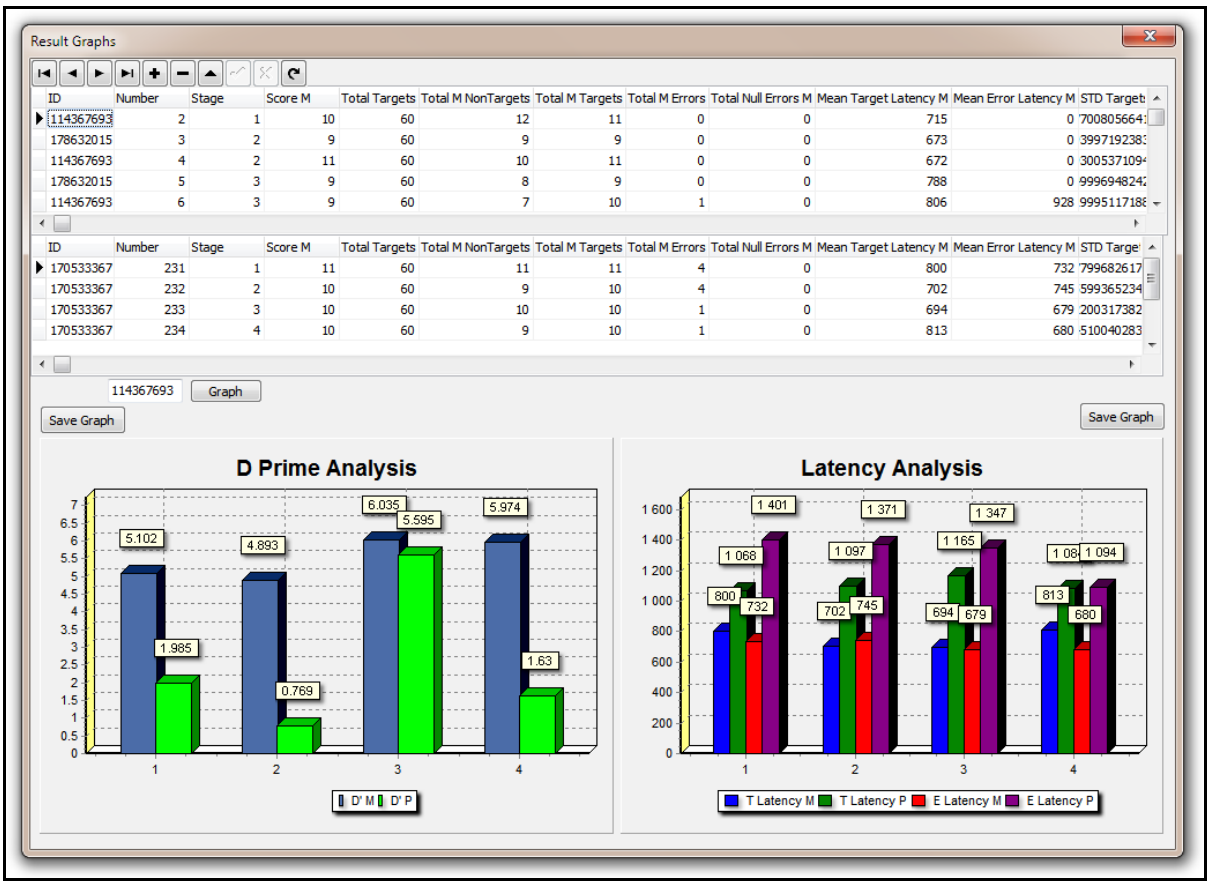
A 'Null' can be displayed in either the 'M' or 'P' condition, and the program selects any item at random, and transforms it to equiluminance for that condition.



Instructions for testing stages.



Concluding information screen.



Result graph.

Number	Timing	Colour Solution	C1 Red	C1 Green	C1 Blue	C2 Red	C2 Green	C2 Blue	Key Pressed
1	50	1	30	30	30	52	0	0	1
2	50	1	30	30	30	52	0	0	1
3	50	2	0	65	65	48	65	0	1
4	50	2	0	65	65	51	65	0	1
5	50	3	0	60	60	0	71	0	1
6	50	3	0	60	60	0	68	0	1
7	50	4	72	0	0	0	70	0	1
8	50	4	69	0	0	0	70	0	1
9	50	5	75	0	75	110	0	0	1
10	55	1	30	30	30	51	0	0	1
11	55	1	30	30	30	48	0	0	1
12	55	1	30	30	30	50	0	0	1
13	55	2	0	65	65	45	65	0	1
14	55	3	0	60	60	0	63	0	1
15	55	3	0	60	60	0	69	0	1
16	55	4	71	0	0	0	70	0	1
17	55	4	79	0	0	0	70	0	1
18	55	4	81	0	0	0	70	0	1
19	62	1	30	30	30	53	0	0	1
20	62	1	30	30	30	51	0	0	1
21	62	2	0	65	65	48	65	0	1
22	62	4	73	0	0	0	70	0	1
23	62	4	83	0	0	0	70	0	1
24	62	4	84	0	0	0	70	0	1
25	71	1	30	30	30	55	0	0	1
26	71	1	30	30	30	65	0	0	1
27	71	2	0	65	65	48	65	0	1
28	71	2	0	65	65	57	65	0	1
29	71	3	0	60	60	0	63	0	1
30	71	4	73	0	0	0	70	0	1
31	71	4	81	0	0	0	70	0	1

Calibration data from one case: response 1 - 31.

Number	Timing	Colour Solution	C1 Red	C1 Green	C1 Blue	C2 Red	C2 Green	C2 Blue	Key Pressed
27	71	2	0	65	65	48	65	0	1
28	71	2	0	65	65	57	65	0	1
29	71	3	0	60	60	0	63	0	1
30	71	4	73	0	0	0	70	0	1
31	71	4	81	0	0	0	70	0	1
32	71	4	72	0	0	0	70	0	1
33	71	4	82	0	0	0	70	0	1
34	71	4	85	0	0	0	70	0	1
35	71	4	70	0	0	0	70	0	1
36	83	1	30	30	30	48	0	0	1
37	83	1	30	30	30	56	0	0	1
38	83	1	30	30	30	49	0	0	1
39	83	2	0	65	65	59	65	0	1
40	83	2	0	65	65	44	65	0	1
41	83	2	0	65	65	43	65	0	1
42	83	3	0	60	60	0	67	0	1
43	83	3	0	60	60	0	71	0	1
44	83	3	0	60	60	0	67	0	1
45	83	4	72	0	0	0	70	0	1
46	83	4	70	0	0	0	70	0	1
47	83	4	76	0	0	0	70	0	1
48	83	4	82	0	0	0	70	0	1
49	83	4	84	0	0	0	70	0	1
50	83	4	75	0	0	0	70	0	1
51	83	4	68	0	0	0	70	0	1
52	83	4	72	0	0	0	70	0	1
53	83	4	78	0	0	0	70	0	1
54	83	4	85	0	0	0	70	0	1
55	83	4	76	0	0	0	70	0	1
56	83	4	68	0	0	0	70	0	1
57	83	5	75	0	75	110	0	0	1

Calibration data from one case: response 27 - 57.

Number	Timing	Colour Solution	C1 Red	C1 Green	C1 Blue	C2 Red	C2 Green	C2 Blue	Key Presse
1	50	1	30	30	30	55	0	0	
2	50	1	30	30	30	55	0	0	
3	50	1	30	30	30	63	0	0	
4	50	1	30	30	30	60	0	0	
5	50	1	30	30	30	53	0	0	
6	50	1	30	30	30	64	0	0	
7	50	1	30	30	30	58	0	0	
8	50	2	0	65	65	47	65	0	
9	50	2	0	65	65	45	65	0	
10	50	2	0	65	65	56	65	0	
11	50	2	0	65	65	54	65	0	
12	50	2	0	65	65	45	65	0	
13	50	2	0	65	65	60	65	0	
14	50	2	0	65	65	44	65	0	
15	50	3	0	60	60	0	67	0	
16	50	3	0	60	60	0	67	0	
17	50	3	0	60	60	0	65	0	
18	50	3	0	60	60	0	68	0	
19	50	3	0	60	60	0	65	0	
20	50	3	0	60	60	0	67	0	
21	50	3	0	60	60	0	65	0	
22	50	3	0	60	60	0	70	0	
23	50	3	0	60	60	0	63	0	
24	50	5	75	0	75	103	0	0	
25	50	5	75	0	75	104	0	0	
26	50	5	75	0	75	110	0	0	
27	50	5	75	0	75	118	0	0	
28	50	5	75	0	75	114	0	0	
29	50	5	75	0	75	104	0	0	
30	50	5	75	0	75	106	0	0	
31	50	5	75	0	75	115	0	0	
32	50	5	75	0	75	114	0	0	
33	50	5	75	0	75	105	0	0	

Calibration data from one case: response 1 - 33.

Number	Timing	Colour Solution	C1 Red	C1 Green	C1 Blue	C2 Red	C2 Green	C2 Blue	Key Pressed
32	50	5	75	0	75	114	0	0	1
33	50	5	75	0	75	105	0	0	1
34	50	5	75	0	75	109	0	0	1
35	55	1	30	30	30	53	0	0	1
36	55	1	30	30	30	58	0	0	1
37	55	1	30	30	30	59	0	0	1
38	55	2	0	65	65	44	65	0	1
39	55	2	0	65	65	46	65	0	1
40	55	2	0	65	65	54	65	0	1
41	55	2	0	65	65	58	65	0	1
42	55	2	0	65	65	48	65	0	1
43	55	2	0	65	65	49	65	0	1
44	55	2	0	65	65	61	65	0	1
45	55	2	0	65	65	49	65	0	1
46	55	2	0	65	65	45	65	0	1
47	55	3	0	60	60	0	66	0	1
48	55	3	0	60	60	0	71	0	1
49	55	3	0	60	60	0	64	0	1
50	55	3	0	60	60	0	63	0	1
51	55	3	0	60	60	0	67	0	1
52	55	3	0	60	60	0	70	0	1
53	55	3	0	60	60	0	63	0	1
54	55	3	0	60	60	0	70	0	1
55	55	5	75	0	75	103	0	0	1
56	55	5	75	0	75	104	0	0	1
57	55	5	75	0	75	110	0	0	1
58	55	5	75	0	75	115	0	0	1
59	55	5	75	0	75	116	0	0	1
60	55	5	75	0	75	108	0	0	1
61	55	5	75	0	75	106	0	0	1
62	55	5	75	0	75	119	0	0	1
63	55	5	75	0	75	101	0	0	1

Calibration data from one case: response 32 - 63

Number	Timing	Colour Solution	C1 Red	C1 Green	C1 Blue	C2 Red	C2 Green	C2 Blue	Key F
64	55	6	75	0	0	114	0	0	
65	62	1	30	30	30	64	0	0	
66	62	1	30	30	30	62	0	0	
67	62	1	30	30	30	51	0	0	
68	62	2	0	65	65	49	65	0	
69	62	2	0	65	65	48	65	0	
70	62	2	0	65	65	58	65	0	
71	62	2	0	65	65	52	65	0	
72	62	2	0	65	65	43	65	0	
73	62	2	0	65	65	56	65	0	
74	62	2	0	65	65	53	65	0	
75	62	2	0	65	65	46	65	0	
76	62	3	0	60	60	0	62	0	
77	62	3	0	60	60	0	68	0	
78	62	3	0	60	60	0	69	0	
79	62	3	0	60	60	0	66	0	
80	62	3	0	60	60	0	62	0	
81	62	3	0	60	60	0	67	0	
82	62	3	0	60	60	0	70	0	
83	62	4	75	0	0	0	70	0	
84	62	5	75	0	75	102	0	0	
85	62	5	75	0	75	117	0	0	
86	62	5	75	0	75	107	0	0	
87	62	5	75	0	75	118	0	0	
88	62	5	75	0	75	104	0	0	
89	62	5	75	0	75	109	0	0	
90	71	1	30	30	30	49	0	0	
91	71	1	30	30	30	58	0	0	
92	71	2	0	65	65	42	65	0	
93	71	2	0	65	65	50	65	0	
94	71	3	0	60	60	0	69	0	
95	71	3	0	60	60	0	71	0	
96	71	3	0	60	60	0	68	0	

Calibration data from another case: response 64 - 96.

Number	Timing	Colour Solution	C1 Red	C1 Green	C1 Blue	C2 Red	C2 Green	C2 Blue	Key Pressed
93	71	2	0	65	65	50	65	0	1
94	71	3	0	60	60	0	69	0	1
95	71	3	0	60	60	0	71	0	1
96	71	3	0	60	60	0	68	0	1
97	71	3	0	60	60	0	66	0	1
98	71	3	0	60	60	0	70	0	1
99	71	5	75	0	75	119	0	0	1
100	71	5	75	0	75	103	0	0	1
101	71	5	75	0	75	115	0	0	1
102	71	5	75	0	75	114	0	0	1
103	83	1	30	30	30	60	0	0	1
104	83	2	0	65	65	59	65	0	1
105	83	2	0	65	65	44	65	0	1
106	83	2	0	65	65	52	65	0	1
107	83	2	0	65	65	59	65	0	1
108	83	2	0	65	65	48	65	0	1
109	83	3	0	60	60	0	64	0	1
110	83	3	0	60	60	0	70	0	1
111	83	3	0	60	60	0	65	0	1
112	83	3	0	60	60	0	68	0	1
113	83	3	0	60	60	0	64	0	1
114	83	3	0	60	60	0	67	0	1
115	83	3	0	60	60	0	66	0	1
116	83	3	0	60	60	0	66	0	1
117	83	3	0	60	60	0	68	0	1
118	83	4	73	0	0	0	70	0	1
119	83	5	75	0	75	102	0	0	1
120	83	5	75	0	75	109	0	0	1
121	83	5	75	0	75	118	0	0	1
122	83	5	75	0	75	106	0	0	1
123	83	5	75	0	75	107	0	0	1
124	83	5	75	0	75	119	0	0	1
125	83	5	75	0	75	107	0	0	1

Calibration data from another case: response 93 - 125.

Number	Timing	Colour Solution	C1 Red	C1 Green	C1 Blue	C2 Red	C2 Green	C2 Blue	Key Press
14	71	1	30	30	30	54	0	0	1
18	83	1	30	30	30	55	0	0	1
4	55	1	30	30	30	48	0	0	1
9	62	1	30	30	30	54	0	0	1
1	50	2	0	65	65	56	65	0	1
10	62	2	0	65	65	47	65	0	1
19	83	2	0	65	65	48	65	0	1
5	55	2	0	65	65	44	65	0	1
15	71	2	0	65	65	47	65	0	1
20	83	3	0	60	60	0	68	0	1
6	55	3	0	60	60	0	64	0	1
11	62	3	0	60	60	0	68	0	1
21	83	3	0	60	60	0	69	0	1
16	71	4	72	0	0	0	70	0	1
22	83	4	73	0	0	0	70	0	1
12	62	4	70	0	0	0	70	0	1
7	55	4	70	0	0	0	70	0	1
2	50	4	70	0	0	0	70	0	1
13	62	5	75	0	75	103	0	0	1
17	71	5	75	0	75	103	0	0	1
8	55	5	75	0	75	103	0	0	1
3	50	5	75	0	75	103	0	0	1
23	83	5	75	0	75	102	0	0	1

Values from a case, sorted by Colour Solution.

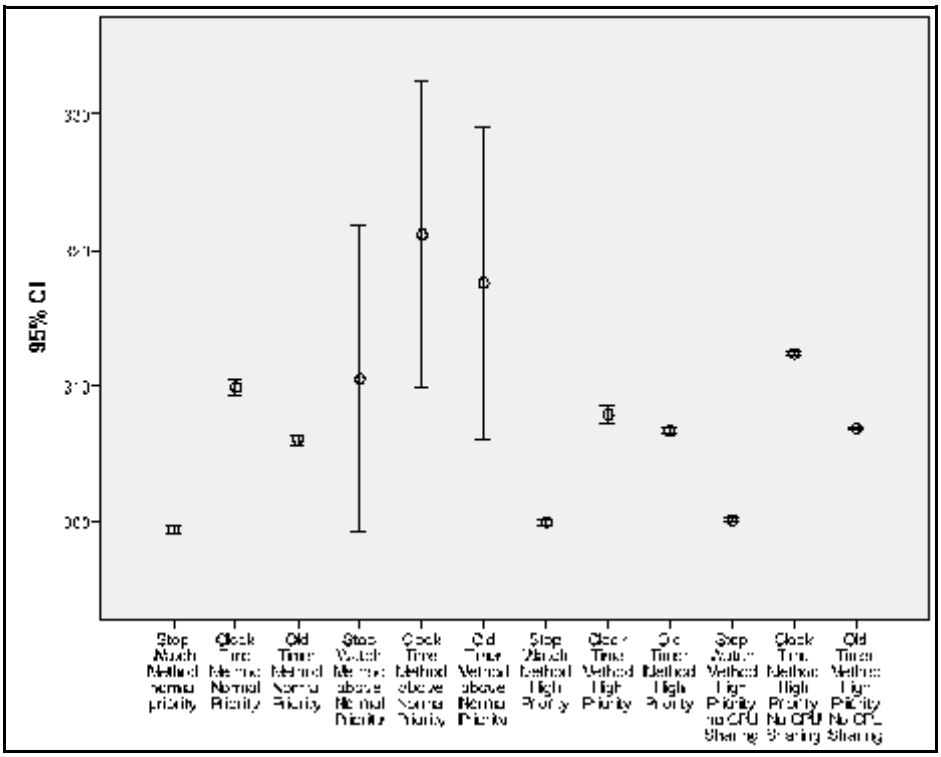
C1RA	C1GA	C1BA	C2RB	C2GB	C2BB	NAME
30	30	30	56	0	0	parvocal100441976C. Solution 1
0	65	65	49	65	0	parvocal100441976C. Solution 2
0	60	60	0	66	0	parvocal100441976C. Solution 3
81	0	0	0	70	0	parvocal100441976C. Solution 4
75	0	75	111	0	0	parvocal100441976C. Solution 5

Mean values for an individual summary, showing appropriately coloured cell pairs in the database table.

Timing Tests

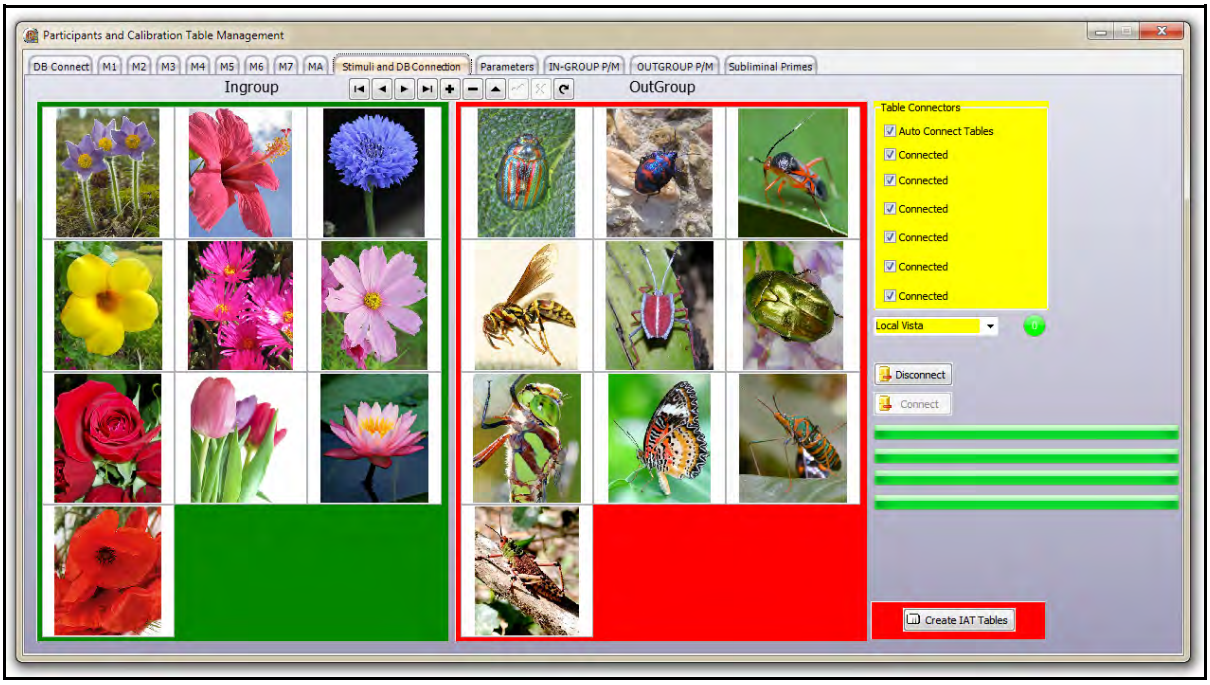
Timing test on i3-2100 CPU @ 3.10GHz (50 ms interval, 500 iterations).

Timing test on Pentium Dual T2390 @ 1.86 GHz (50 ms interval, 500 iterations).



Timing tests using different methods, and thread priorities (timing interval = 300 ms).

IAT Experiment 1: Species Bias Test



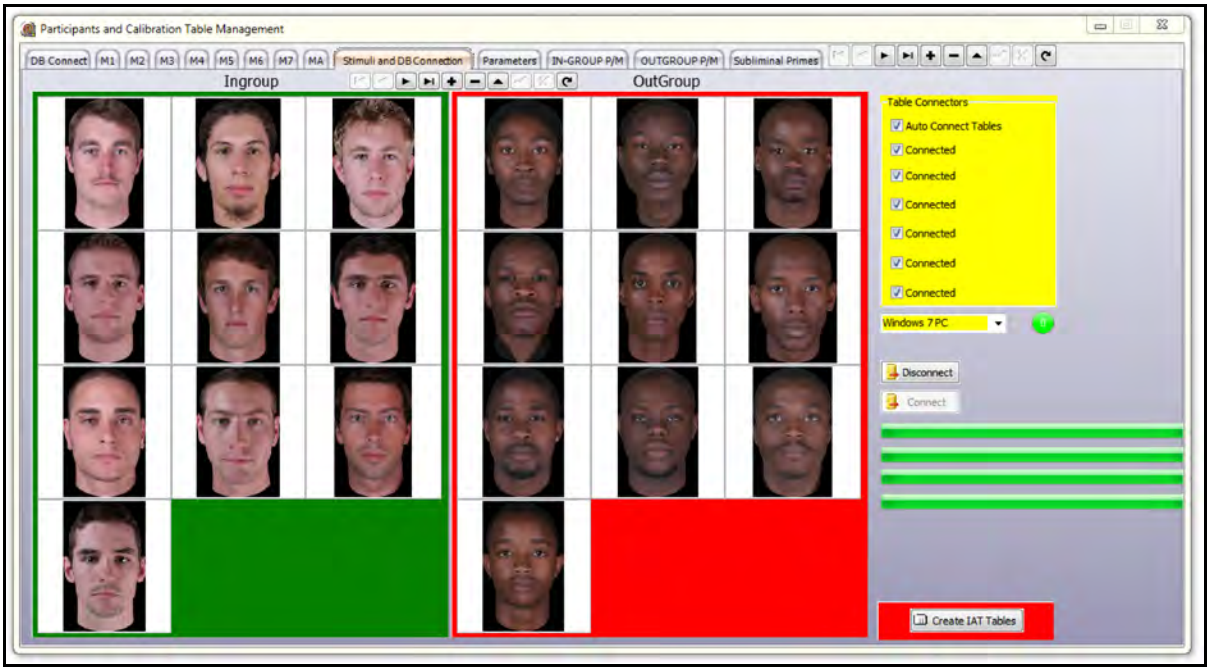
IAT settings utility showing pictures used for Species IAT.

The other tabs are used to upload the instruction screens, questions, and other parameters. It is essentially an interface which communicates with any MySQL database with a host name or IP, where user access privileges are set up. The stimuli were mined from various sources on the internet, and converted to bitmaps, cropped, and colour corrected. There were 10 exemplars of each category, and 10 words in each list.

Pleasant	Unpleasant
Lovely	Pain
Joy	Hatred
Gentle	Chaos
Peaceful	Danger
Beauty	Conflict
Wealth	Anger
Generous	Horrible
Trust	Suspicious
Calm	Dishonest
Happy	Fear

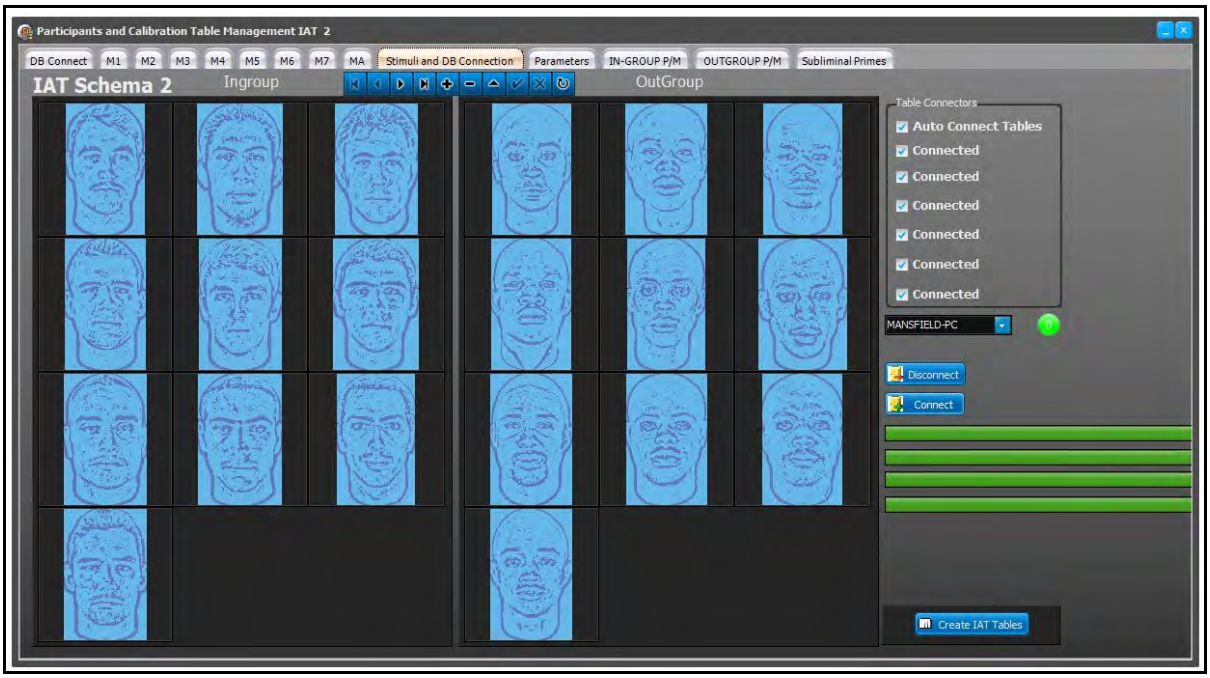
Pleasant/Unpleasant word lists.

IAT Experiment 2: Race Test

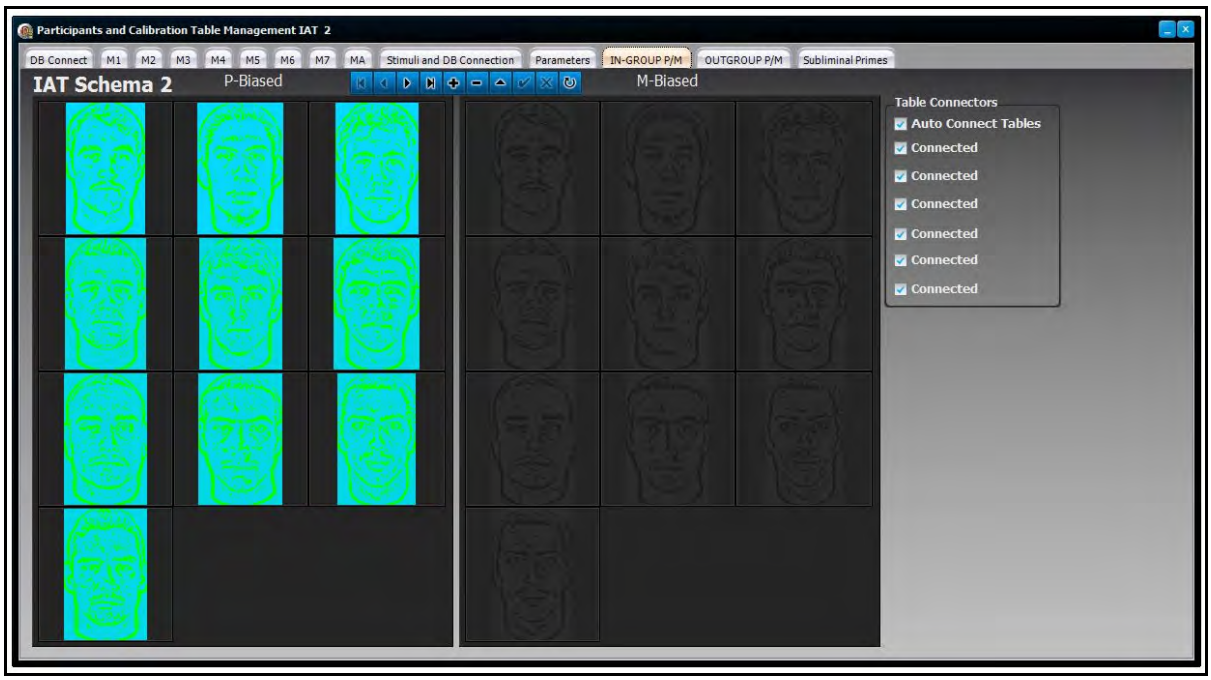


IAT settings utility showing pictures used for Race IAT.

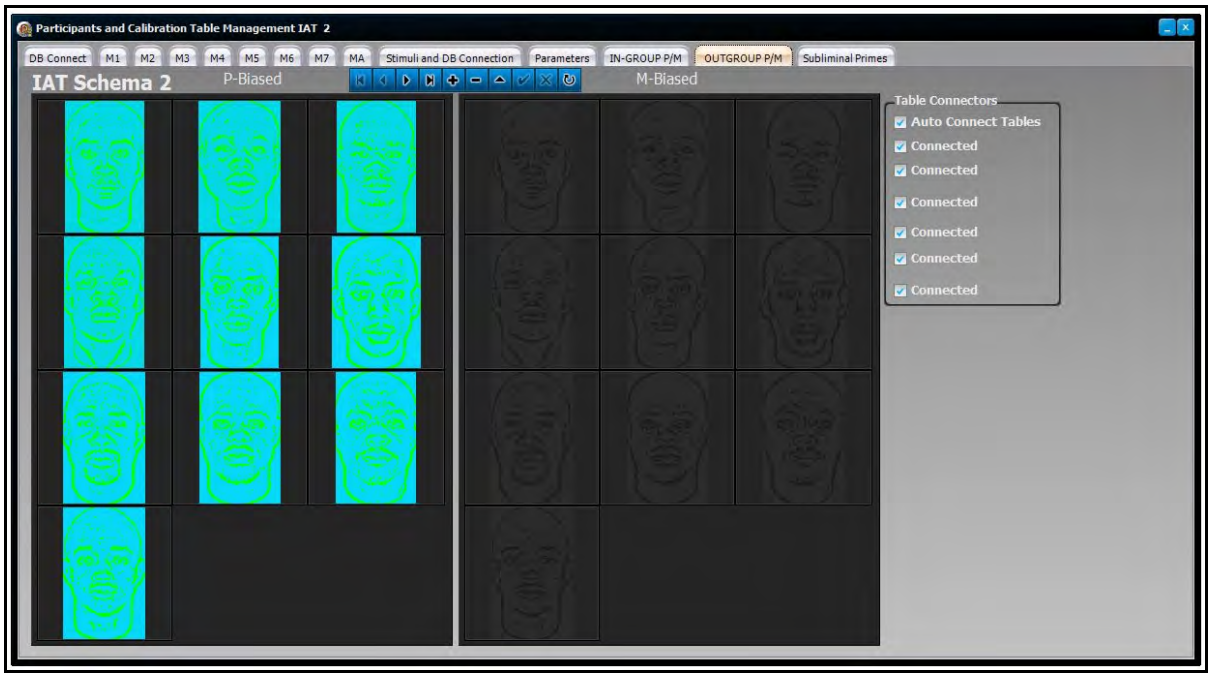
IAT Experiment 3: Race test with M/P-biasing intervention



IAT settings utility showing pictures used in IAT Experiment 3: blocks 2, 4.



IAT settings utility showing pictures used for IAT experiment 3: blocks 3/5 ('White' images).



IAT settings utility showing pictures used for IAT Experiment 3: blocks 3/5 ('Black' images).

Appendix 2

Personal correspondence with Moshe Bar: Nov 2, 2010, at 5:36 AM

On Nov 2, 2010, at 5:36 AM, Douglas Mansfield wrote:

‘Dear Moshe

I was introduced to your work by Sid Segalowitz of Brock University in 2004, and have been following your publications with interest. My main interest focus is implicit cognition, so your dissociation of these pathways has suggested various possibilities for my research.

I am attempting to replicate your 2007 'Magnocellular Projections as the Trigger of Top-Down Facilitation in Recognition' experiment as the first step in a PhD project. I have written the computer code to run on the Windows platform, as we don't have Matlab, so I think I can appreciate the complexity of the experiment. The calibration of the M and P items means that the images have to be processed at runtime, and this means multithreading the image transformations efficiently.

I would be really grateful if you could give me some advice about the procedure you used.

The first question I have been wrestling with, is the choice of red/green for the P-biased stimuli:

Did you consider any other colour combinations for the isoluminant (P) condition? In my flicker procedure, I can get reasonably close to the 'merge' point with red/green, but only at about 18 - 20 hz, and the merging seems like flicker fusion since the target appears yellowish in the brief moments when it seems to become steady.

Secondly, the procedure your team used (Our luminance threshold-finding algorithm computed the mean of the turnaround points above and below medium-gray background and reliably converged around the true background value. From this threshold, the appropriate luminance (~3.5% Weber contrast)
value was computed for the grayscale line drawings to be used in the low-luminance-contrast (M-biased) condition.) refers to Weber contrast - something I am not quite familiar with.

Since I am manipulating a 3 byte array of pixels (RGB) in which each byte can vary from 0 to 255 (hex FF) would this calculation (to get the 3.5% Weber contrast) be correct: e.g. ((Red 122, Green 122, Blue 122) - background value(e.g. R 118 G 118 B 118)) * 8.925) = R 4 G 4 B 4 * 8.925;

Foreground = R 122 G 122 B 122+ R 36 G 36 B 36 (8.925 = 3.5% of 255 rounding 8.925 up to 9 since I am working with bytes, not a floating point type).

I am wondering if this calculation makes the images look way too bright - certainly brighter than the pictures in your article.

Lastly, your article states that 'Stimuli of both types were presented in a centered 256 x 256 pixel square for 1500 ms, with 480 stimulus trials
collected in each condition. We used a rapid event-related design and therefore added

33% (240) null trials across four experimental runs...'

In your experiment, the images were projected onto a screen and visible through to participants via a mirror system built into the head coil. I am wondering how big the images were (in terms of how many degrees of the visual field), since image size does seem to be a factor when you consider the relatively poor spatial resolution of the M system compared with the P system.

It looks there were 360 * 2 = 720 trials + a total of 240 null trials = 960 trials divided into 4 blocks. So the P, M, and null P/M were presented in random order (240 trials per block). Does this sound correct? It wasn't really a trivial task to do all this randomizing, and then track the response type, accuracy, and latencies, and record them to a database - so it seems to me that the devil certainly does reside in the details!

I'm really sorry to ask you all these tiresome technical details. If you have the time, or can refer me to a lab assistant or someone else involved in the project, I would so appreciate it.

Kind regards

Doug.

Douglas Mansfield
Clinical Psychologist
Lecturer: School of Psychology
University of Kwa-Zulu Natal
P/Bag X01
Scottsville
3209
SOUTH AFRICA

Tel: +27-33-260-5100 (w)
Fax: +27-33-260-5809

E-mail: MansfieldD@ukzn.ac.za'

Personal correspondence from Moshe Bar: Nov 2, 2010, at 2:04 PM

bar@nmr.mgh.harvard.edu

'Doug,

thank you for your detailed email and interest in our work. i am cc-ing Kestas, who was the postdoc who programmed and ran the experiment, and i believe he can answer your questions once he gets to it.

thanks and good luck with your work,

Moshe'

Personal correspondence with Kestas Kveraga (2010/11/26 8:09 PM)

kestas@nmr.mgh.harvard.edu

‘Dear Doug,

thank you for your interest in our work. I'm quite busy right now, so I will answer your questions in detail in a few days. My quick thought regarding isoluminance is that Windows timing can be quite unreliable - have you checked the actual display durations with independent means? Though 18-20 Hz is in the right range.

Kestas’

Personal correspondence from Kestas Kveraga (2010/11/26 08:09 PM)

‘Hi Doug,
not really, but here's a quick description of the isoluminance particulars. The color values were as follows from what I recall: green was held constant at around 141 (on the RGB scale of 0-255), while red went up and down depending on the response ("flickering or steady?"), crossing the isoluminance range several times, and the average usually ended up being plus or minus couple points around 150.
The background for finding isoluminance was middle gray, I think around 127 or so. The fused drawings do not have an easily definable color, they're sort of grayish-yellowish I guess, but it's hard to say.

This method worked well with several Mac machines (laptops and desktops) using MATLAB and Psychtoolbox and a variety of display types (CRTs, LCDs, projectors in MR and MEG environments), and has been used in several experiments in Moshe's lab and I think in other labs as well. I created the displays dynamically, reading in jpegs of line drawings, finding foreground/background, and then adjusting the foreground and background values to suit the M or P biasing. In other words, for isoluminance finding the exact same pixels (matrix indices) would switch between red and green at around 14 Hz, with the red adjusted until they fused. Now, in MR/MEG projection environments the fusion wasn't always absolutely perfect (depending on the subject, some could see the tiniest bit of twitching) but that wasn't something we could control and in any case it worked well enough to yield reliable behavioral and neural effects. Also, I seem to recall that some labs (Gegenfurtner?) don't even bother calibrating to the individual, but use average isoluminance values.

I'm not exactly sure, but I think converting the matrices into graphics files (as you seem to be doing) may introduce artifacts (other color values that aren't isoluminant red or green) which may be preventing fusion.
When we saved the dynamically created images as jpegs (I think) and analyzed the spatial frequency content for another study, we found some artifacts (other, non-isoluminant values surrounding the stimuli) which, if flickered, would probably prevent fusion. The other potential concern is timing nonlinearities - if the cycle sometimes gets extented or reduced because of whatever Windows background processes are doing, it would mess with the fusion.

Hope this helps,
Kestas'

Personal correspondence with Anthony Greenwald on 27 August, 2012

'Dear Anthony

Thank you for the link, and for making contact. I have used a version of this, and somehow lost track of the details.

I will send you comparative data on race and species, if you wish.

Thanks again

Doug'

27 August 2012

'Hi Doug -

The flower-insect IAT appeared in the first article on the IAT in JPSP 1998. You can find it at http://faculty.washington.edu/agg/pdf/Gwald_McGh_Schw_JPSP_1998.OCR.pdf

That IAT was intended more as a demonstration showing that the method could work than for actual interest in studying individual differences implicit attitudes toward flowers and insects. However, subsequently it was used to find that entomology PhD students differed from Psychology PhD students by not showing "automatic preference" for flowers over (this wasn't ever published).

Thanks for your interest (and reminding me that the first public presentation was at Claremont -- I had forgotten that).

Best wishes,
- Tony'

Image Processor

BackGround

0 | 201 | 254

0 | 60 | 0

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

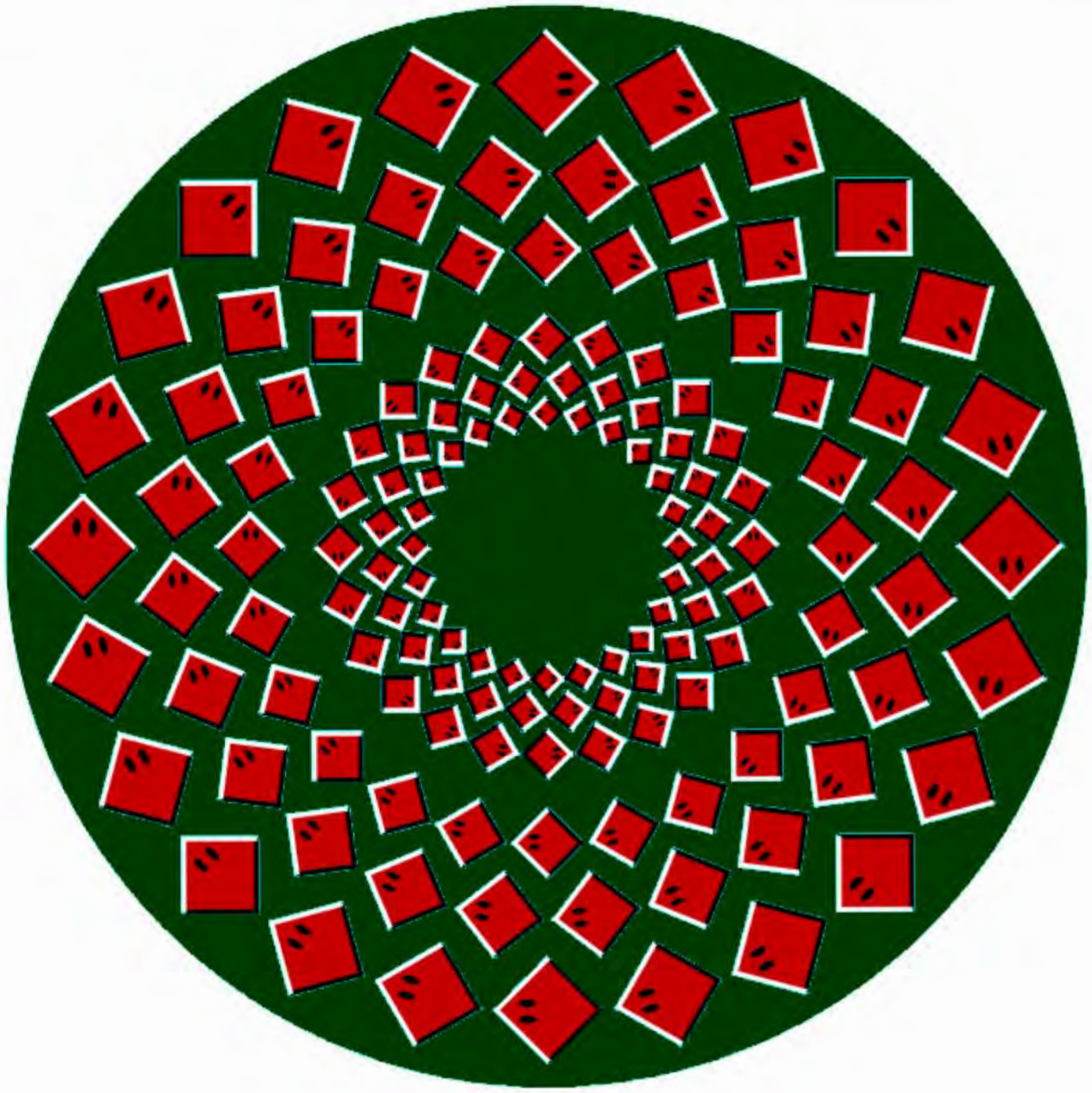


Image Processor

BackGround

0 | 201 | 254

0 | 70 | 0

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

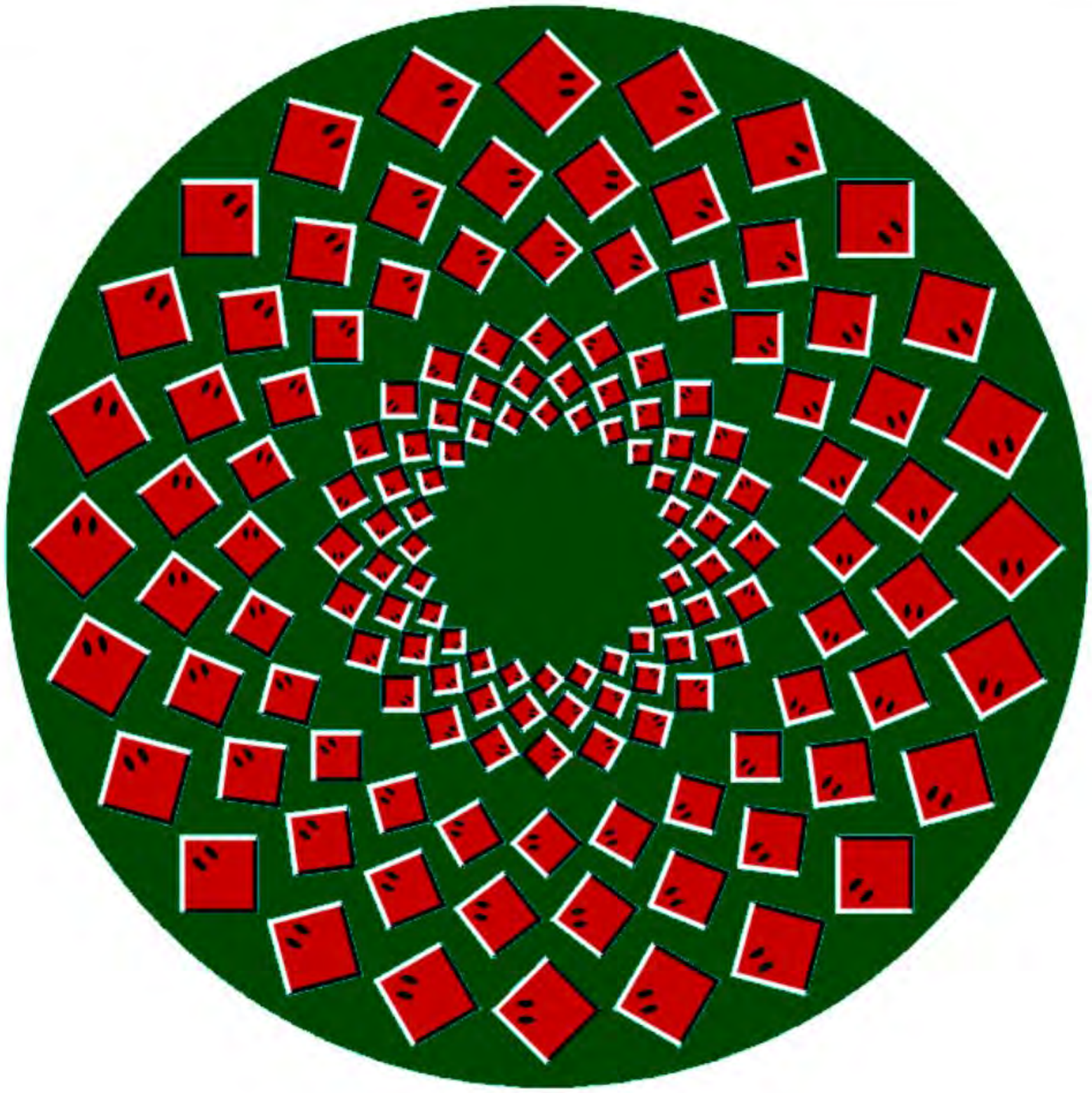


Image Processor

BackGround

0 | 201 | 254

0 | 80 | 0

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

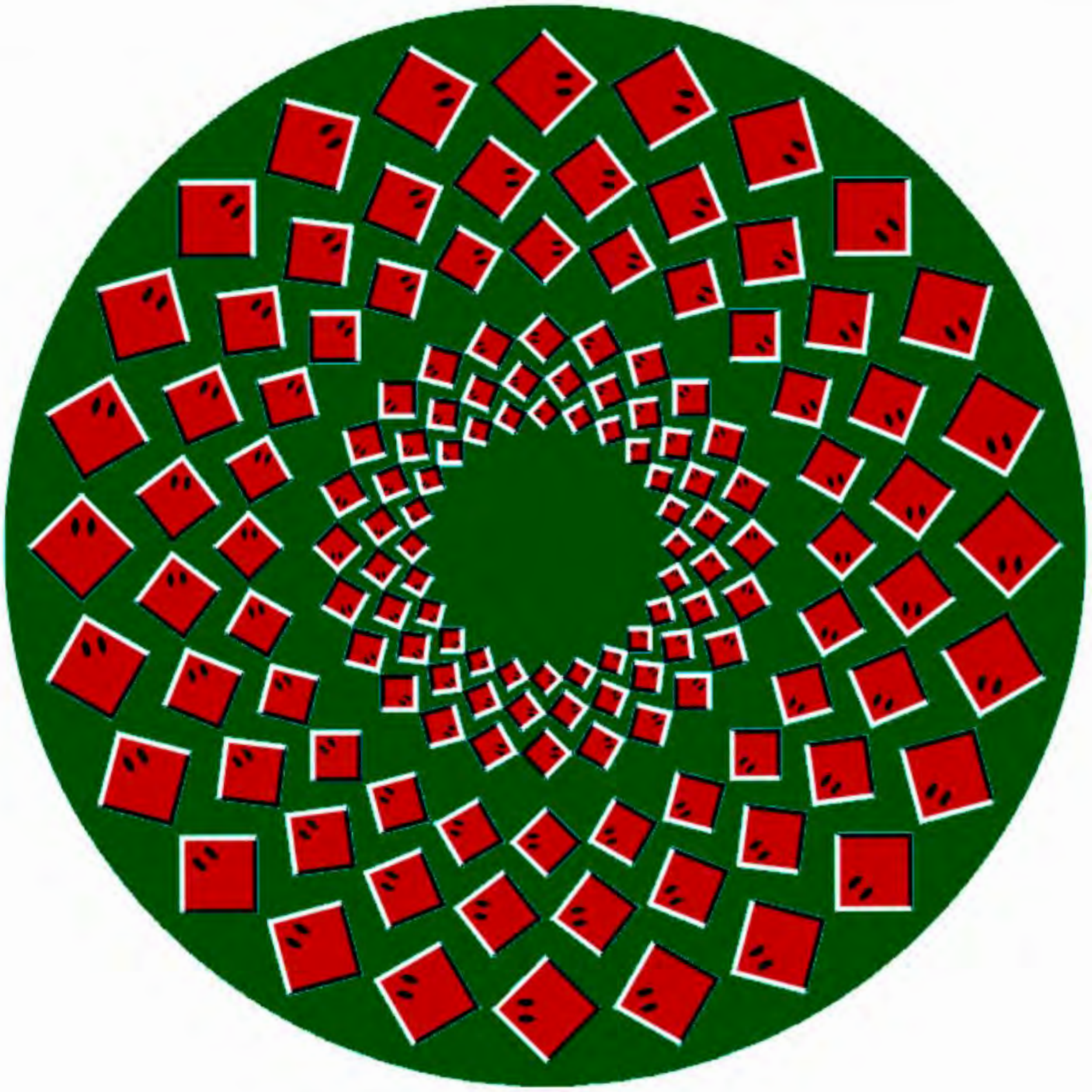


Image Processor

BackGround

0 | 201 | 254

0 | 90 | 0

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

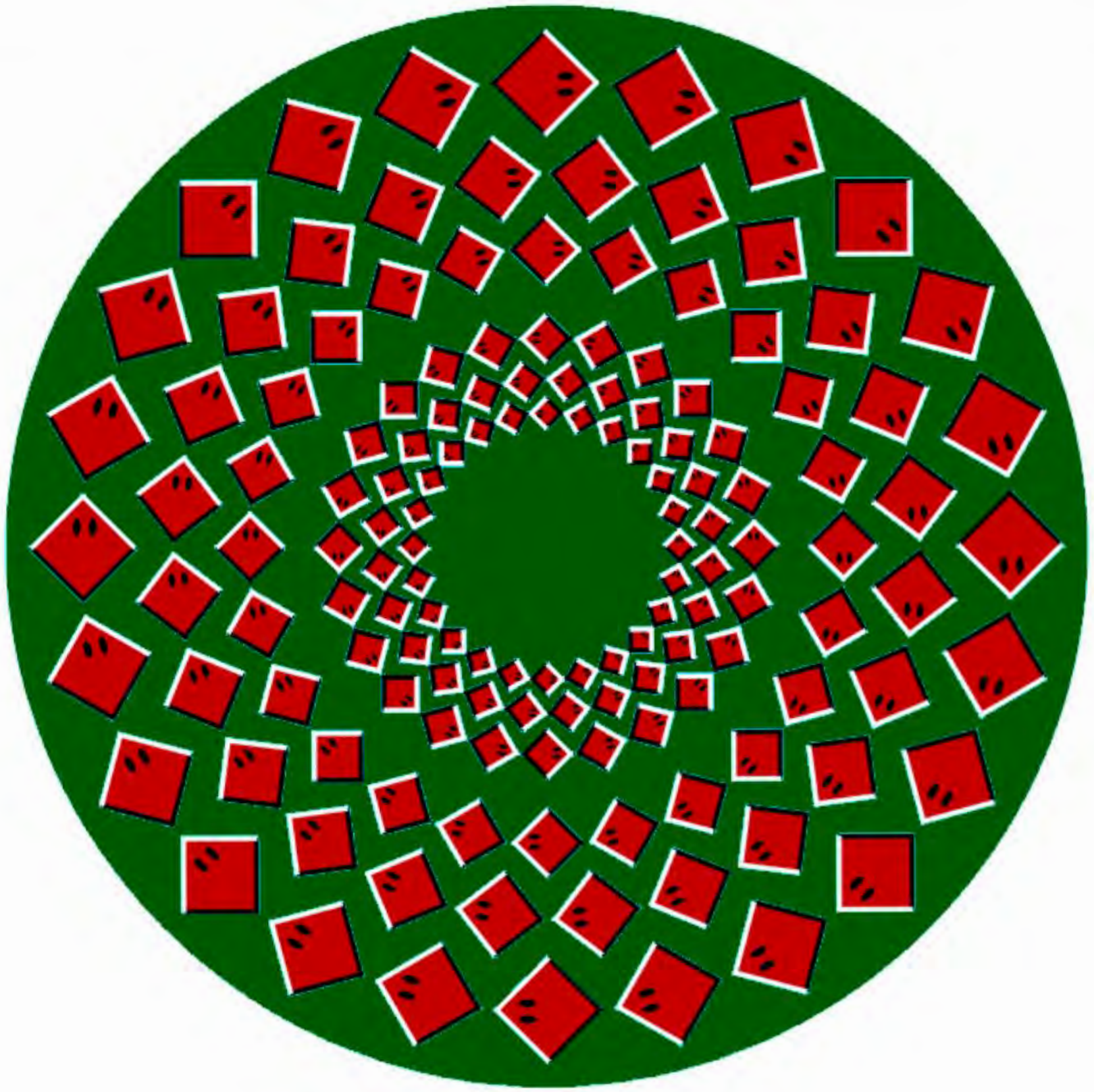


Image Processor

BackGround

0 | 201 | 254

0 | 100 | 0

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

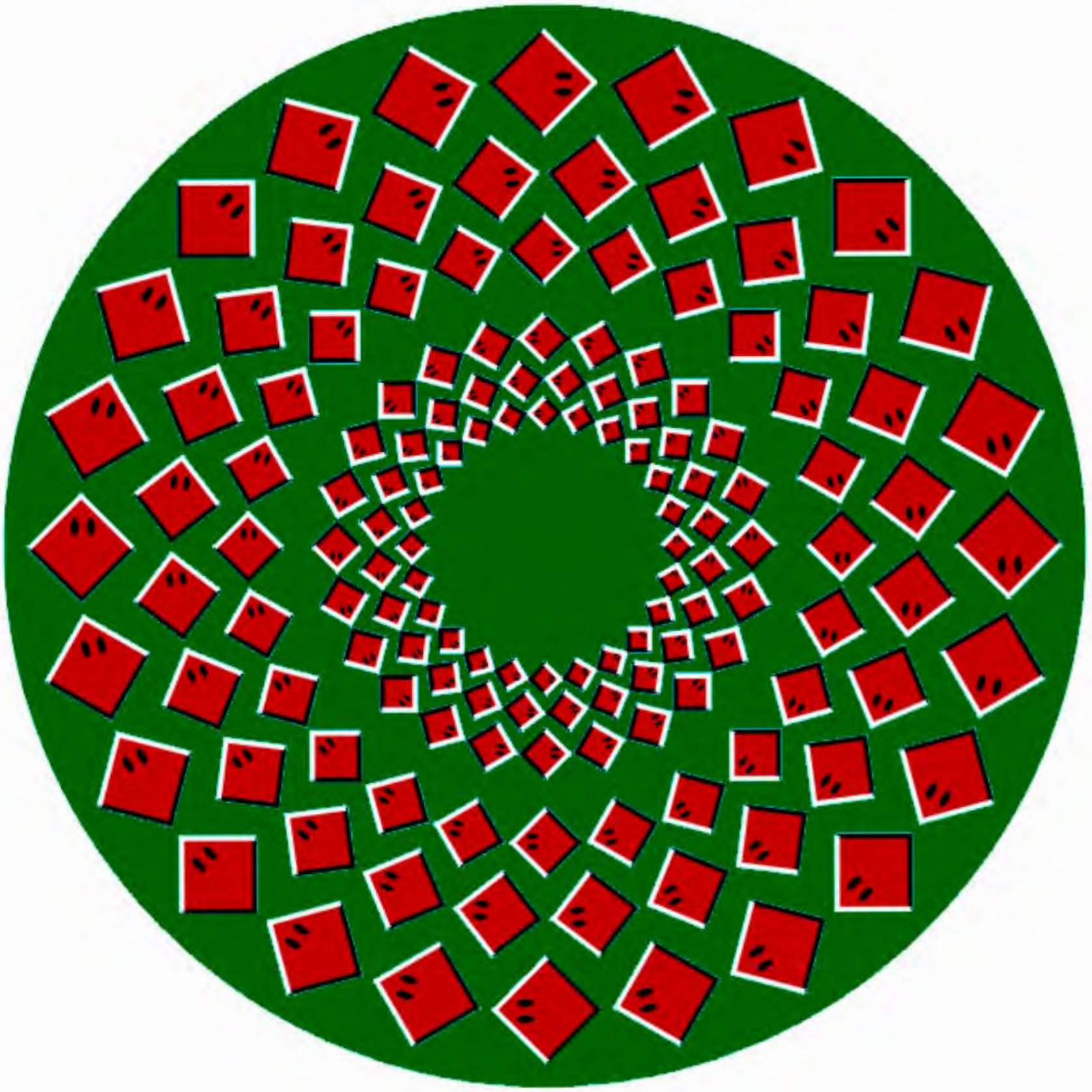


Image Processor

BackGround

0 | 201 | 254

120 | 120 | 120

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

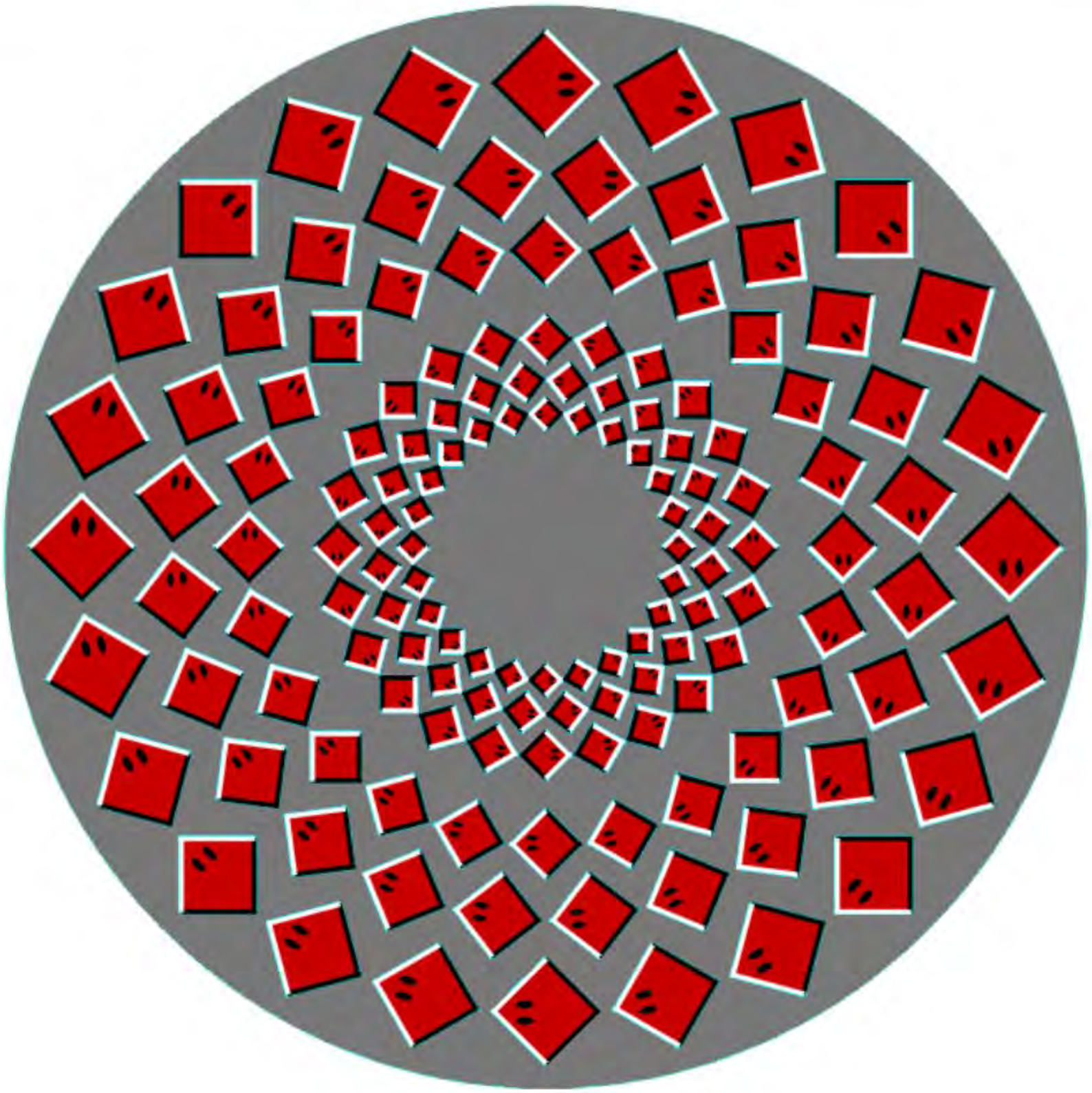


Image Processor

BackGround

0 | 201 | 254

0 | 150 | 0

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

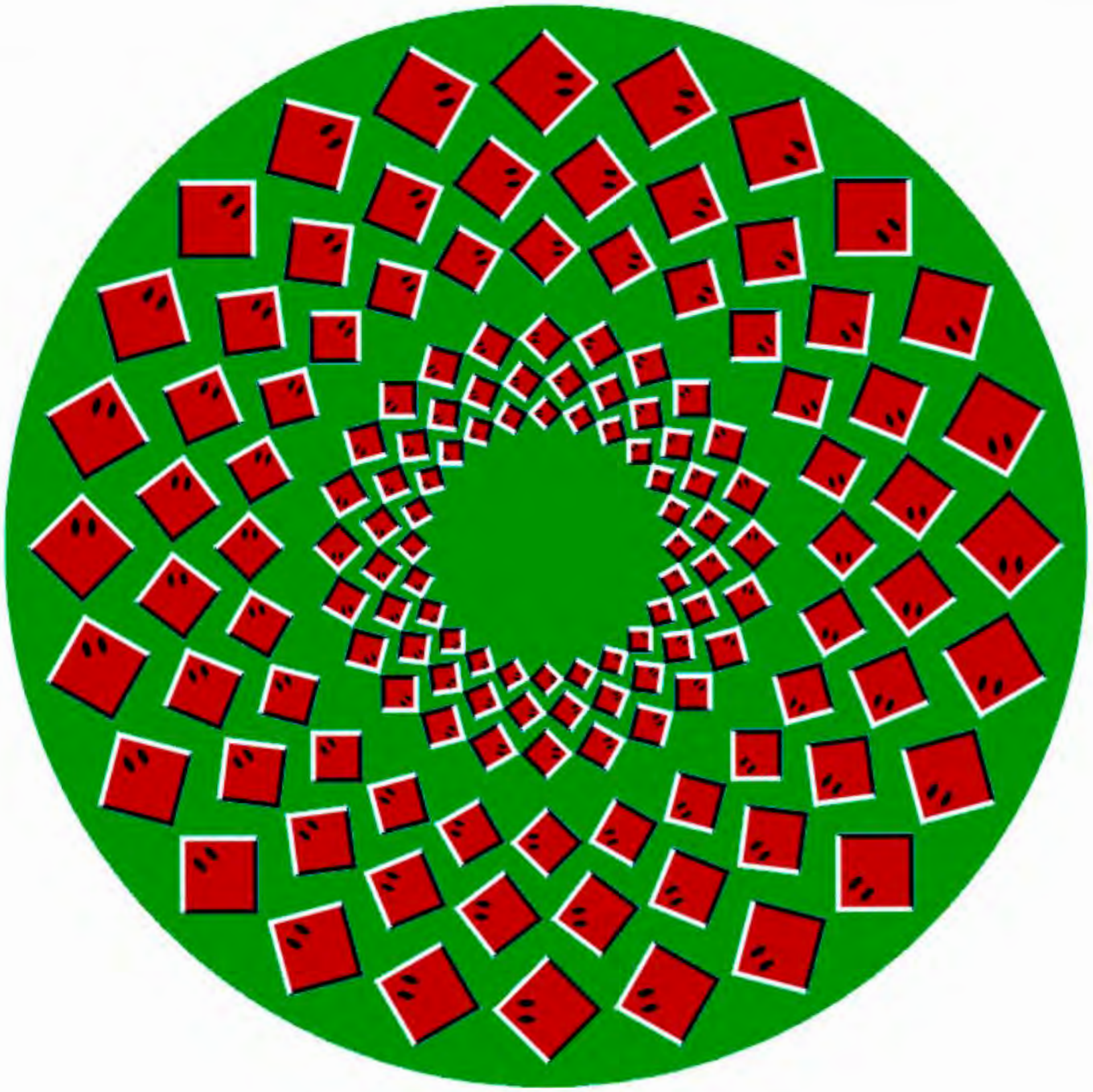


Image Processor

BackGround

0 | 201 | 254

0 | 200 | 0

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

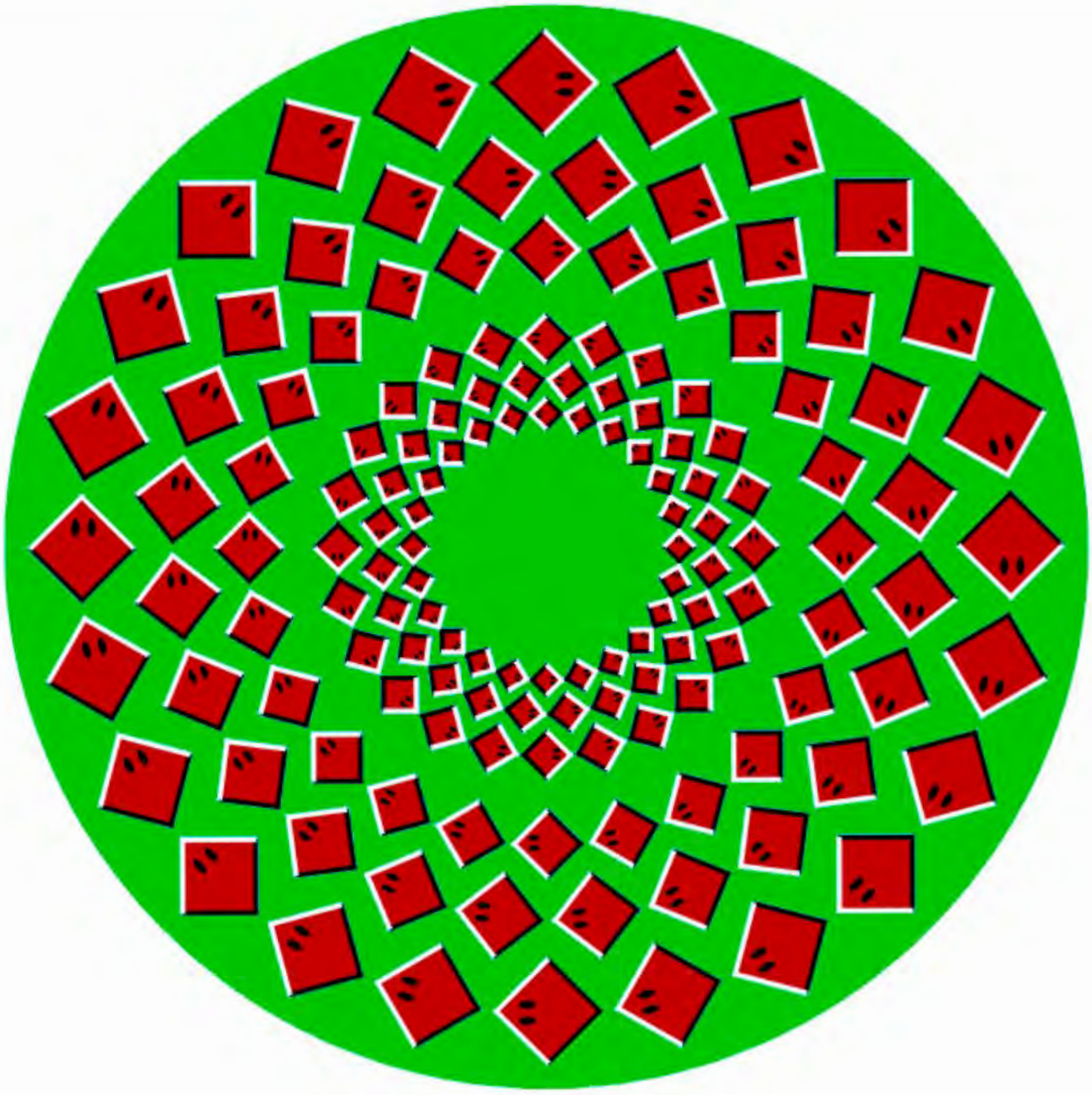


Image Processor

BackGround

0 | 201 | 254

0 | 0 | 255

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

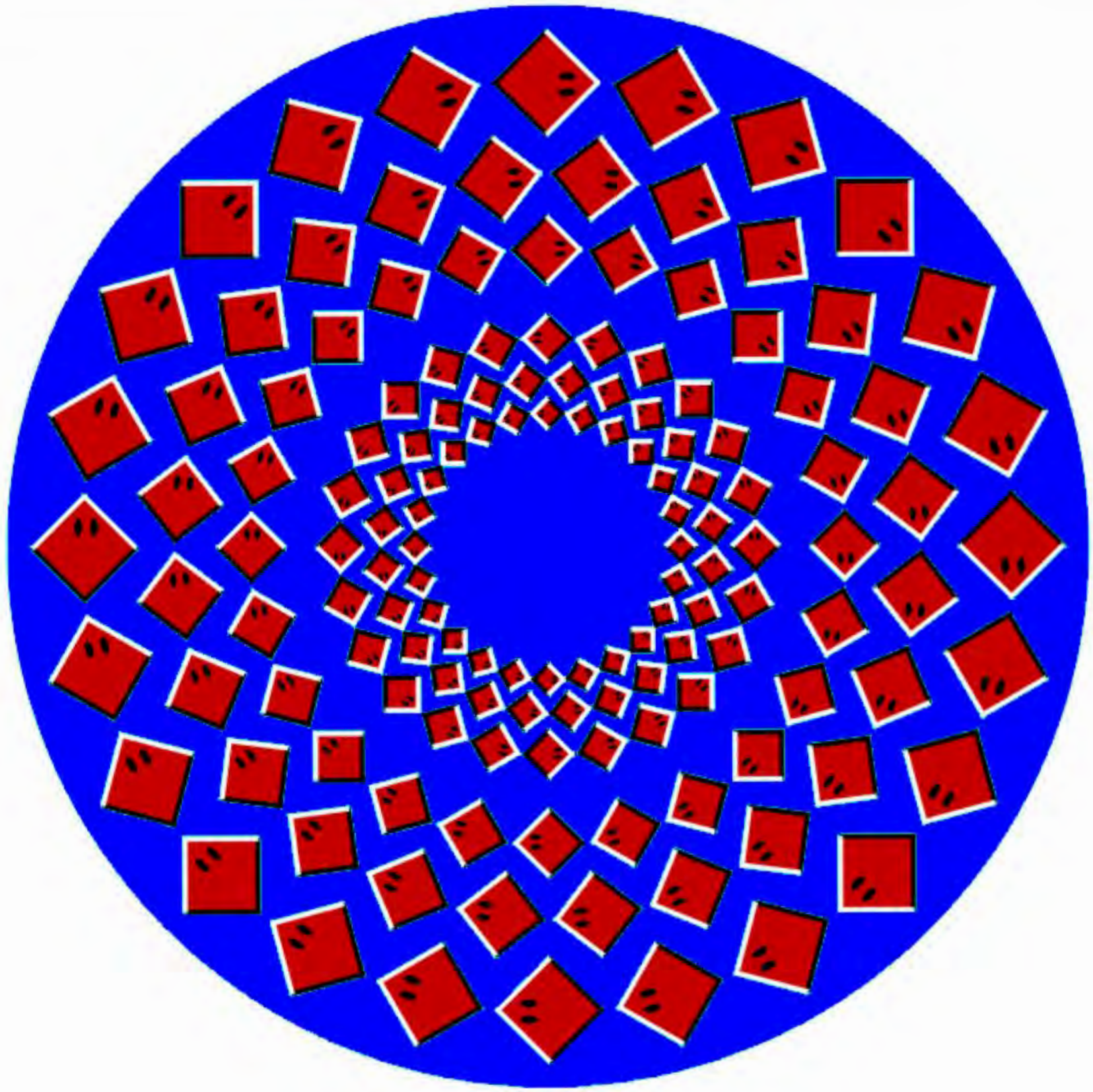
Rotate CW | Rotate CCW

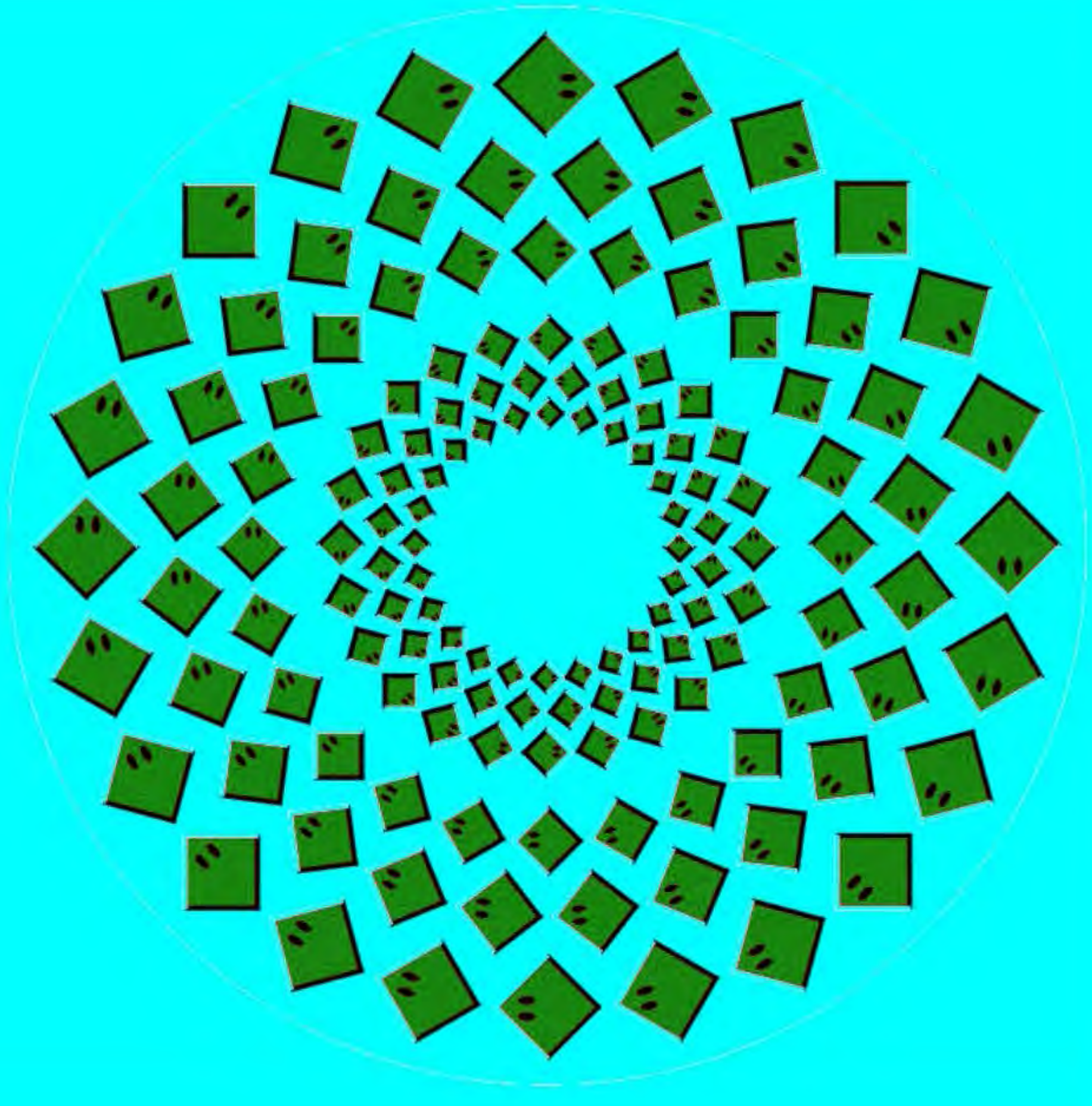
0°

Save Image

Load Image

Reload Image





```
unit ArrayFunctions;


interface
uses FileFunctions,ReportUnit,StrUtils;
Var
Checkarray : Array[1..10] OF BOOLEAN;
Indexarr1 : ARRAY[1..10] OF STRING[13];
Indexarr2 : Array[1..10] OF STRING[13];
Typearray : Array[1..20] OF BOOLEAN;
Controlarray20 : Array[1..20] OF String[1];
SubJoin : ARRAY[1..20] OF STRING[13];
Joinarray : Array[1..20] OF STRING[13];
Block24Combine : Array[1..40] OF Double;
Block35Combine : Array[1..80] OF Double;
Controlarray40,IDArray40 : Array[1..40] OF String[2];
Procedure RandomsortP(Var Pleasant: TPleasant; Pl : Integer); External
'RandomPU.dll';
Procedure RandomsortU(Var Unpleasant : TPleasant; Ul : Integer); External
'RandomPU.dll';
Procedure RandomsortF(Var Flowers : TPleasant; Fl : Integer);  External
'RandomPU.dll';
Procedure RandomsortC(Var Condoms: TPleasant; Cl : Integer);External
'RandomPU.dll';
Procedure JoinB1;      //PLEASANT AND UNPLEASANT
Procedure JoinB2;
Procedure JoinB4;
Procedure Combine24;
Procedure Combine35;
Procedure Join35;  //Blacks, Whits, PLEASANT UNPLEASANT
Procedure Join53;  //Blacks, Whites, PLEASANT UNPLEASANT



implementation

Procedure GetRandom(VAR Index : Integer);
BEGIN
REPEAT
Index := Random(10)+1;
UNTIL (Index >0) AND (Index <11);
END;

Procedure Random4(Var IndexJ: Integer);
BEGIN
REPEAT
   Indexj:=(Random(4)+1);
 UNTIL (Indexj=1) OR (Indexj<=4);
END;

Procedure Random2(Var Indexj: Integer);
BEGIN
  REPEAT
```

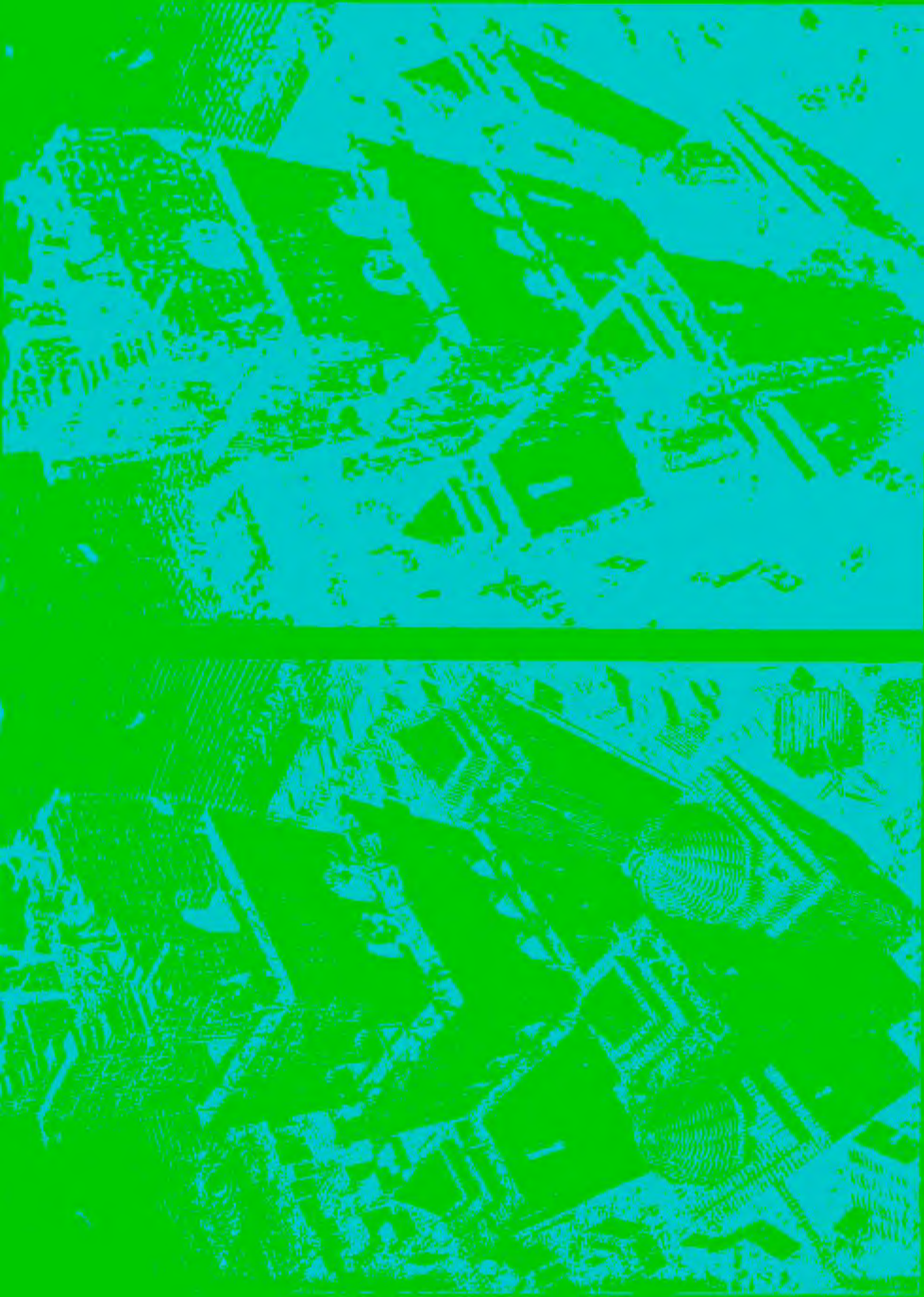
```
  Indexj:=(Random(2)+1);
  Indexj:=(Random(2)+1);
  Indexj:=(Random(2)+1);
  Indexj:=(Random(2)+1);
  UNTIL (Indexj=1) OR (Indexj=2);
END;

{This procedure merges the 2 arrays so that the words appear in random order
from both lists}
Procedure JoinB1;      //PLEASANT AND UNPLEASANT
VAR
CounterP, CounterU, LoopJ,IndexJ : INTEGER;
BEGIN
  CounterP := 0;
  CounterU := 0;
  LoopJ := 0;
  REPEAT
    Random2(IndexJ);
    IF ((Indexj =1) AND (CounterP <PL)) THEN
    BEGIN
      LoopJ:= LoopJ +1;
      CounterP:=CounterP+1;
      Joinarray[Loopj]:= Pleasant[CounterP];
      ControlArray20[LoopJ]:='P';
      Typearray[loopj]:=True;
    END;
    IF ((Indexj =2) AND (CounterU <UL)) THEN
    BEGIN
      Loopj:=Loopj+1;
      CounterU:=CounterU+1;
      Joinarray[Loopj] := Unpleasant[CounterU];
      ControlArray20[LoopJ]:='Q';
      Typearray[Loopj]:=False;
    END;
  UNTIL (Loopj =(PL+UL));
  END;

  {This procedure merges the 2 arrays so that the words appear in random order
from both lists}
// NB N O T E WELL!!!!! THIS PROCEDURE ALSO MERGES THE GRAPHICS U AND P ARRAYS
Procedure JoinB2;
VAR
CounterP, CounterU, LoopJ,IndexJ : INTEGER;
BEGIN
  CounterP := 0;
  CounterU := 0;
  LoopJ := 0;
  REPEAT
    Random2(IndexJ);
    IF ((Indexj =1) AND (CounterP <FL)) THEN
    BEGIN
      LoopJ:= LoopJ +1;
      CounterP:=CounterP+1;
      Joinarray[Loopj]:= Blacks[CounterP];
```

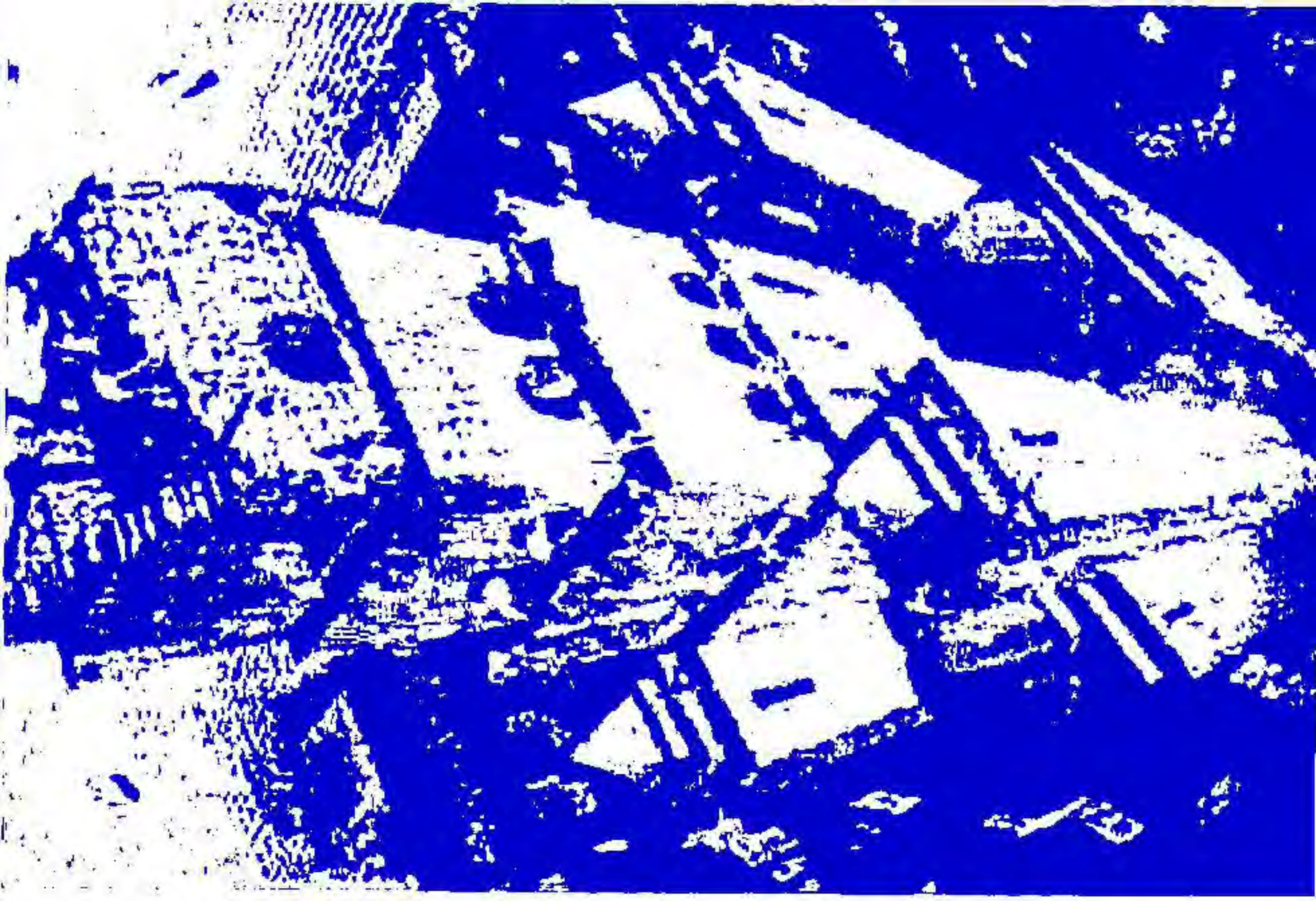
```
      Subjoin[LoopJ]:= SubP[CounterP]; // assign the subliminal array
      ControlArray20[LoopJ]:='Q';
      Typearray[loopj]:=True;
    END;
    IF ((Indexj =2) AND (CounterU <CL)) THEN
    BEGIN
      Loopj:=Loopj+1;
      CounterU:=CounterU+1;
      Joinarray[Loopj] := Whites[CounterU];
      Subjoin[Loopj] := SubU[CounterU]; // assign the subliminal array
      ControlArray20[LoopJ]:='P';
      Typearray[Loopj]:=False;
    END;
  UNTIL (Loopj =(FL+CL));
  END;

Procedure JoinB4;      //FLOWERS AND COMDOMS reverse P and Q
VAR
CounterP, CounterU, LoopJ,IndexJ : INTEGER;
BEGIN
  CounterP := 0;
  CounterU := 0;
  LoopJ := 0;
  REPEAT
    Random2(IndexJ);
    IF ((Indexj =1) AND (CounterP <FL)) THEN
    BEGIN
      LoopJ:= LoopJ +1;
      CounterP:=CounterP+1;
      Joinarray[Loopj]:= Blacks[CounterP];
      Subjoin[LoopJ]:= SubU[CounterP]; // assign the subliminal array
      ControlArray20[LoopJ]:='P';
      Typearray[loopj]:=True;
    END;
    IF ((Indexj =2) AND (CounterU <CL)) THEN
    BEGIN
      Loopj:=Loopj+1;
      CounterU:=CounterU+1;
      Joinarray[Loopj] := Whites[CounterU];
      Subjoin[Loopj] := SubP[CounterU]; // assign the subliminal array
      ControlArray20[LoopJ]:='Q';
      Typearray[Loopj]:=False;
    END;
  UNTIL (Loopj =(FL+CL));
  END;

  Procedure Combine24;
Var Loopstreet : Integer;
BEGIN
    For LoopStreet:=1 TO (FL+CL) DO
    Begin
      Block24Combine[LoopStreet]:=TempB2[LoopStreet];
    End;
  For Loopstreet:=(FL+CL+1) TO ((FL+CL)*2) DO
```

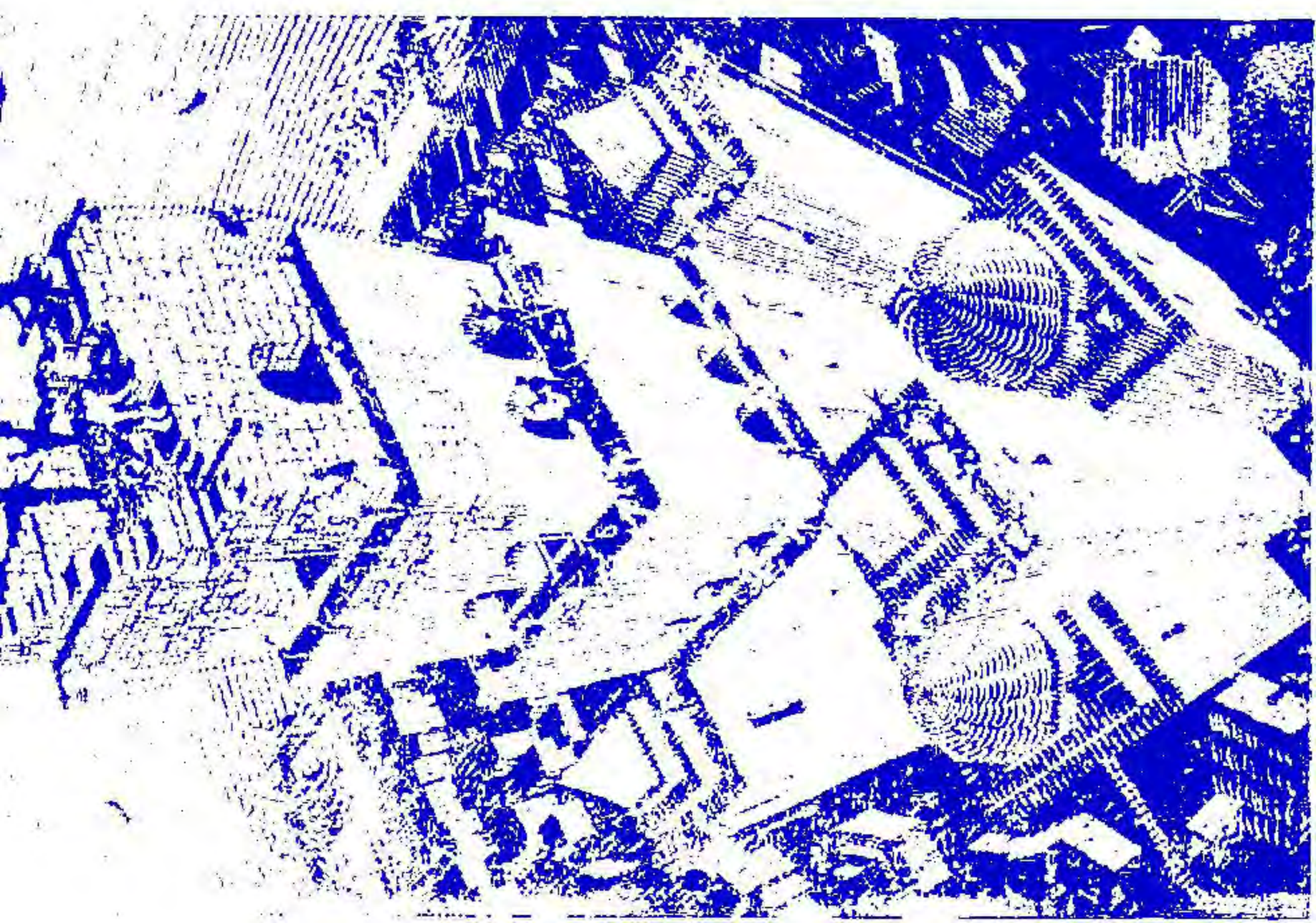
```
    Begin
      Block24Combine[LoopStreet]:=TempB4[LoopStreet-(FL+CL)];
    End;
End;

Procedure Combine35;
Var Loopstreet : Integer;
BEGIN
  For LoopStreet:= 1 To (FL+CL+PL+UL) DO
    Begin
      Block35Combine[LoopStreet]:=TempB3[LoopStreet];
    End;
  For Loopstreet:=(FL+CL+PL+UL+1) TO ((Fl+CL+PL+UL)*2) DO
    Begin
      Block35Combine[LoopStreet]:=TempB5[Loopstreet-(FL+CL+PL+UL)];
    End;
END;


Procedure WhatR;
Var
SWhat : String;
Begin
  SWhat:=LeftStr(Whatis,2);
  if SWhat='MB' Then Whatis:='MB';
  if SWhat='FB' Then Whatis:='FB';
  if Swhat='MW' then Whatis:='MW';
  if SWhat='FW' then Whatis:='FW';
End;


Procedure Join35;  //blacks, whites, PLEASANT UNPLEASANT
VAR
CounterP, CounterU, CounterF, CounterI, LoopJ,IndexJ : INTEGER;
BEGIN
  CounterP := 0;
  CounterU := 0;
  CounterF := 0;
  CounterI := 0;
  LoopJ := 0;
  REPEAT
    Random4(IndexJ);
    IF ((IndexJ =1) AND (CounterP <PL)) THEN
    BEGIN
      LoopJ:= LoopJ +1;
      CounterP:=CounterP+1;
      Joinarray40[LoopJ]:= Pleasant[CounterP];
      Typearray40[loopJ]:='1';
      ControlArray40[LoopJ]:='P';
      IDArray40[LoopJ]:='P';
    END;
    IF ((IndexJ =2) AND (CounterU <UL)) THEN
    BEGIN
      LoopJ:=LoopJ+1;
```

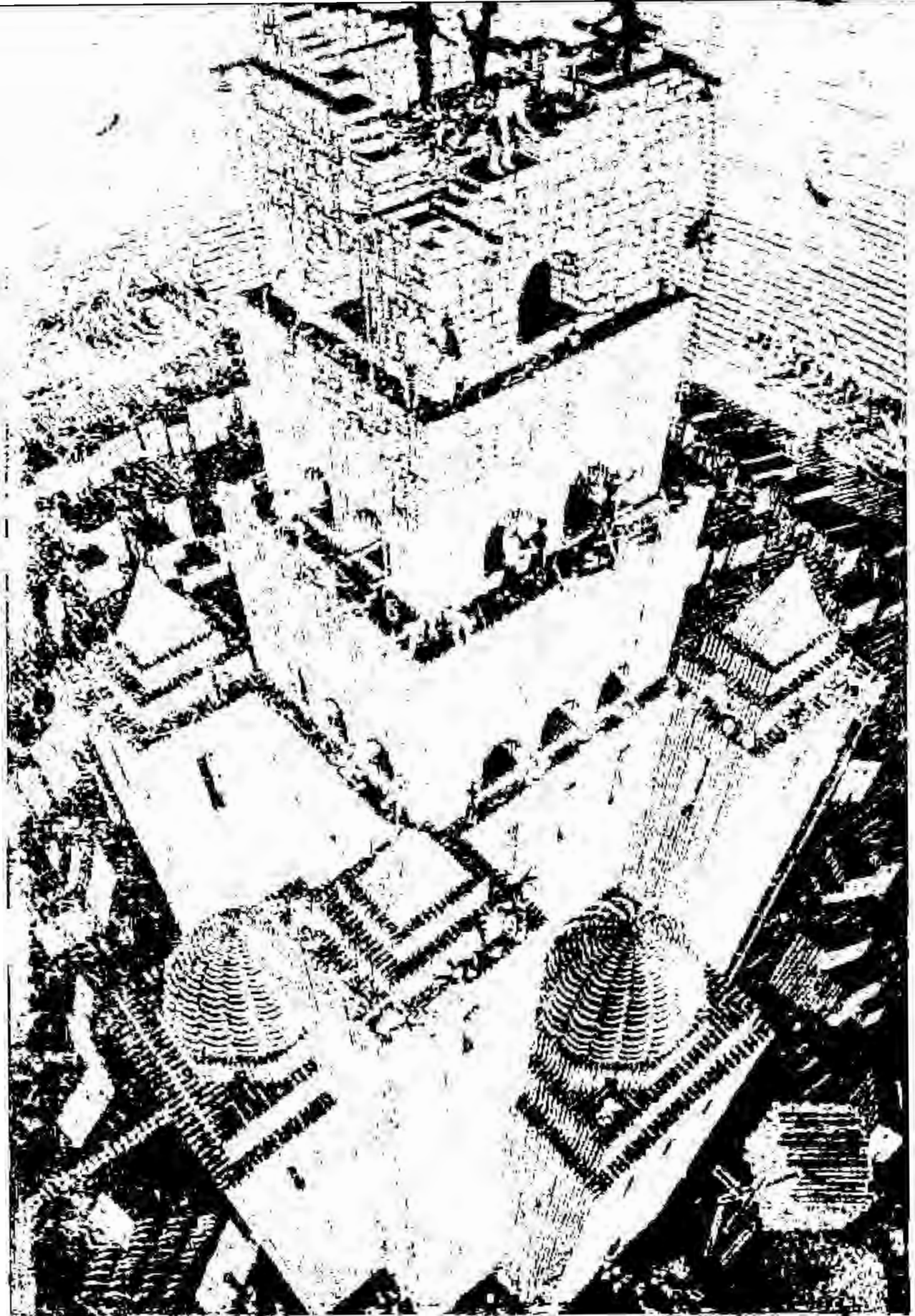
```
    CounterU:=CounterU+1;
    Joinarray40[LoopJ] := Unpleasant[CounterU];
    Typearray40[LoopJ]:='2';
    ControlArray40[LoopJ]:='Q';
    IDArray40[LoopJ]:='U';
   END;
   IF ((IndexJ =3) AND (CounterF <FL)) THEN
   BEGIN
    LoopJ:= LoopJ +1;
    CounterF:=CounterF+1;
    Joinarray40[LoopJ]:= Blacks[CounterF];
    Whatis:=JoinArray40[LoopJ];
    Typearray40[loopJ]:='3';
    ControlArray40[LoopJ]:='Q';
    IDArray40[LoopJ]:='B';
   END;
   IF ((IndexJ =4) AND (CounterI <CL)) THEN
   BEGIN
    LoopJ:= LoopJ +1;
    CounterI:=CounterI+1;
    Joinarray40[LoopJ]:= Whites[CounterI];
    Whatis:=JoinArray40[LoopJ];
    Typearray40[loopJ]:='4';
    ControlArray40[LoopJ]:='P';
    IDArray40[LoopJ]:='W';
   END;
  UNTIL (LoopJ =(PL+UL+FL+CL));
END;

Procedure Join53;  //blacks, whites, PLEASANT UNPLEASANT
VAR  //swapped scoring for trial 5
CounterP, CounterU, CounterF, CounterI, LoopJ,IndexJ : INTEGER;
BEGIN
  CounterP := 0;
  CounterU := 0;
  CounterF := 0;
  CounterI := 0;
  LoopJ := 0;
  REPEAT
   Random4(IndexJ);
   IF ((IndexJ =1) AND (CounterP <PL)) THEN
   BEGIN
    LoopJ:= LoopJ +1;
    CounterP:=CounterP+1;
    Joinarray40[LoopJ]:= Pleasant[CounterP];
    Typearray40[loopJ]:='1';
    ControlArray40[LoopJ]:='P';
    IDArray40[LoopJ]:='P';
   END;
   IF ((IndexJ =2) AND (CounterU <UL)) THEN
   BEGIN
    LoopJ:=LoopJ+1;
    CounterU:=CounterU+1;
    Joinarray40[LoopJ] := Unpleasant[CounterU];
```

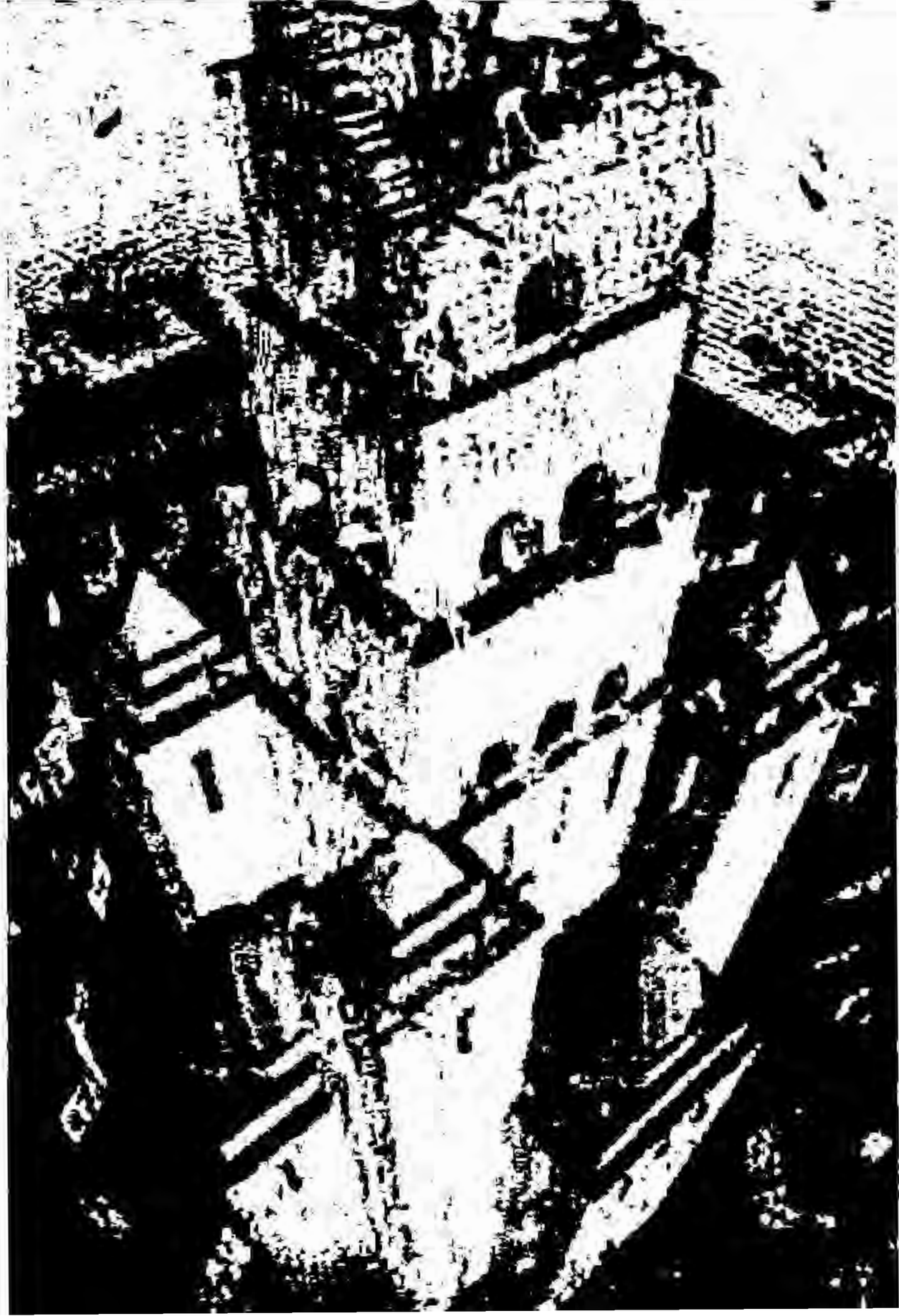
```
    Typearray40[LoopJ]:='2';
    ControlArray40[LoopJ]:='Q';
    IDArray40[LoopJ]:='U';
  END;
  IF ((IndexJ =3) AND (CounterF <FL)) THEN
  BEGIN
    LoopJ:= LoopJ +1;
    CounterF:=CounterF+1;
    Joinarray40[LoopJ]:= Blacks[CounterF];
    Whatis:=JoinArray40[LoopJ];
    Typearray40[loopJ]:='3';
    ControlArray40[LoopJ]:='P';
    IDArray40[LoopJ]:='B';
  END;
  IF ((IndexJ =4) AND (CounterI <CL)) THEN
  BEGIN
    LoopJ:= LoopJ +1;
    CounterI:=CounterI+1;
    Joinarray40[LoopJ]:= Whites[CounterI];
    Whatis:=JoinArray40[LoopJ];
    Typearray40[loopJ]:='4';
    ControlArray40[LoopJ]:='Q';
    IDArray40[LoopJ]:='W';
  END;
 UNTIL (LoopJ =(PL+UL+FL+CL));
END;


end.
```

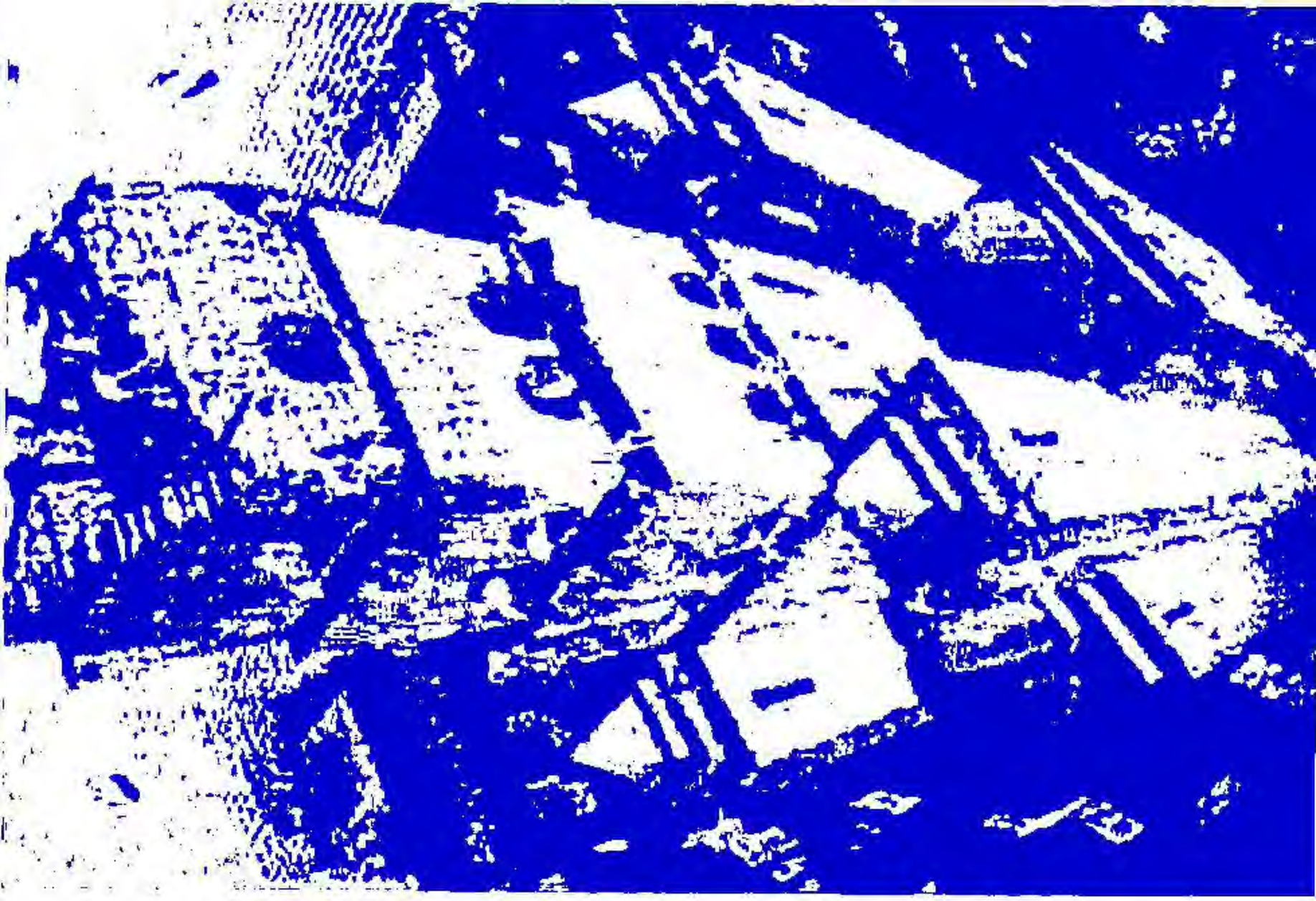


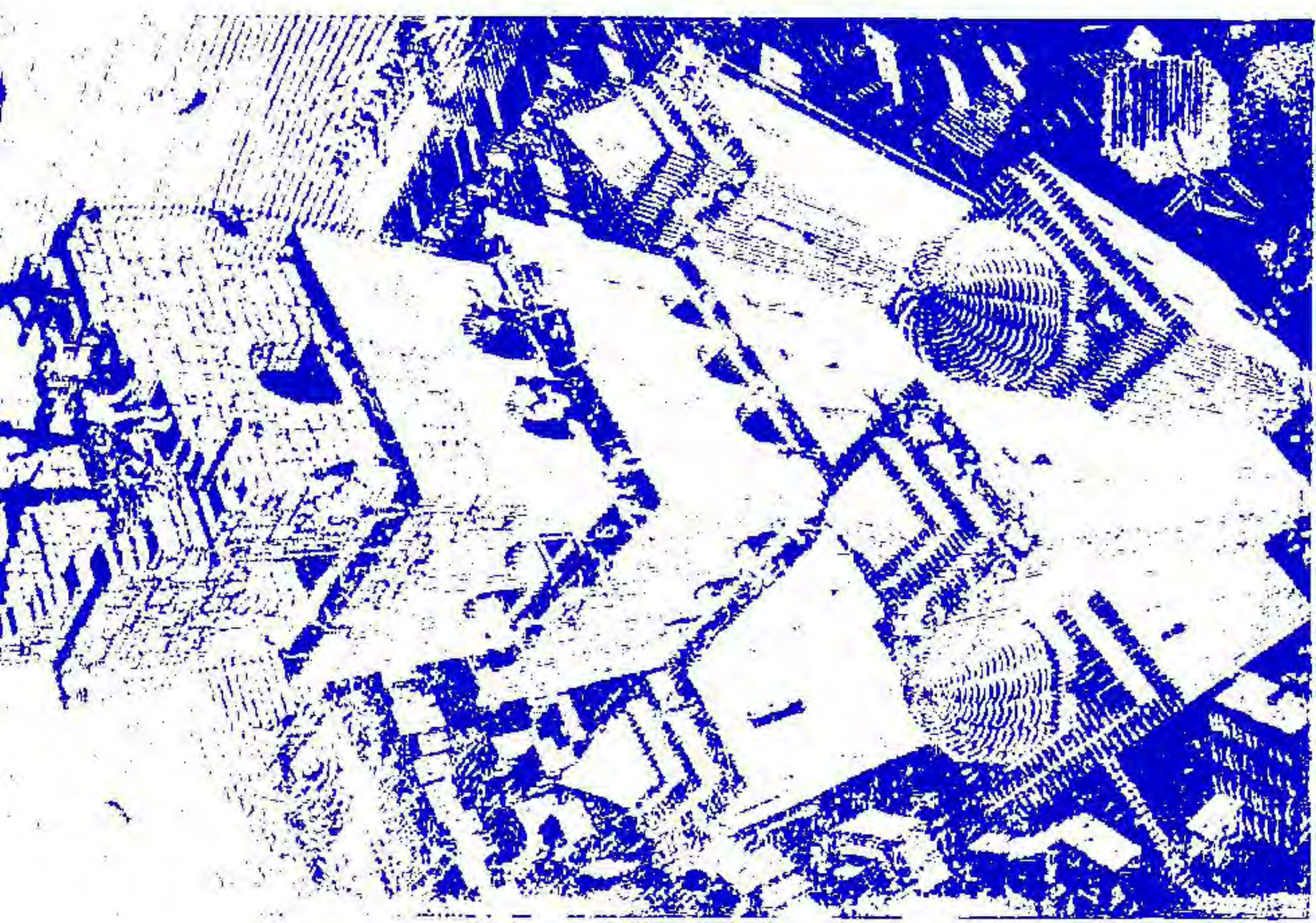












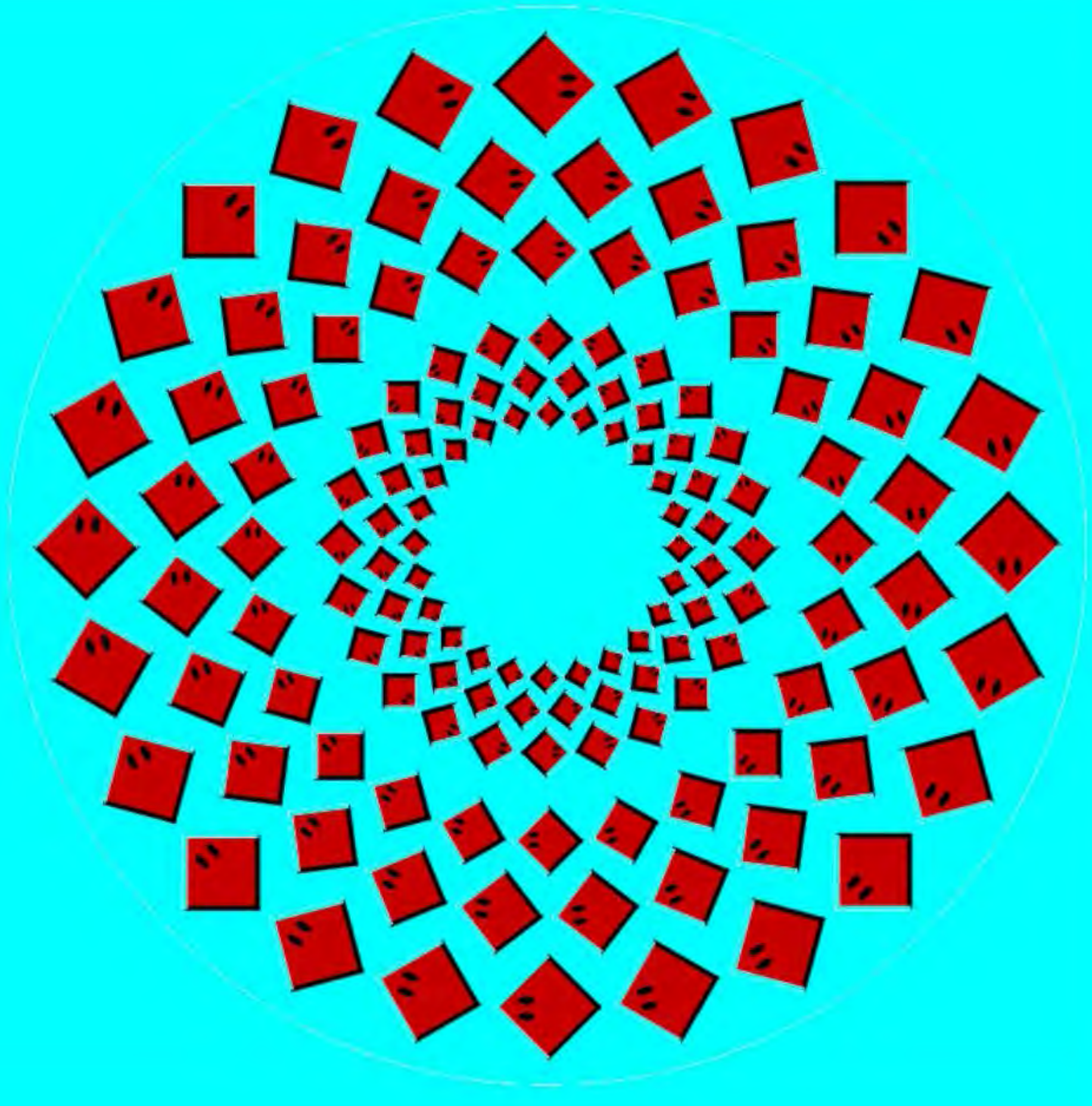


Image Processor

BackGround

0 | 201 | 254

0 | 253 | 253

Enabled

ForeGround

255 | 255 | 255

0 | 253 | 253

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

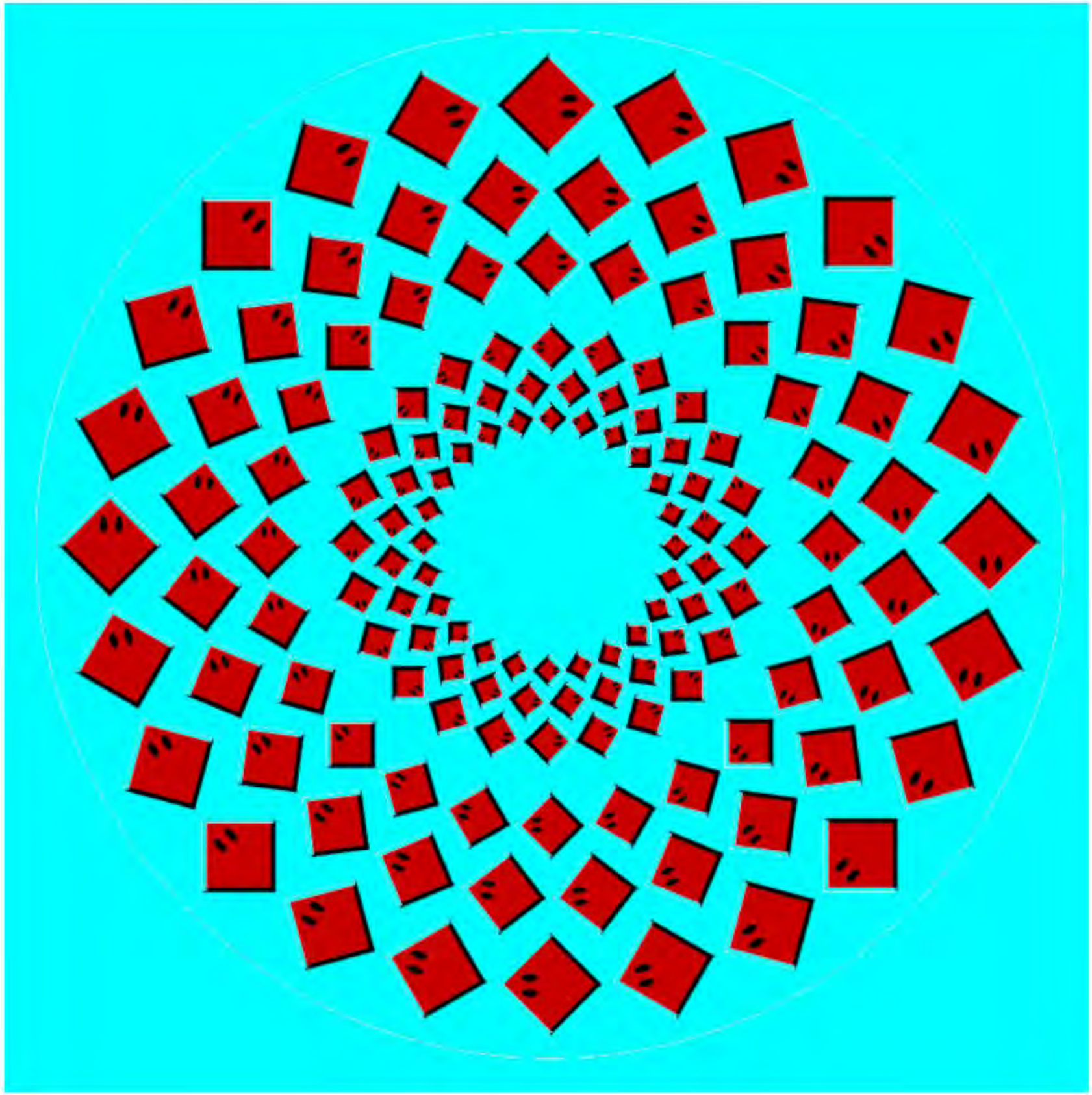


Image Processor

BackGround

0 | 201 | 254

0 | 255 | 255

Enabled

ForeGround

255 | 255 | 255

0 | 255 | 255

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

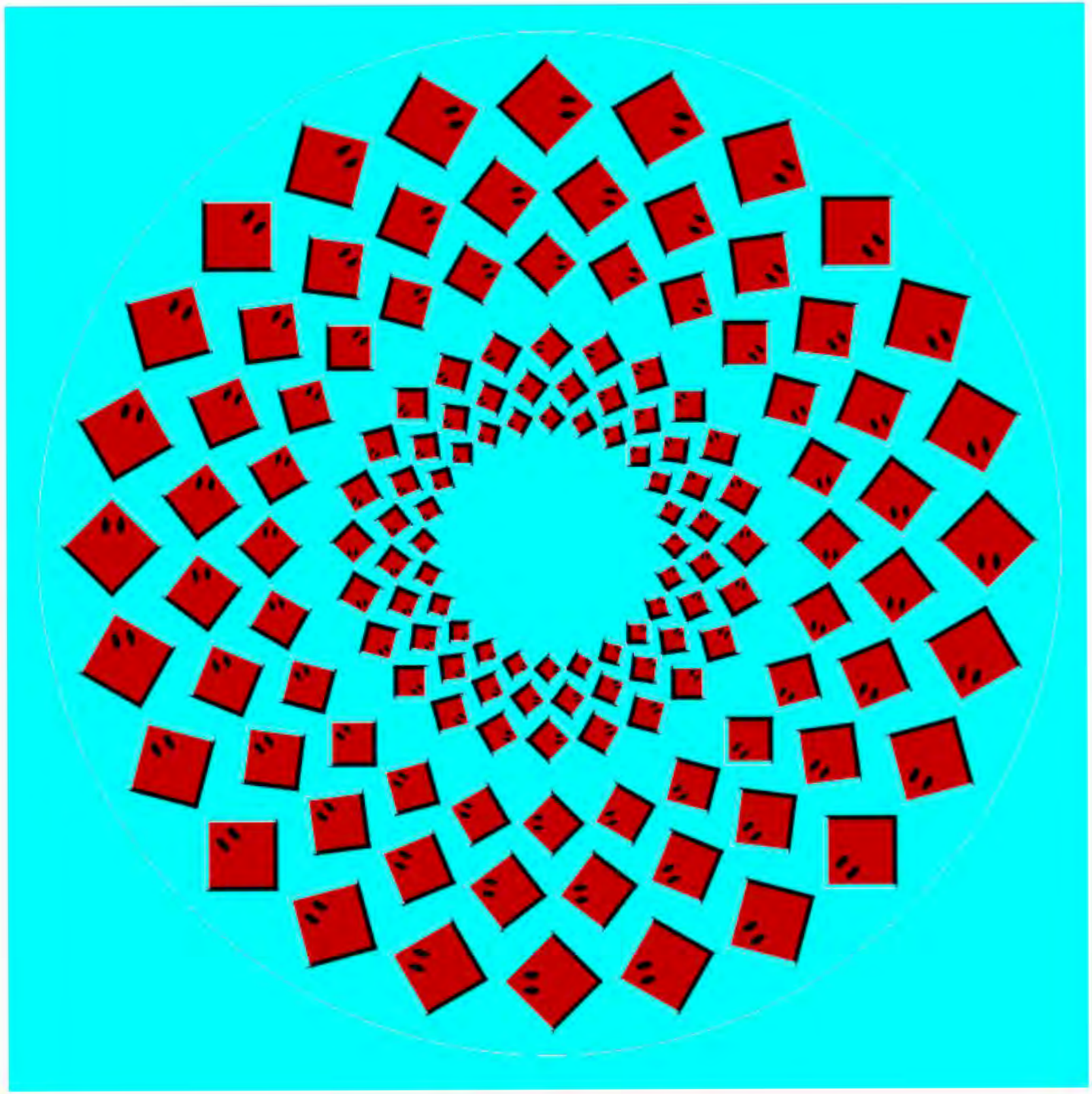


Image Processor

BackGround

0 | 201 | 254

0 | 255 | 0

Enabled

ForeGround

255 | 255 | 255

0 | 255 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

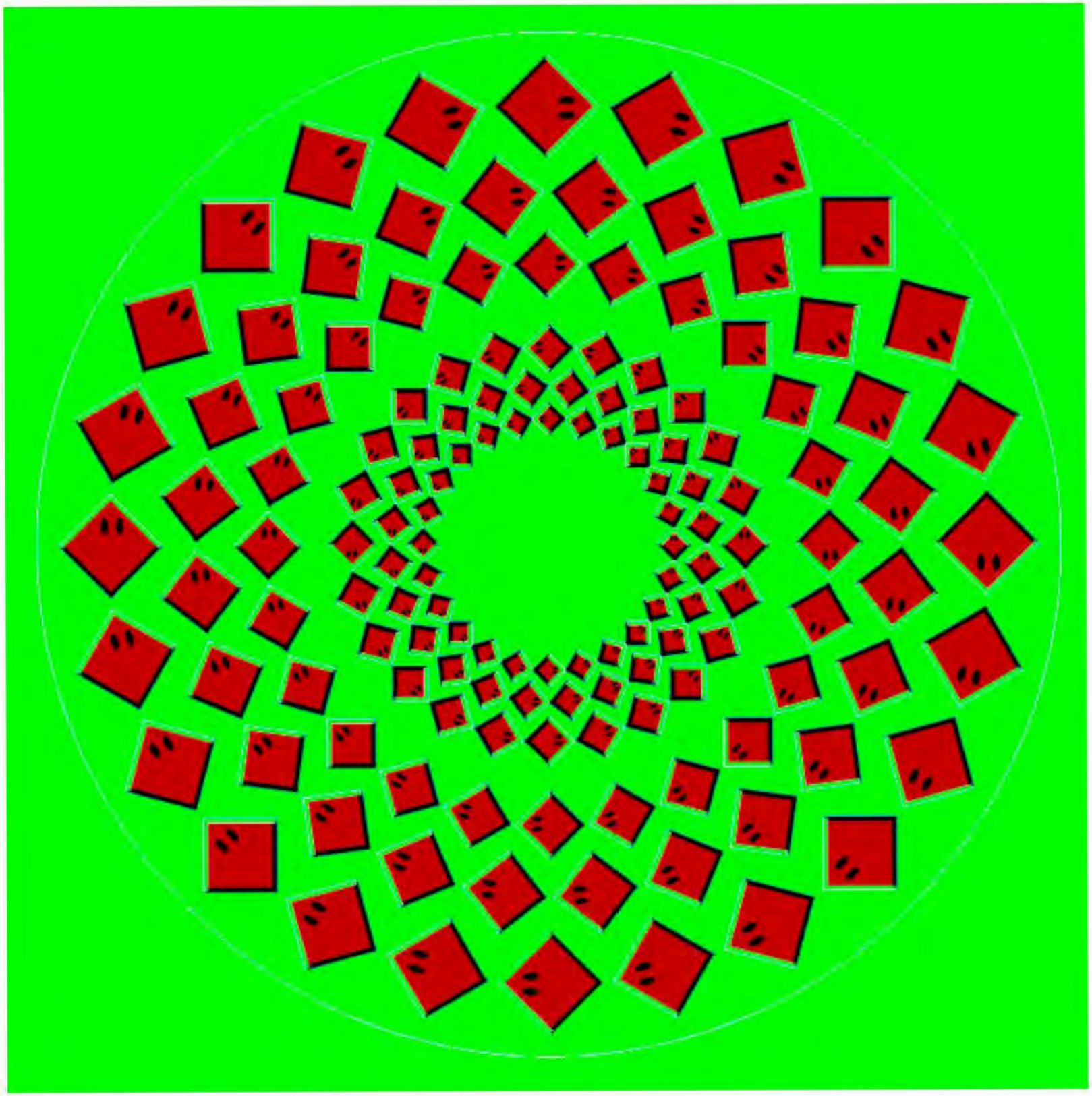


Image Processor

BackGround

0 | 201 | 254

0 | 230 | 0

Enabled

ForeGround

255 | 255 | 255

0 | 230 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

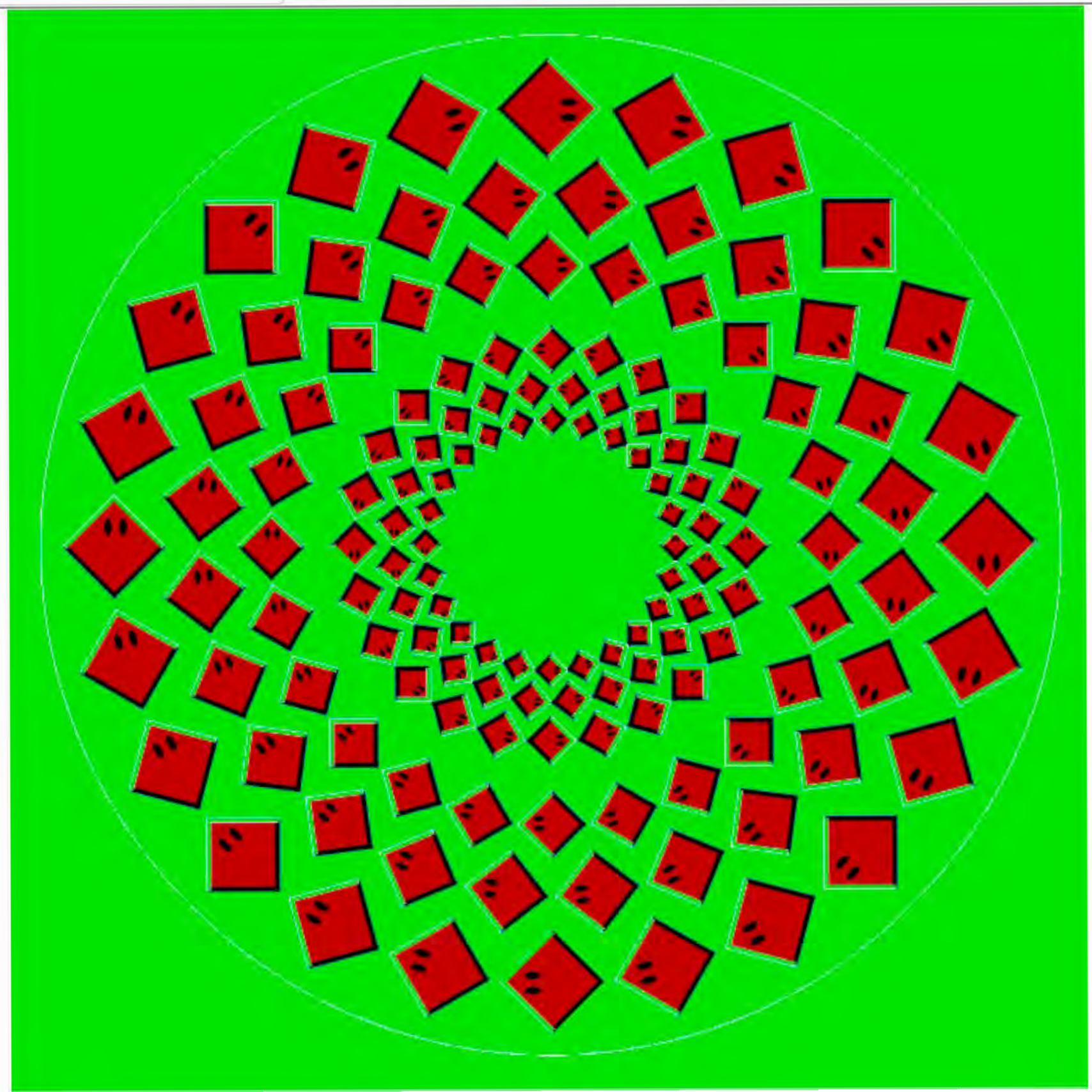


Image Processor

BackGround

0 | 201 | 254

0 | 250 | 0

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

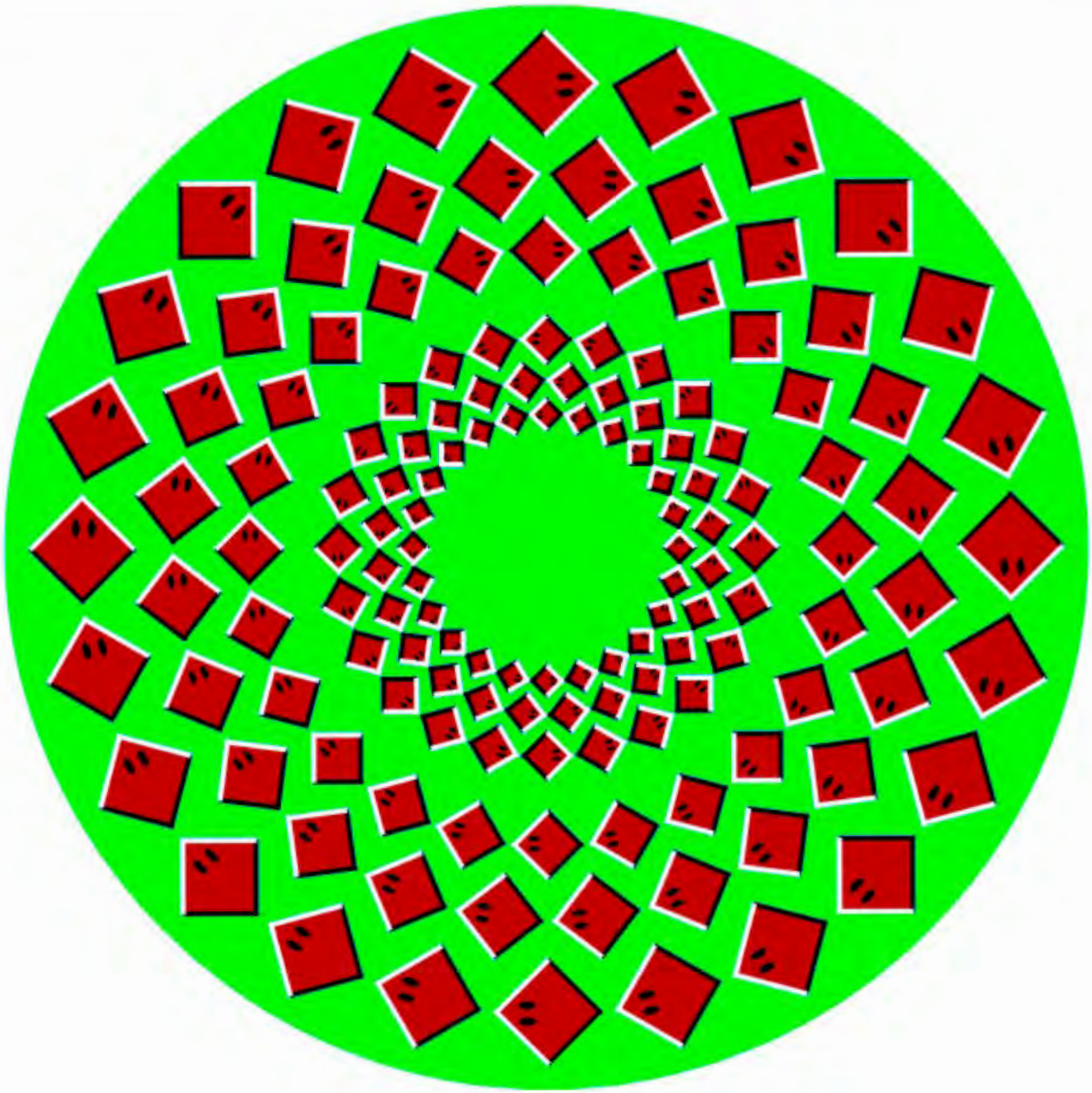


Image Processor

BackGround

0 | 201 | 254

250 | 250 | 0

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

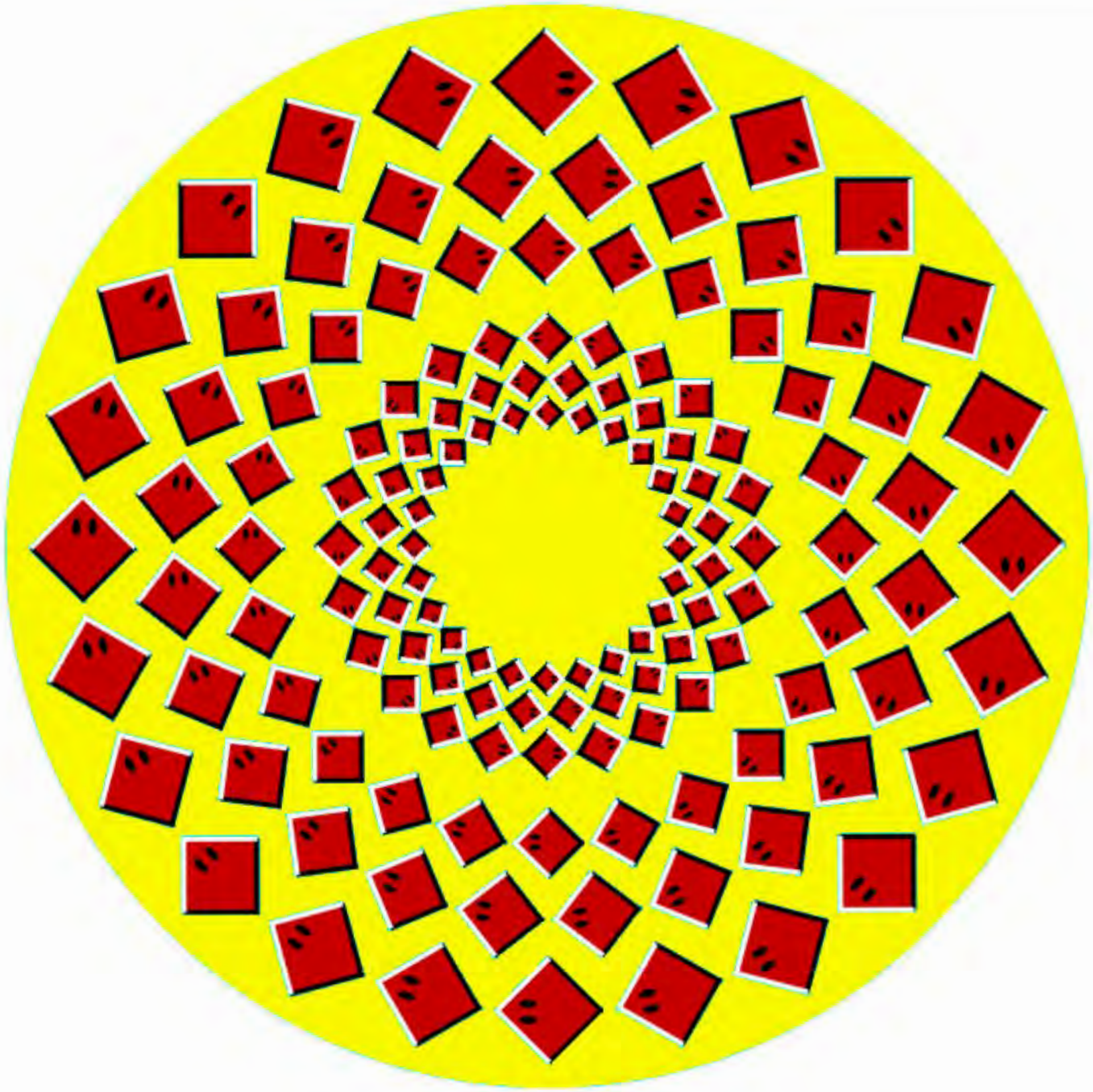
Rotate CW | Rotate CCW

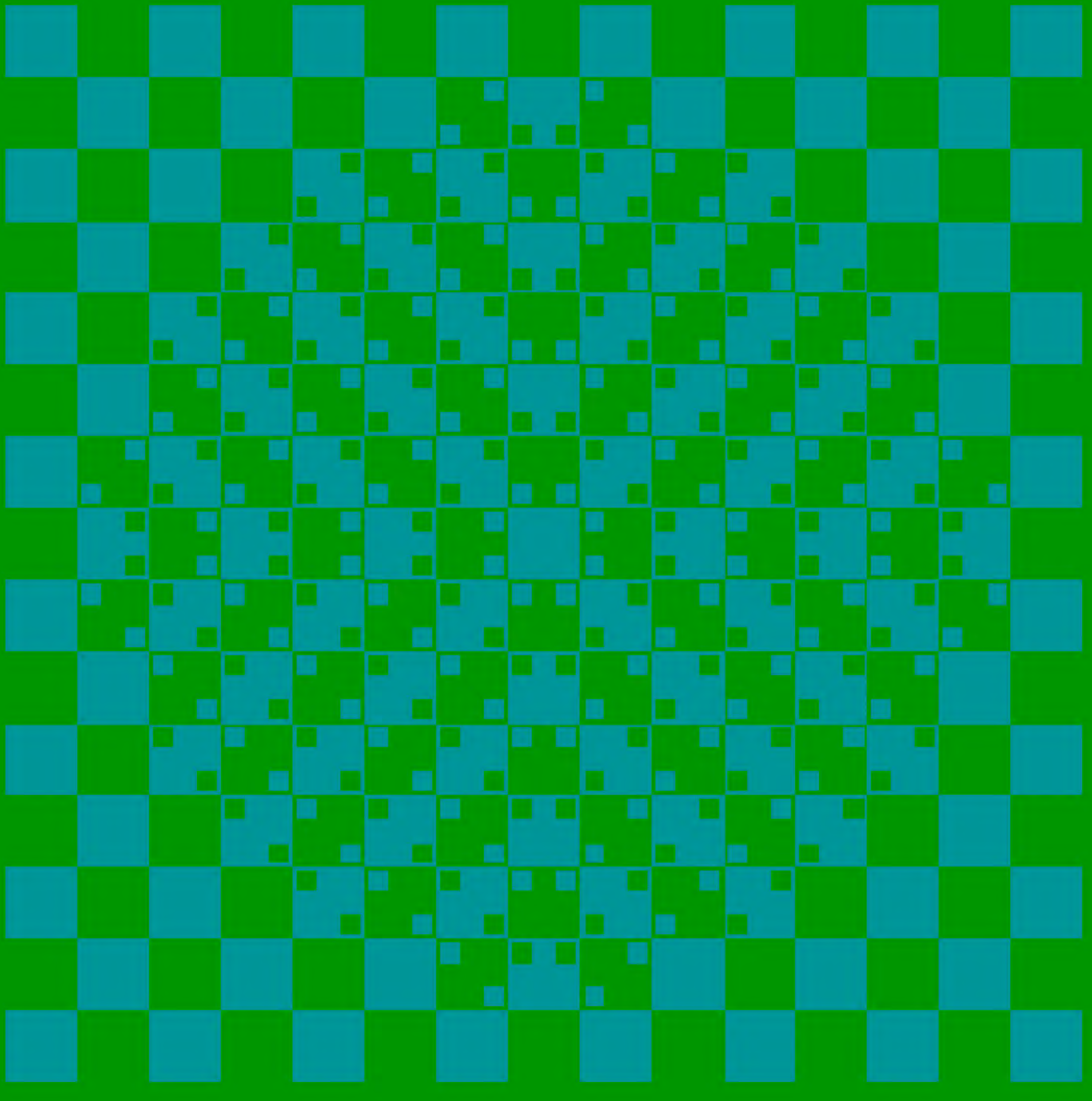
0°

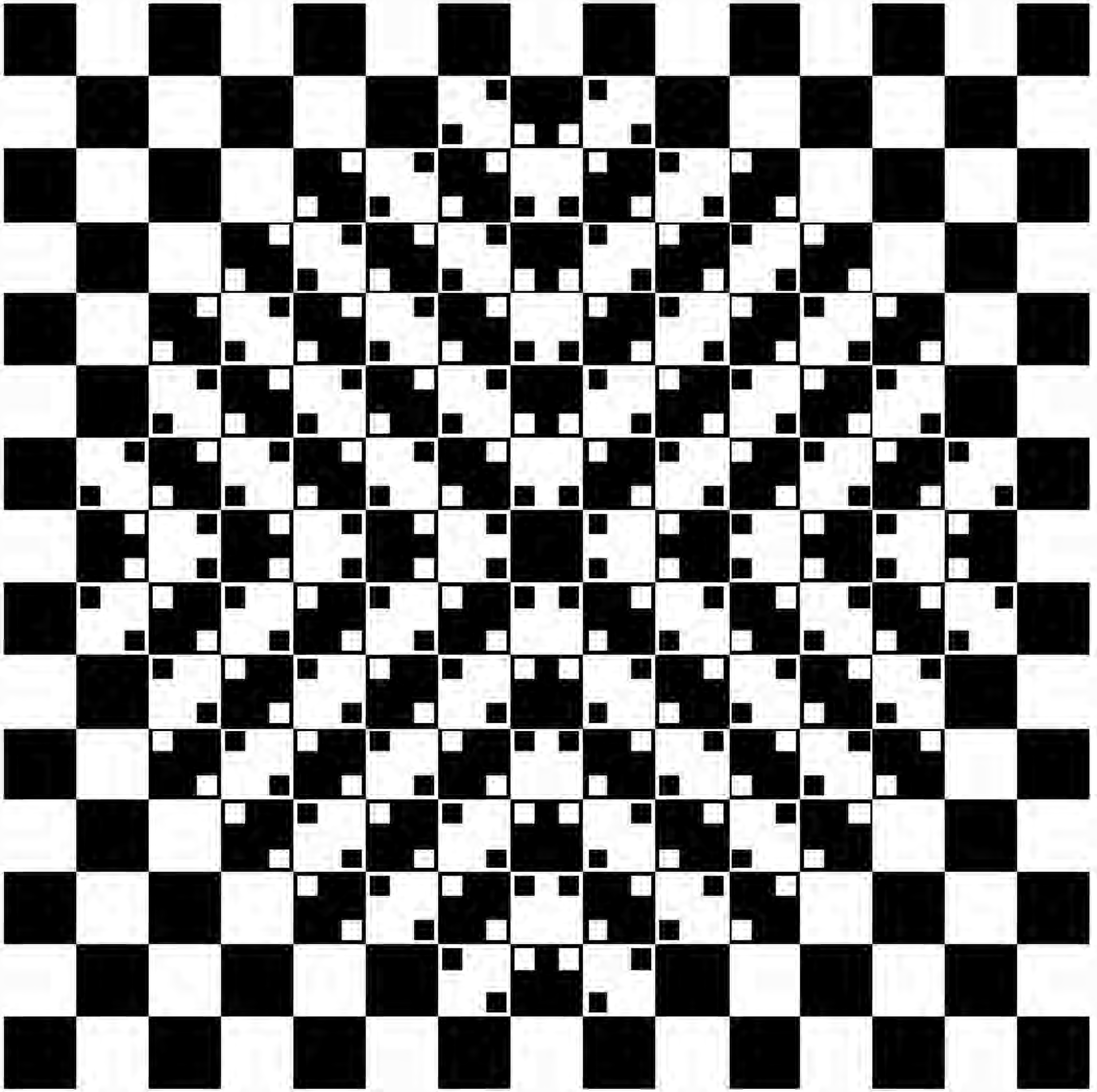
Save Image

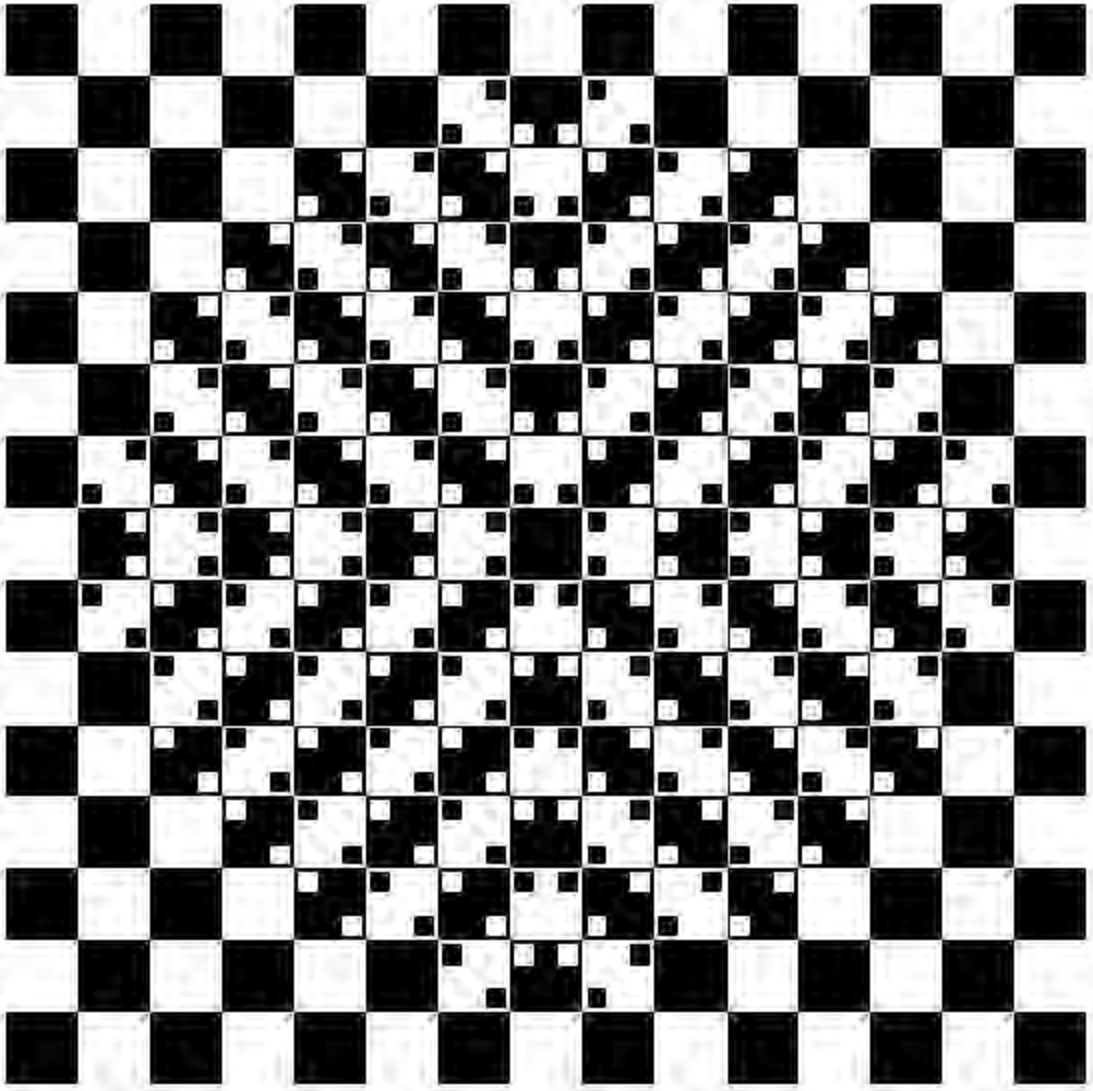
Load Image

Reload Image









```
unit CalcBlocks;


interface
Procedure CalcBlock1(VAR Block1Avg,Block1Std : Double);
Procedure CalcBlock2(VAR Block2Avg,Block2Std : Double);
Procedure CalcBlock3(VAR Block3Avg,Block3Std : Double);
Procedure CalcBlock4(VAR Block4Avg,Block4Std : Double);
Procedure CalcBlock5(VAR Block5Avg,Block5Std : Double);
Procedure GetStd24(Var Block24Std : Double);
Procedure GetStd35(Var Block35Std : Double);

implementation

uses ArrayFunctions, FileFunctions, Math,IATWinFormUnit;

Procedure CalcBlock1(VAR Block1Avg,Block1Std : Double);
VAR Loopstreet,IsValid, Max : Integer;
   TempAvg,TAVG,VarianceT,Variance  : Double;
BEGIN
  Max:=BT1;
  IsValid:=0;
  Block1Avg :=0;
  FOR Loopstreet:=1 TO Max DO
    Begin
      If (Block1Arr[Loopstreet]<=10000) Then
        Begin
          IsValid:=IsValid+1;
          TempB2[IsValid]:=Block1Arr[Loopstreet];
          Block1Avg:=Block1Avg+Block1Arr[Loopstreet];
        End;
    End;
    TempAvg :=Round(Block1Avg/IsValid);
    IF (IsValid<Max) THEN
      BEGIN
        For LoopStreet:=(IsValid+1) TO Max DO
        BEGIN
          TempB2[LoopStreet]:=TempAvg;
        End;
      End;
      TAVG:=0;
  VarianceT:=0;
    For LoopStreet:=1 TO Max Do
      BEGIN
        TAVG:=TAVG+(TempB2[LoopStreet]);
      End;
    Block1Avg:= TAVG/Max;
     For LoopStreet :=1 TO Max DO
      Begin
        Variance:=(Block1Avg - TempB2[LoopStreet]);
        VarianceT:=VarianceT+(Variance*Variance);
      End;
      VarianceT:=(VarianceT/(Max-1));
```

```
    Block1Std:=Round(Sqrt(VarianceT));
END;

Procedure CalcBlock2(VAR Block2Avg,Block2Std : Double);
VAR Loopstreet,IsValid, Max : Integer;
   TempAvg,TAVG,VarianceT,Variance : Double;
BEGIN
  Max:=BT2;
  IsValid:=0;
  Block2Avg :=0;
  FOR Loopstreet:=1 TO Max DO
    Begin
     If (Block2Arr[Loopstreet]<=10000) Then
       Begin
         IsValid:=IsValid+1;
         TempB2[IsValid]:=Block2Arr[Loopstreet];
         Block2Avg:=Block2Avg+Block2Arr[Loopstreet];
       End;
    End;
    TempAvg :=Round(Block2Avg/IsValid);
    IF (IsValid<Max) THEN
     BEGIN
       For LoopStreet:=(IsValid+1) TO Max DO
       BEGIN
         TempB2[LoopStreet]:=TempAvg;
       End;
     End;
     // initialise the variables
     TAVG:=0;
     VarianceT:=0;
  // calculate the block average
    For LoopStreet:=1 TO Max Do
     BEGIN
       TAVG:=TAVG+(TempB2[LoopStreet]);
     End;
    Block2Avg:= TAVG/Max;
    // calculate the block std
    For LoopStreet :=1 TO Max DO
     Begin
       Variance:=(Block2Avg - TempB2[LoopStreet]);
       VarianceT:=VarianceT+(Variance*Variance);
     End;
     VarianceT:=(VarianceT/(Max-1));
     Block2Std:=Round(Sqrt(VarianceT));
END;

Procedure CalcBlock3(VAR Block3Avg,Block3Std : Double);
VAR Loopstreet,IsValid, Max : Integer;
   TempAvg,TAVG,VarianceT,Variance : Double;
BEGIN
Max:=BT3;
  IsValid:=0;
  Block3Avg :=0;
  FOR Loopstreet:=1 TO Max DO
```

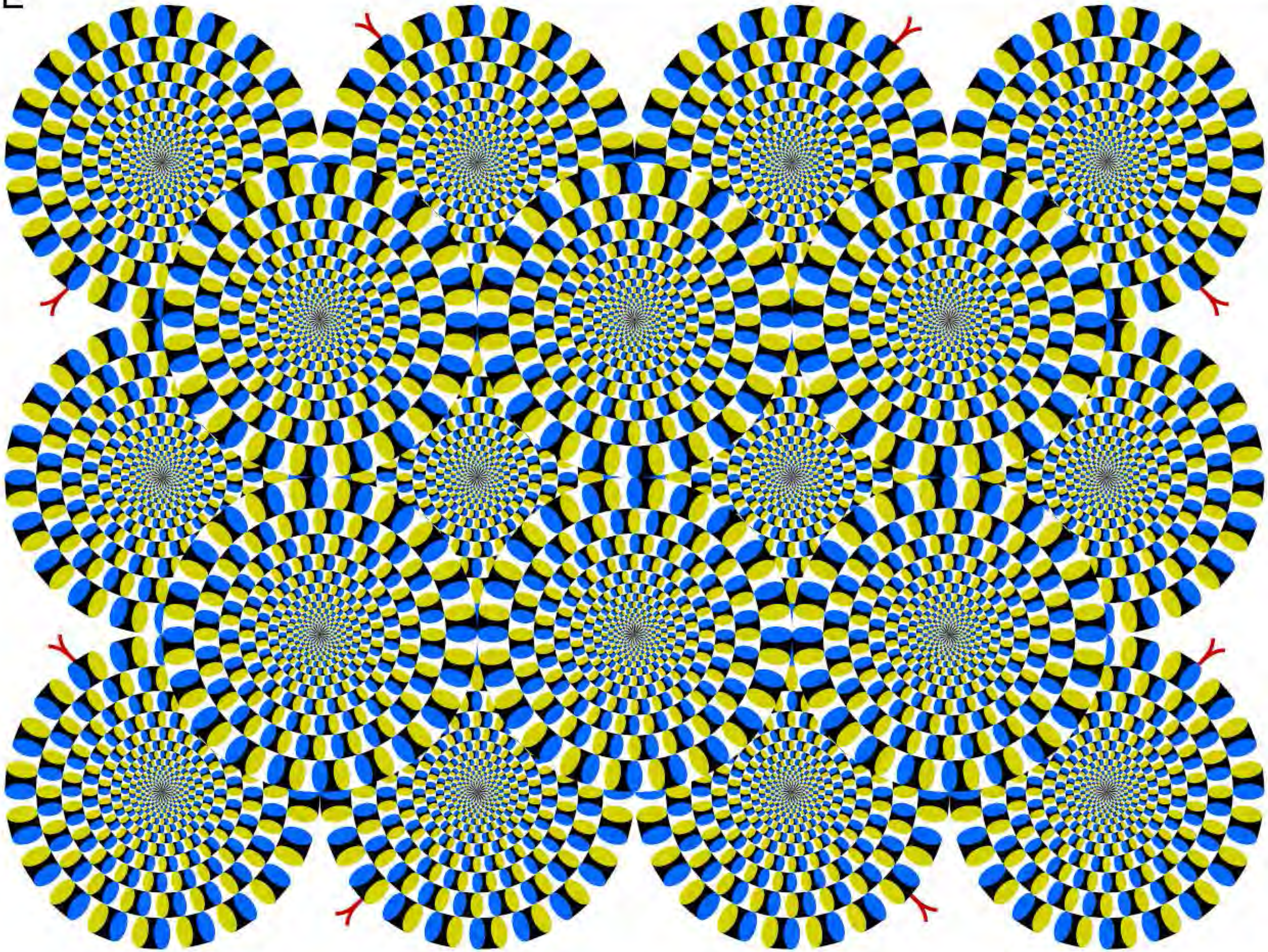
```
   Begin
     If (Block3Arr[Loopstreet]<=10000) Then
       Begin
         IsValid:=IsValid+1;
         TempB3[IsValid]:=Block3Arr[Loopstreet];
         Block3Avg:=Block3Avg+Block3Arr[Loopstreet];
       End;
   End;
   TempAvg :=Round(Block3Avg/IsValid);
   IF (IsValid<Max) THEN
     BEGIN
       For LoopStreet:=(IsValid+1) TO Max DO
       BEGIN
         TempB3[LoopStreet]:=TempAvg;
       End;
     End;
 // initialise the variables
     TAVG:=0;
     VarianceT:=0;
 // calculate the block average
   For LoopStreet:=1 TO Max Do
     BEGIN
       TAVG:=TAVG+(TempB3[LoopStreet]);
     End;
   Block3Avg:= TAVG/Max;
// calculate the block std
   For LoopStreet :=1 TO Max DO
     Begin
       Variance:=(Block3Avg - TempB3[LoopStreet]);
       VarianceT:=VarianceT+(Variance*Variance);
     End;
     VarianceT:=(VarianceT/(Max-1));
     Block3Std:=Round(Sqrt(VarianceT));
END;

Procedure CalcBlock4(VAR Block4Avg,Block4Std : Double);
VAR Loopstreet,IsValid, Max : Integer;
   TempAvg,TAVG,VarianceT,Variance : Double;
BEGIN
 Max:=BT4;
 IsValid:=0;
 Block4Avg :=0;
 FOR Loopstreet:=1 TO Max DO
   Begin
     If (Block4Arr[Loopstreet]<=10000) Then
       Begin
         IsValid:=IsValid+1;
         Tempb4[IsValid]:=Block4Arr[Loopstreet];
         Block4Avg:=Block4Avg+Block4Arr[Loopstreet];
       End;
   End;
   TempAvg :=Round(Block4Avg/IsValid);
   IF (IsValid<Max) THEN
     BEGIN
```

```
      For LoopStreet:=(IsValid+1) TO Max DO
      BEGIN
        Tempb4[LoopStreet]:=TempAvg;
      End;
    End;
    // initialise the variables
    TAVG:=0;
    VarianceT:=0;
 // calculate the block average
   For LoopStreet:=1 TO Max Do
    BEGIN
      TAVG:=TAVG+(TempB4[LoopStreet]);
    End;
   Block4Avg:= TAVG/Max;
// calculate the block std
   For LoopStreet :=1 TO Max DO
    Begin
      Variance:=(Block4Avg - TempB4[LoopStreet]);
      VarianceT:=VarianceT+(Variance*Variance);
    End;
    VarianceT:=(VarianceT/(Max-1));
    Block4Std:=Round(Sqrt(VarianceT));
END;

Procedure CalcBlock5(VAR Block5Avg,Block5Std : Double);
VAR Loopstreet,IsValid, Max : Integer;
   TempAvg,TAVG,VarianceT,Variance : Double;
BEGIN
Max:=BT5;
 IsValid:=0;
 Block5Avg :=0;
 FOR Loopstreet:=1 TO Max DO
   Begin
    If (Block5Arr[Loopstreet]<=10000) Then
      Begin
        IsValid:=IsValid+1;
        Tempb5[IsValid]:=Block5Arr[Loopstreet];
        Block5Avg:=Block5Avg+Block5Arr[Loopstreet];
      End;
   End;
   TempAvg :=Round(Block5Avg/IsValid);
   IF (IsValid<Max) THEN
    BEGIN
      For LoopStreet:=(IsValid+1) TO Max DO
      BEGIN
        Tempb5[LoopStreet]:=TempAvg;
      End;
    End;
    // initialise the variables
    TAVG:=0;
    VarianceT:=0;
 // calculate the block average
   For LoopStreet:=1 TO Max Do
    BEGIN
```

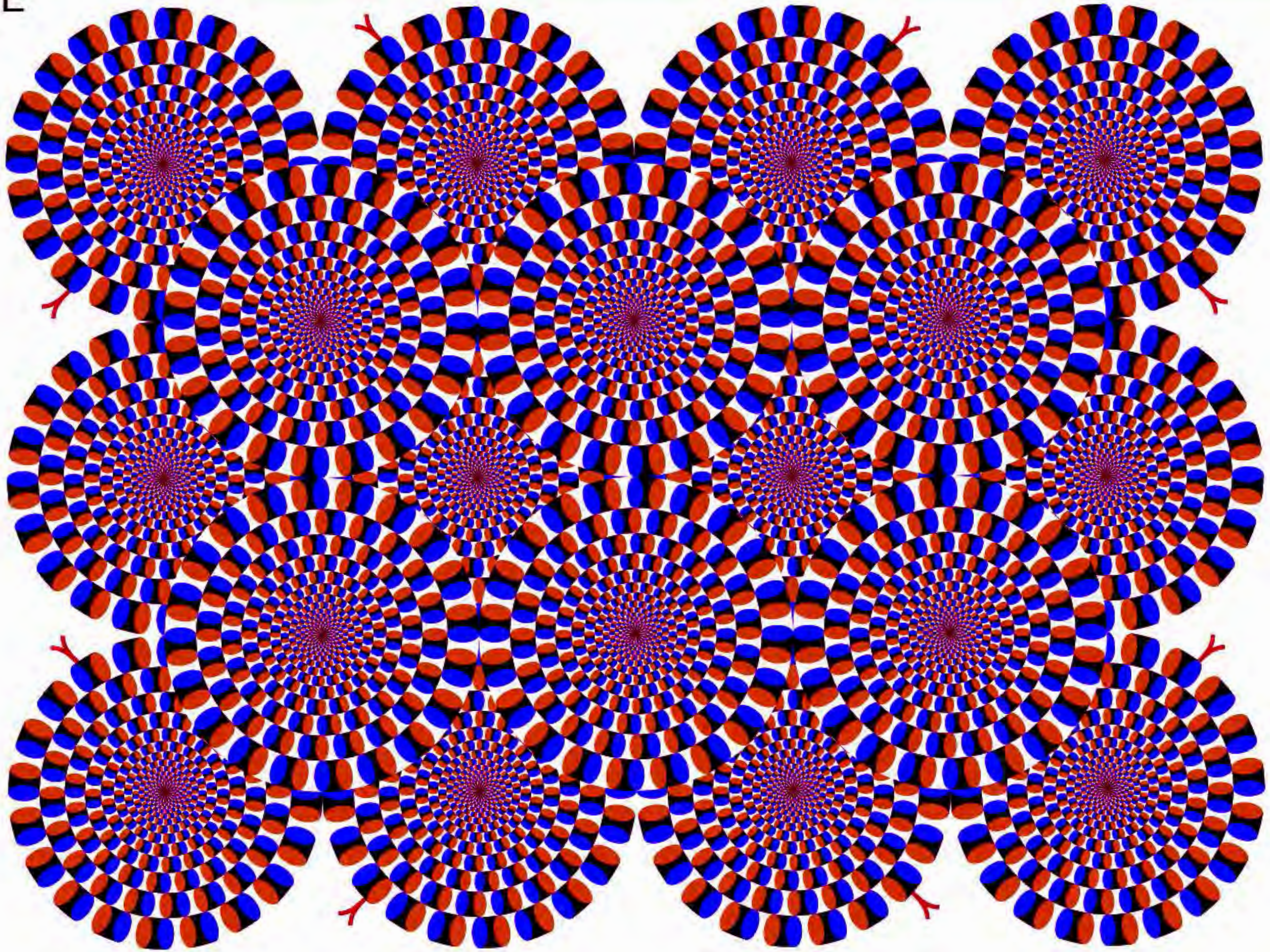
```
      TAVG:=TAVG+(TempB5[LoopStreet]);
     End;
   Block5Avg:= TAVG/Max;
 // calculate the block std
   For LoopStreet :=1 TO Max DO
    Begin
     Variance:=(Block5Avg - TempB5[LoopStreet]);
     VarianceT:=VarianceT+(Variance*Variance);
    End;
    VarianceT:=(VarianceT/(Max-1));
    Block5Std:=Round(Sqrt(VarianceT));
END;

Procedure GetStd24(Var Block24Std : Double);
VAR Loopstreet, Max : Integer;
   TTot,VarianceT,Variance, Mean24 : Double;
BEGIN
Max:=BT2+BT4;
TTot :=0;
  FOR Loopstreet:=1 TO Max DO
   Begin
    TTot:=TTot+Block24Combine[LoopStreet];
   End;
   Mean24:=TTot/Max;
   VarianceT:=0;
  // calculate the block std
   For LoopStreet :=1 TO Max DO
    Begin
     Variance:=(Mean24 - Block24Combine[LoopStreet]);
     VarianceT:=VarianceT+(Variance*Variance);
    End;
    VarianceT:=(VarianceT/(Max-1));
    Block24Std:=Round(Sqrt(VarianceT));
END;

Procedure GetStd35(Var Block35Std : Double);
VAR Loopstreet, Max : Integer;
   TTot,VarianceT,Variance, Mean35 : Double;
BEGIN
Max:=BT3+BT5;
TTot :=0;
  FOR Loopstreet:=1 TO Max DO
   Begin
    TTot:=TTot+Block35Combine[LoopStreet];
   End;
   Mean35:=TTot/Max;
   VarianceT:=0;
  // calculate the block std
   For LoopStreet :=1 TO Max DO
    Begin
     Variance:=(Mean35 - Block35Combine[LoopStreet]);
     VarianceT:=VarianceT+(Variance*Variance);
    End;
    VarianceT:=(VarianceT/(Max-1));
```

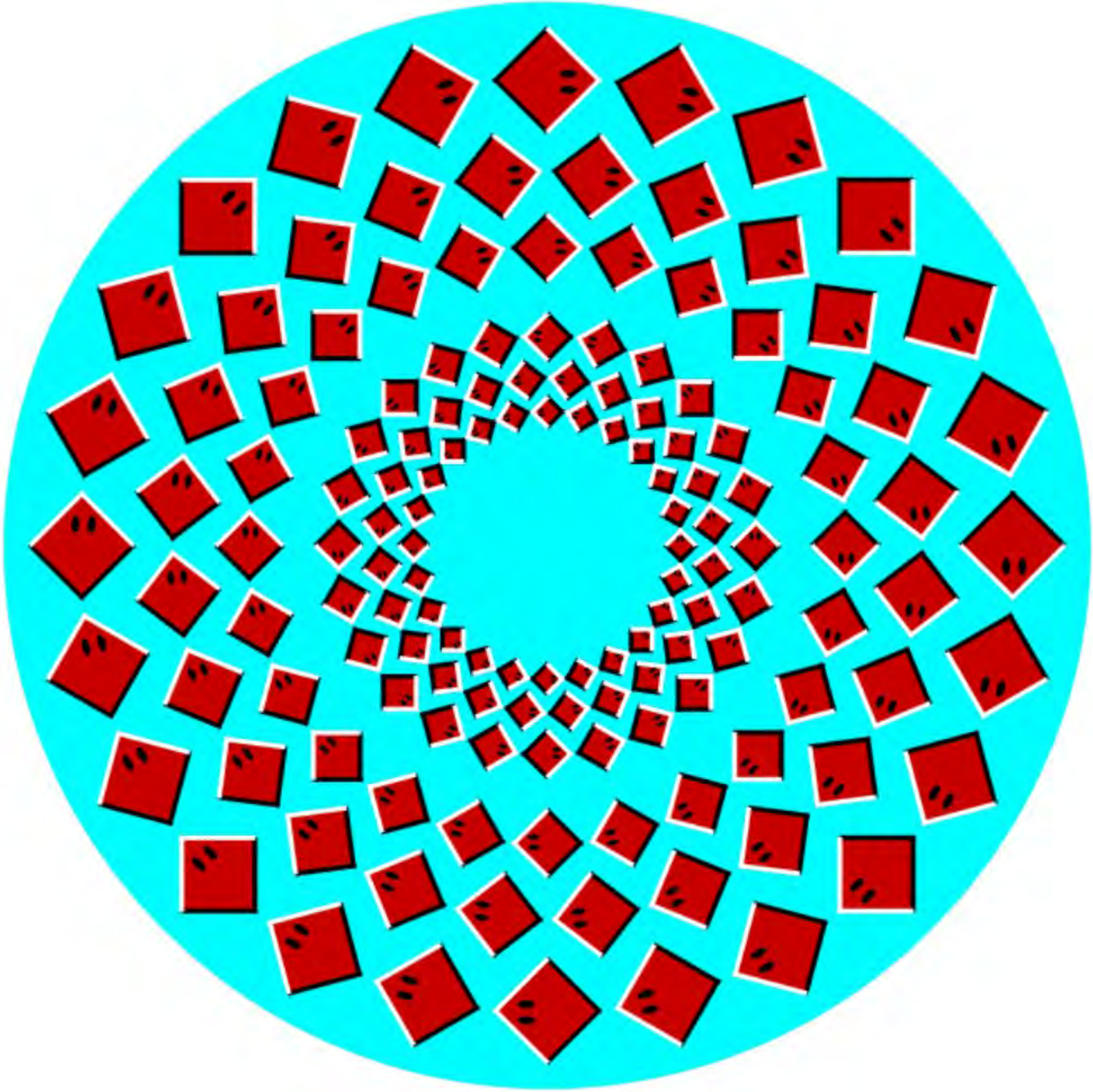
```
    Block35Std:=Round(Sqrt(VarianceT));
END;


end.
```

E



E





```
unit ControlLogic;

interface
Procedure SetOrder(Var B2Second, B3Second : Boolean);
Procedure CyanOrRed;

implementation

Uses FileFunctions,IatWinformUnit;
Procedure TrueFalse(Var Flag: Boolean); External
'TF.dll';

Procedure SetOrder(Var B2Second,B3Second: Boolean);
//Var FF, Modul, LLL : Integer;
Var Flag : Boolean;
Begin
   Flag:=False;
    TrueFalse(Flag);
    if Not Flag then B2Second:=False;
    if Flag  then B2Second:=True;
    TrueFalse(Flag);
    if Not Flag then B3Second:=False;
    if Flag then B3Second:=True;
    Case B2Second Of
     False : Block2Order:=2;
     True : Block2Order:=4;
    End;
    Case B2Second Of
      False : Block4Order:=4;
      True : Block4Order:=2;
    End;
    Case B3Second Of
      False : Block3Order:=3;
      True : Block3Order:=5;
    End;
    Case B3Second Of
      False : Block5Order:=5;
      True : Block5Order:=3;
    End;
    Block1Order:=1;
End;

Procedure CyanOrRed;
Var Flag : Boolean;
Begin
  Flag:=False;
  TrueFalse(Flag);
  IF (Flag) THEN Ccyan:=True;
  IF (Not Flag) THEN Ccyan:=False;
END;
end.
```

Image Processor

BackGround

0 | 201 | 254

0 | 150 | 150

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

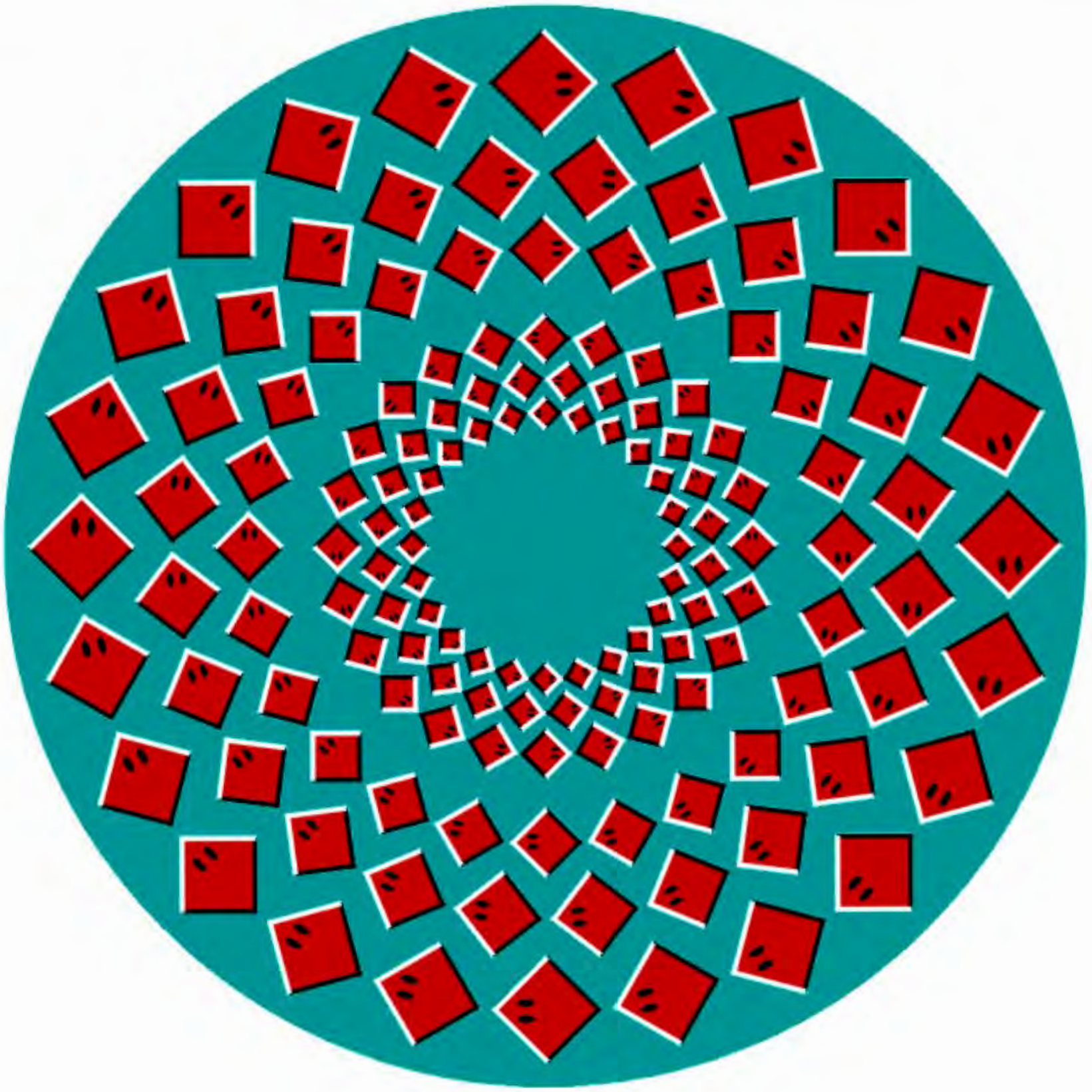


Image Processor

BackGround

50 120 50

0 100 0

Enabled

ForeGround

180 180 180

0 200 200

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

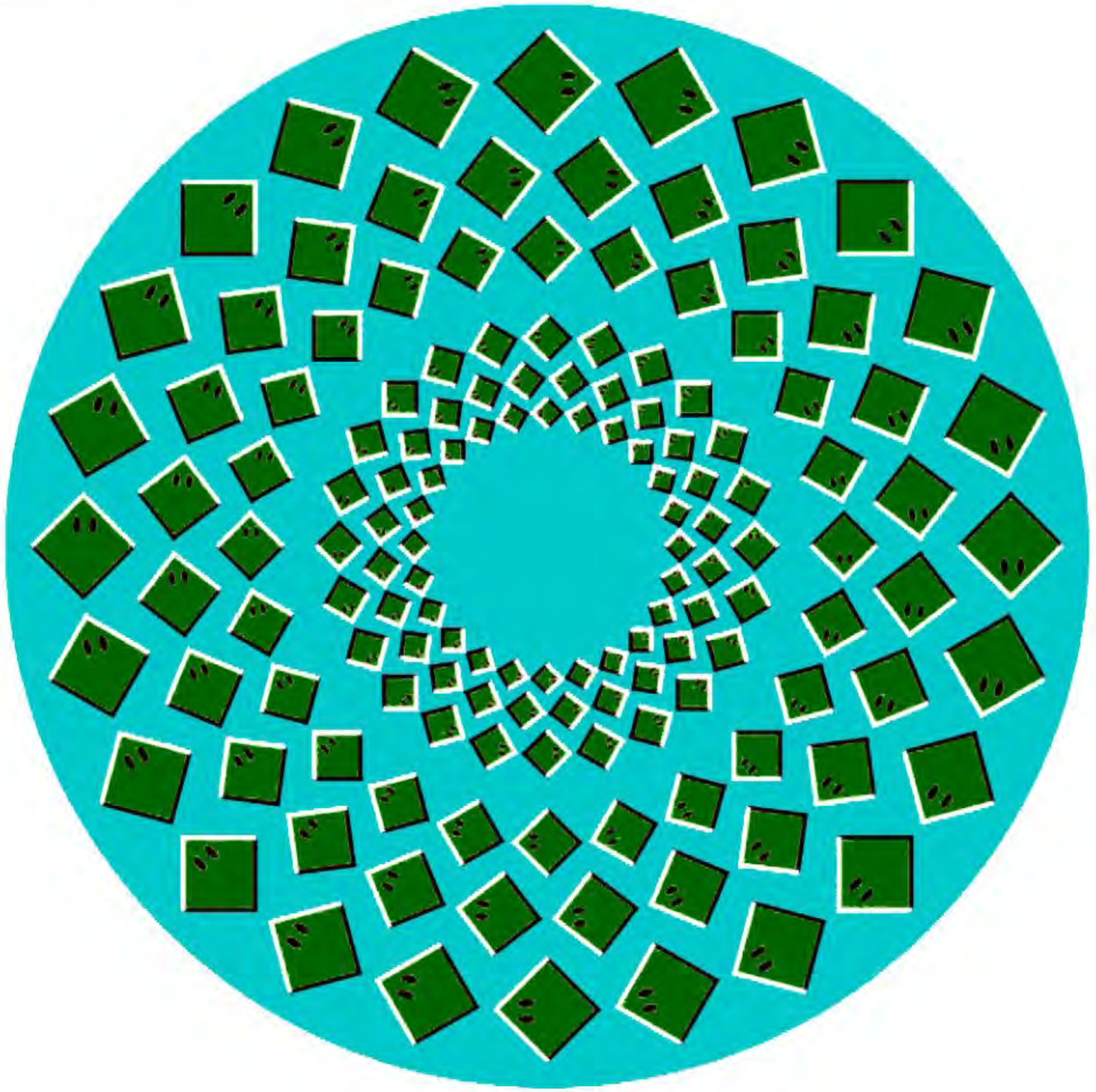


Image Processor

BackGround

50 120 50

0 120 0

Enabled

ForeGround

180 180 180

0 200 200

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

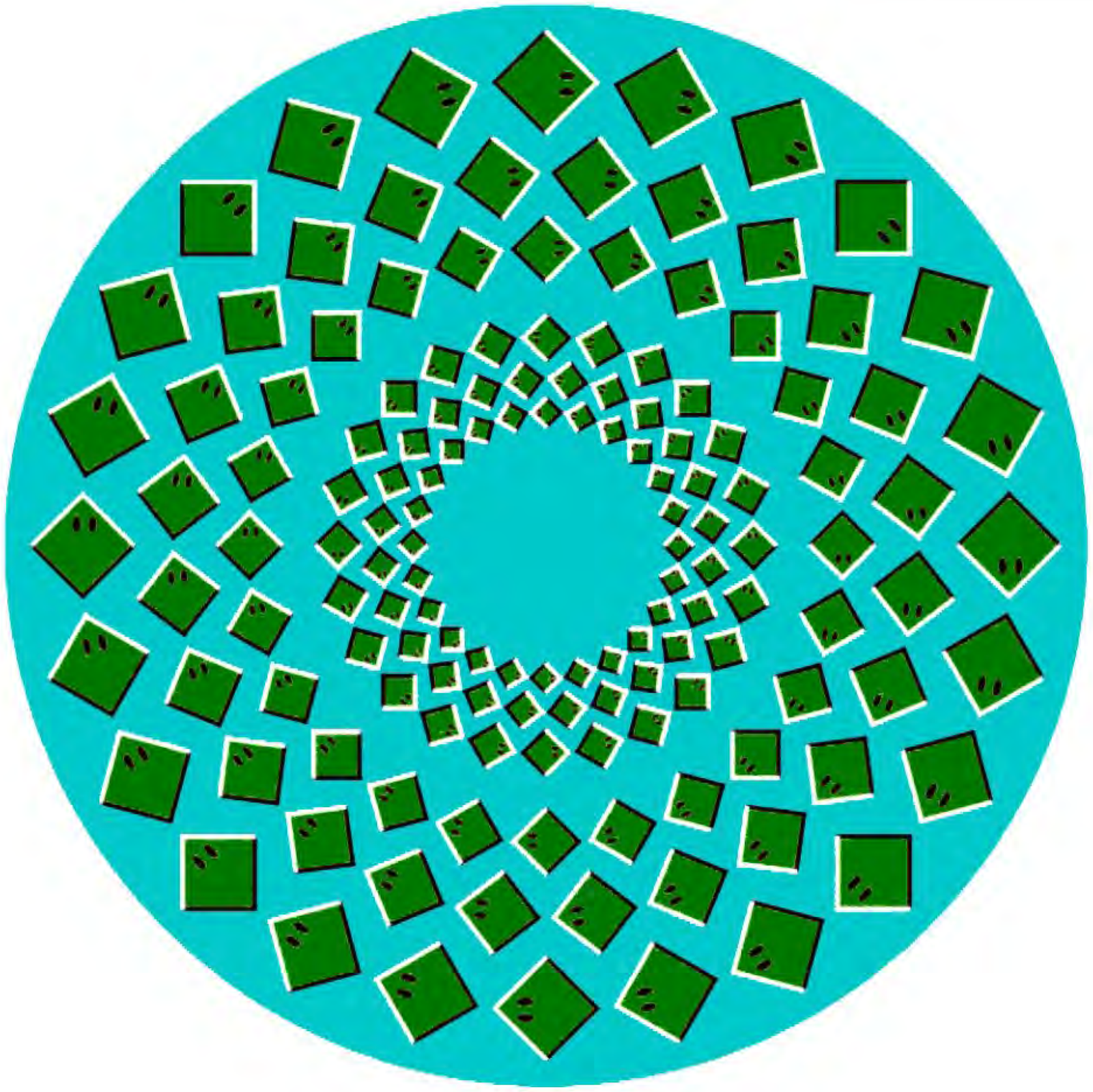


Image Processor

BackGround

50 120 50

0 140 0

Enabled

ForeGround

180 180 180

0 200 200

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

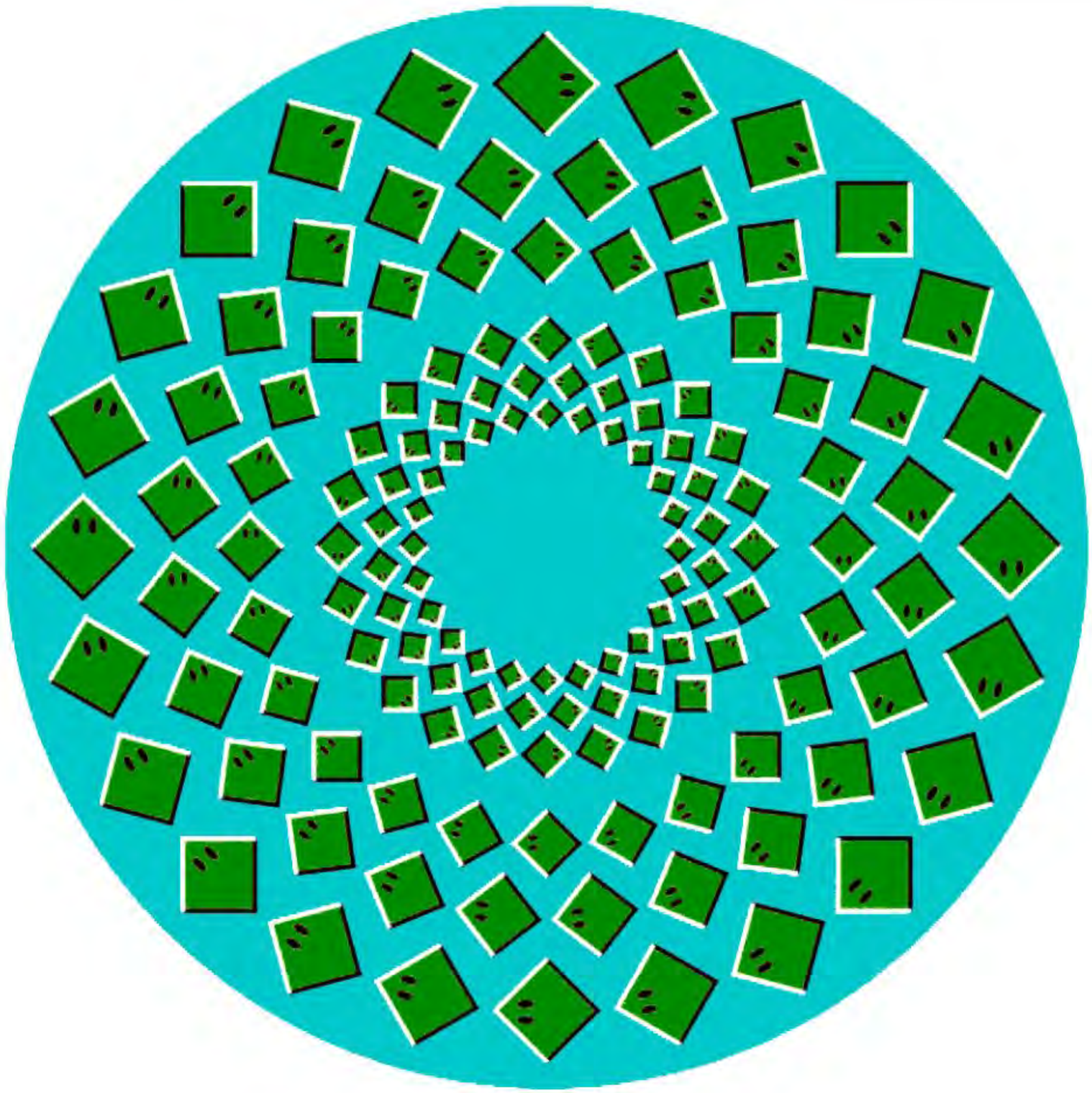


Image Processor

BackGround

50 120 50

0 160 0

Enabled

ForeGround

180 180 180

0 200 200

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

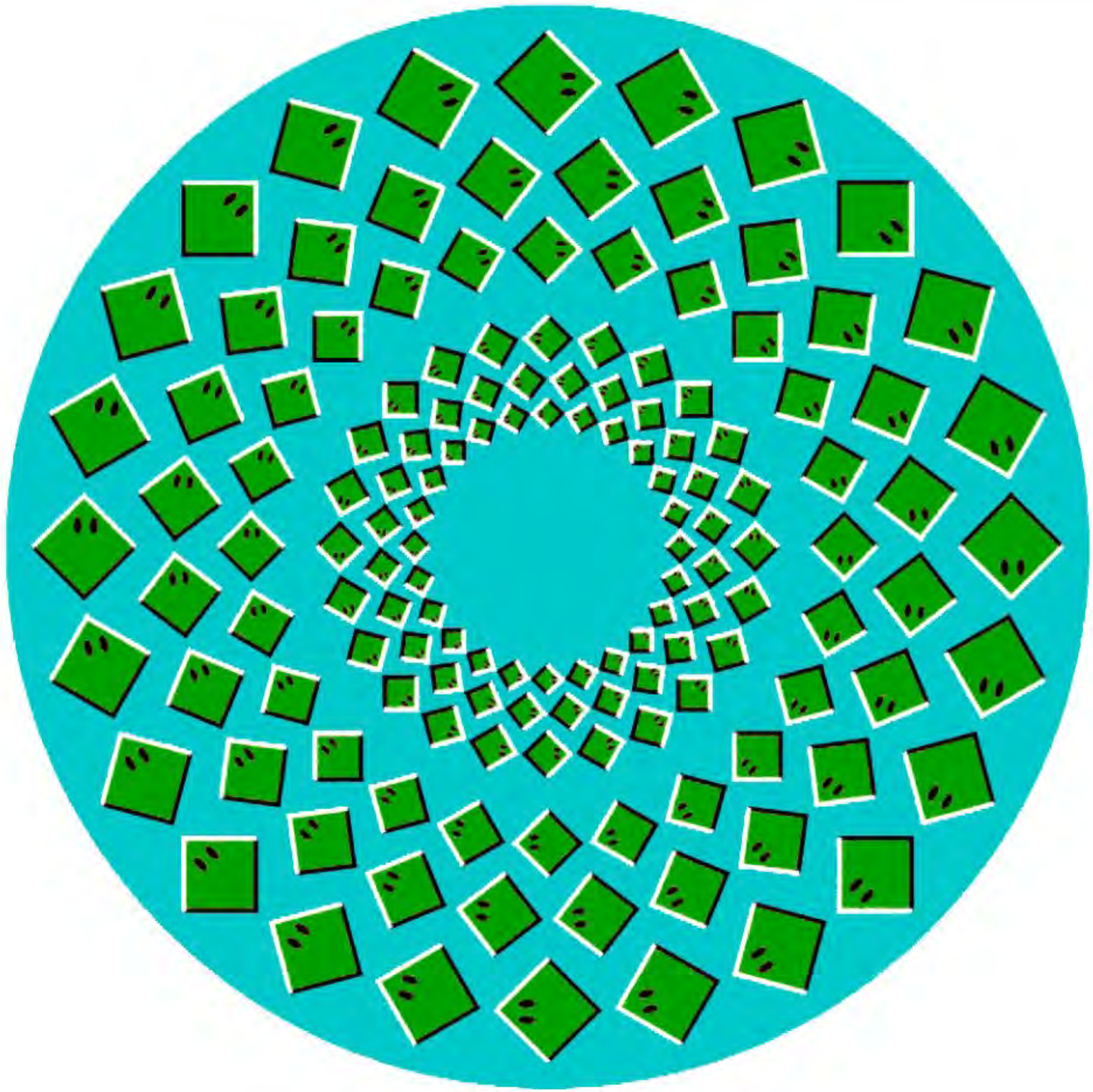


Image Processor

BackGround

50 120 50

0 180 0

Enabled

ForeGround

180 180 180

0 200 200

Enabled

Expander1

Apply

Reload and Apply

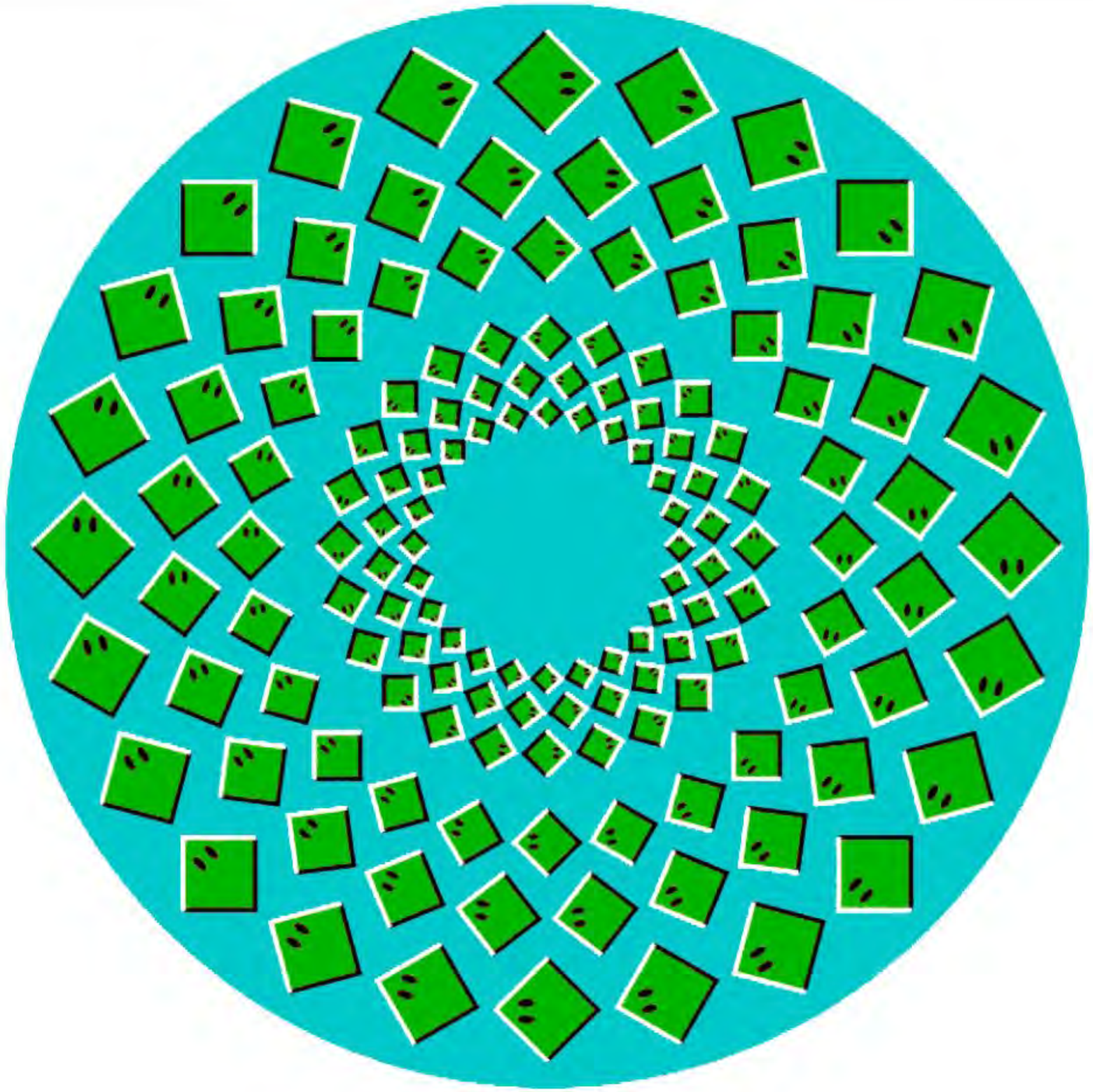
Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image



BackGround

50 120 50

0 200 0

Enabled

ForeGround

180 180 180

0 200 200

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

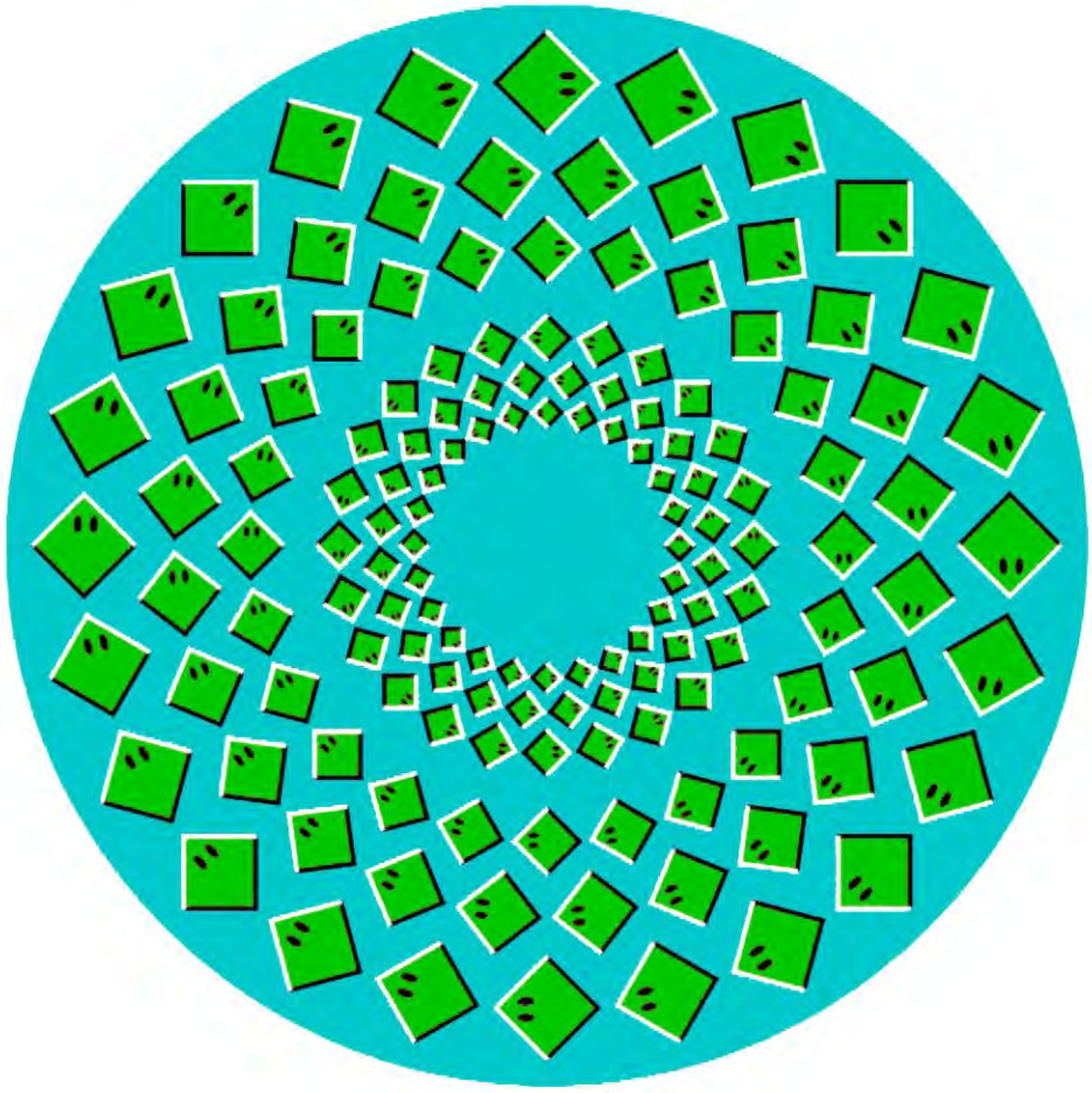


Image Processor

BackGround

50 120 50

0 240 0

Enabled

ForeGround

180 180 180

0 200 200

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

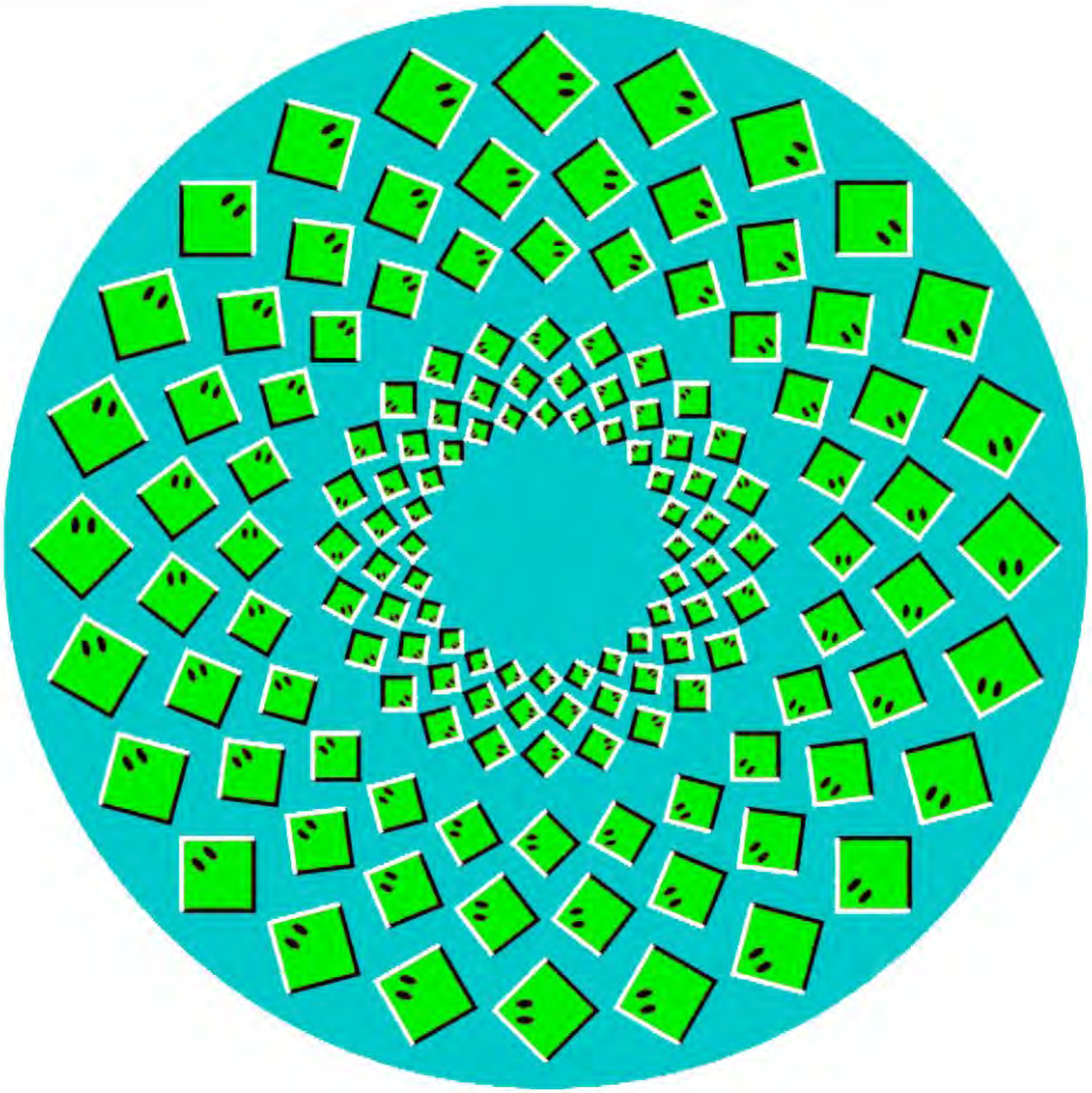


Image Processor

BackGround

50 120 50

0 240 0

Enabled

ForeGround

180 180 180

0 240 240

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

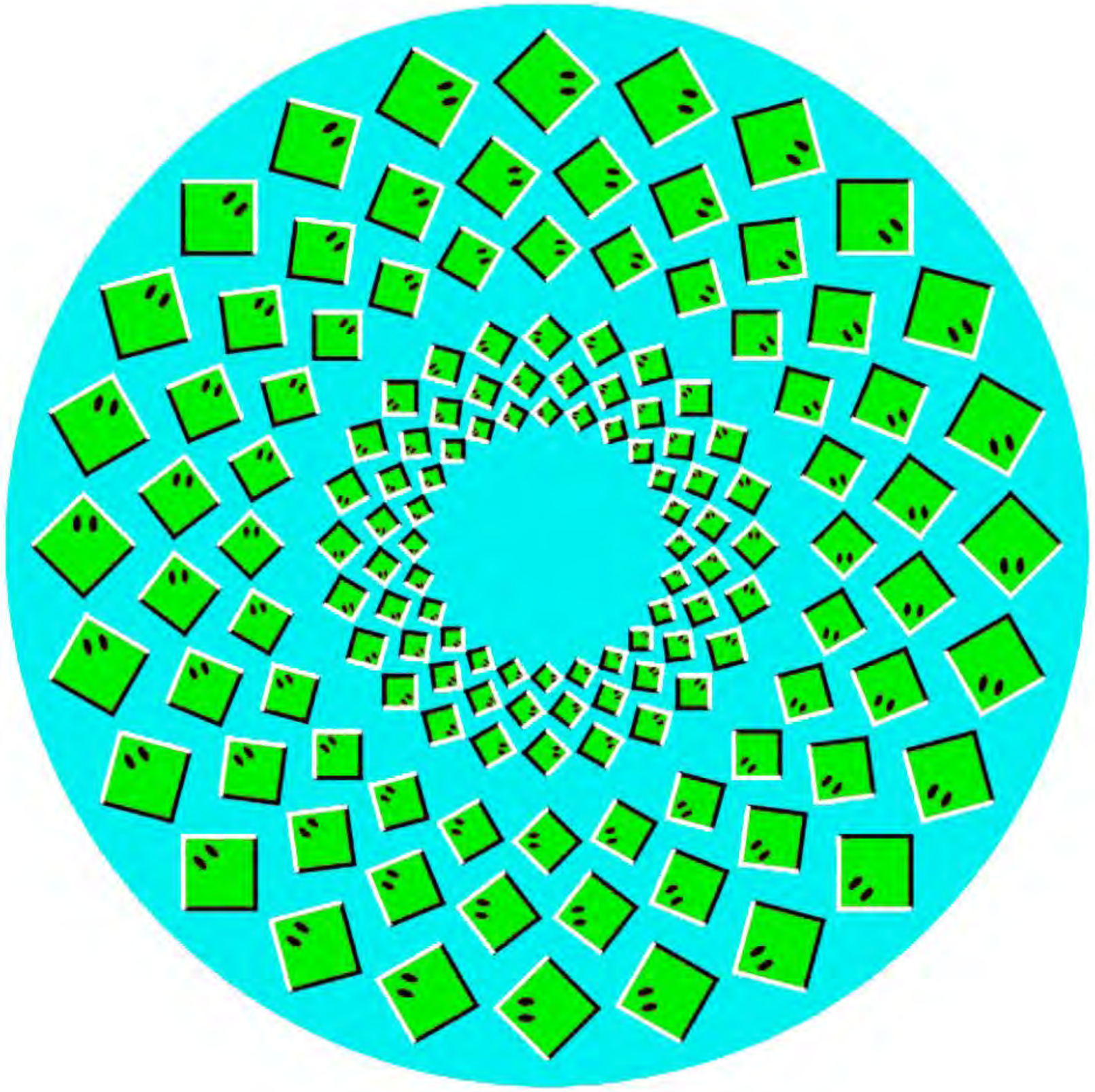


Image Processor

BackGround

50 120 50

0 150 0

Enabled

ForeGround

180 180 180

0 150 150

Enabled

Expander1

Apply

Reload and Apply

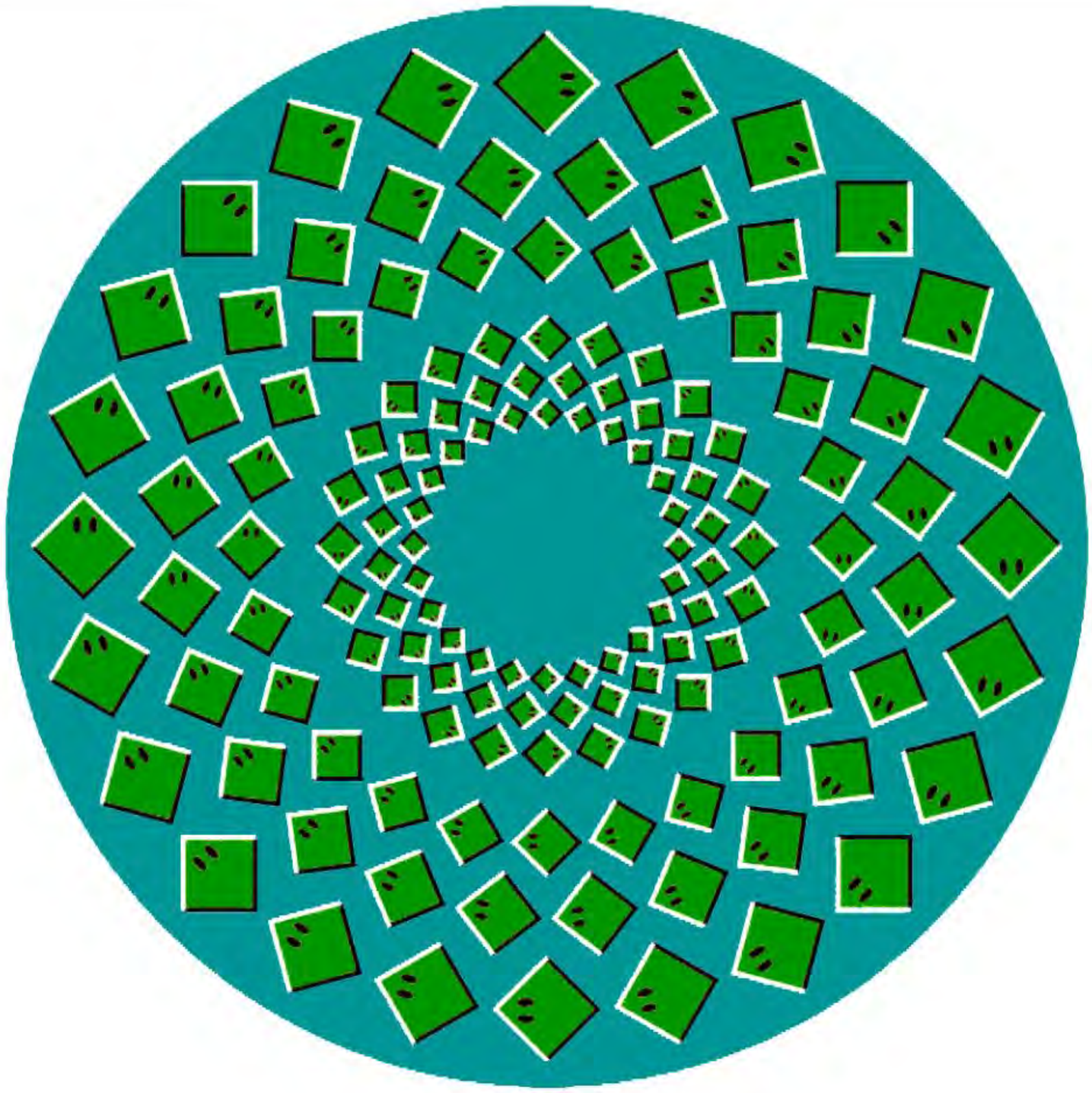
Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image



BackGround

50 120 50

0 150 0

Enabled

ForeGround

180 180 180

220 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

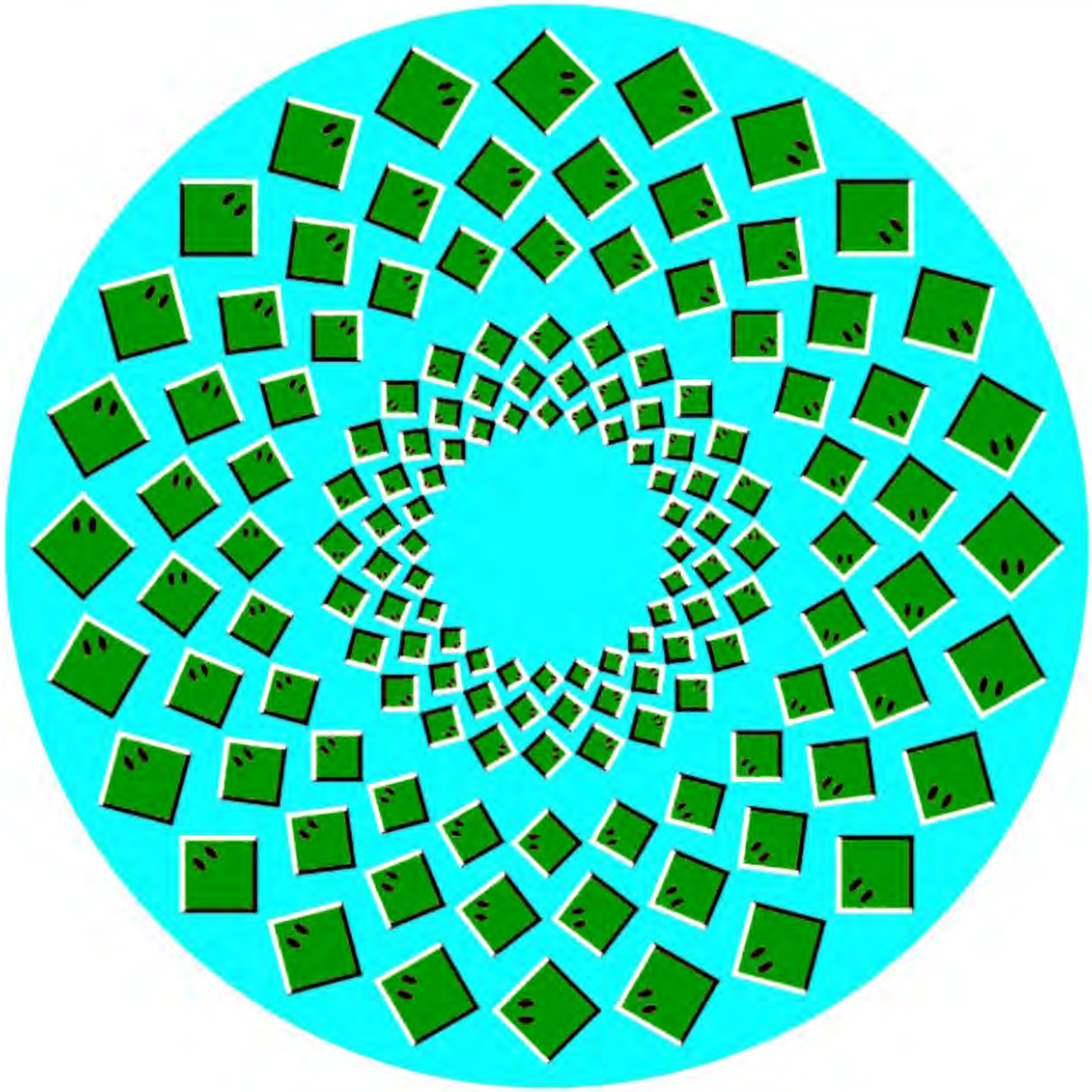


Image Processor

BackGround

50 120 50

0 200 0

Enabled

ForeGround

180 180 180

0 150 150

Enabled

Expander1

Apply

Reload and Apply

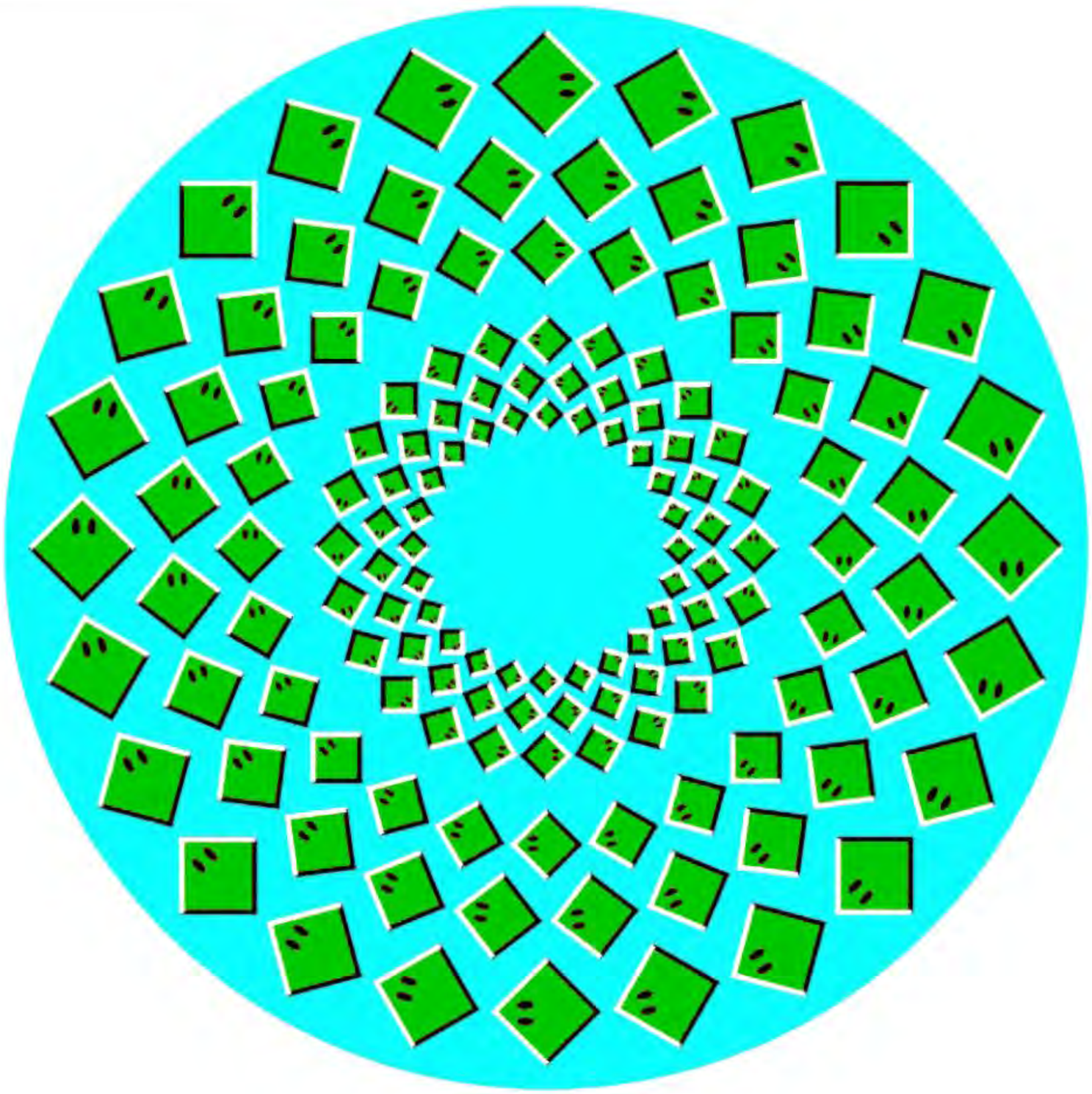
0°

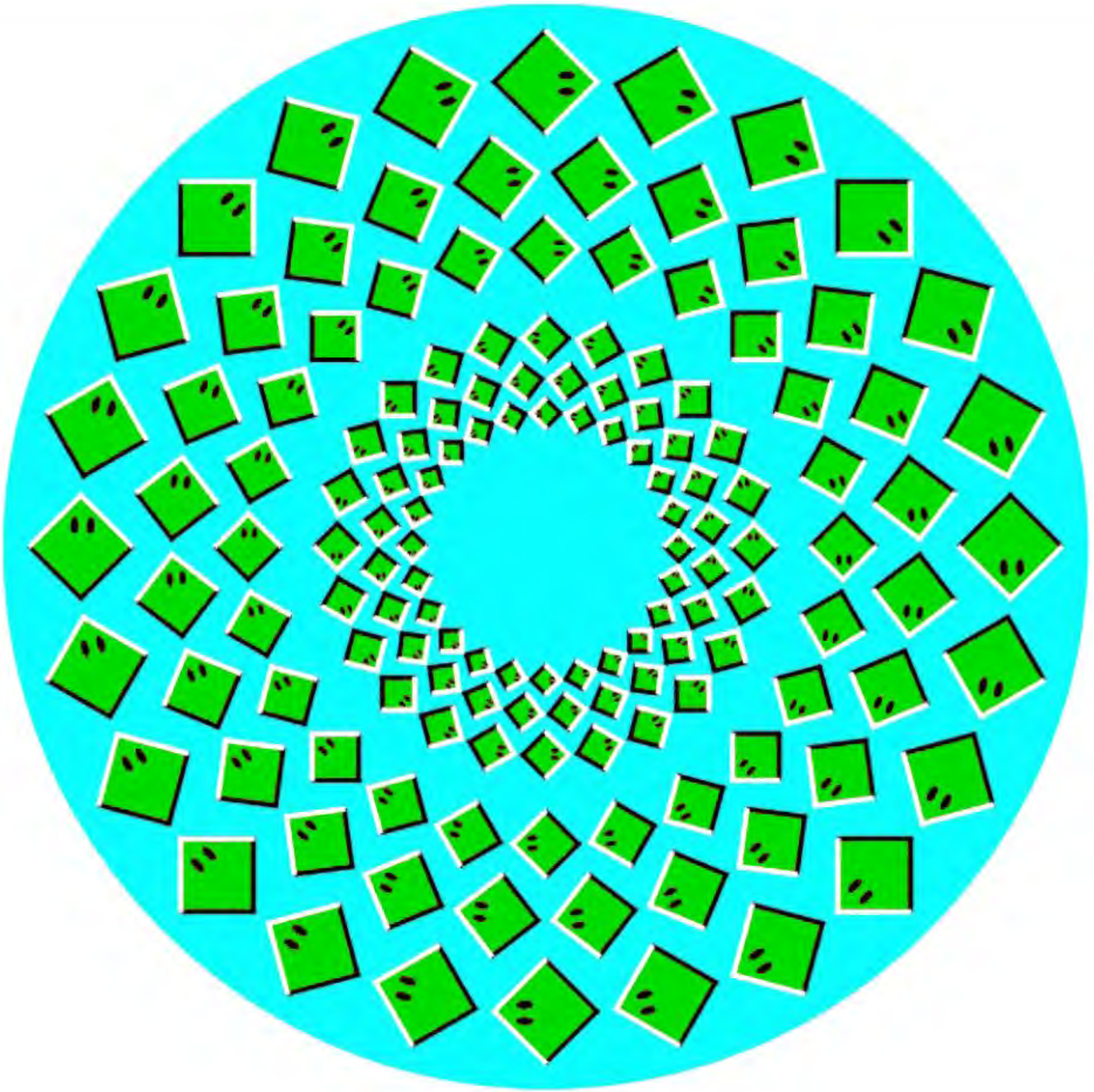
Rotate CW Rotate CCW

Save Image

Load Image

Reload Image





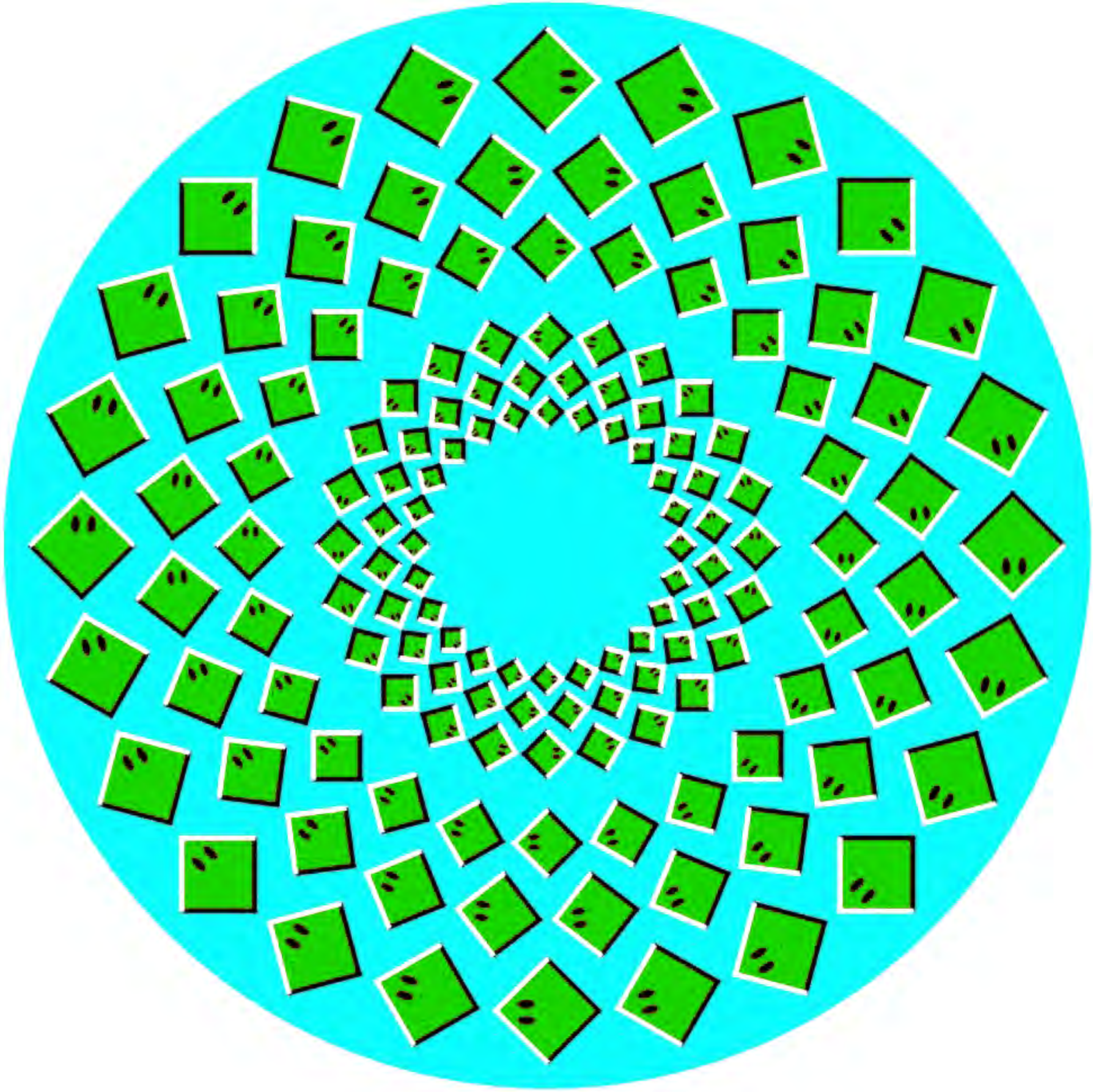


Image Processor

BackGround

50 | 120 | 50

0 | 90 | 0

Enabled

ForeGround

180 | 180 | 180

0 | 90 | 90

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

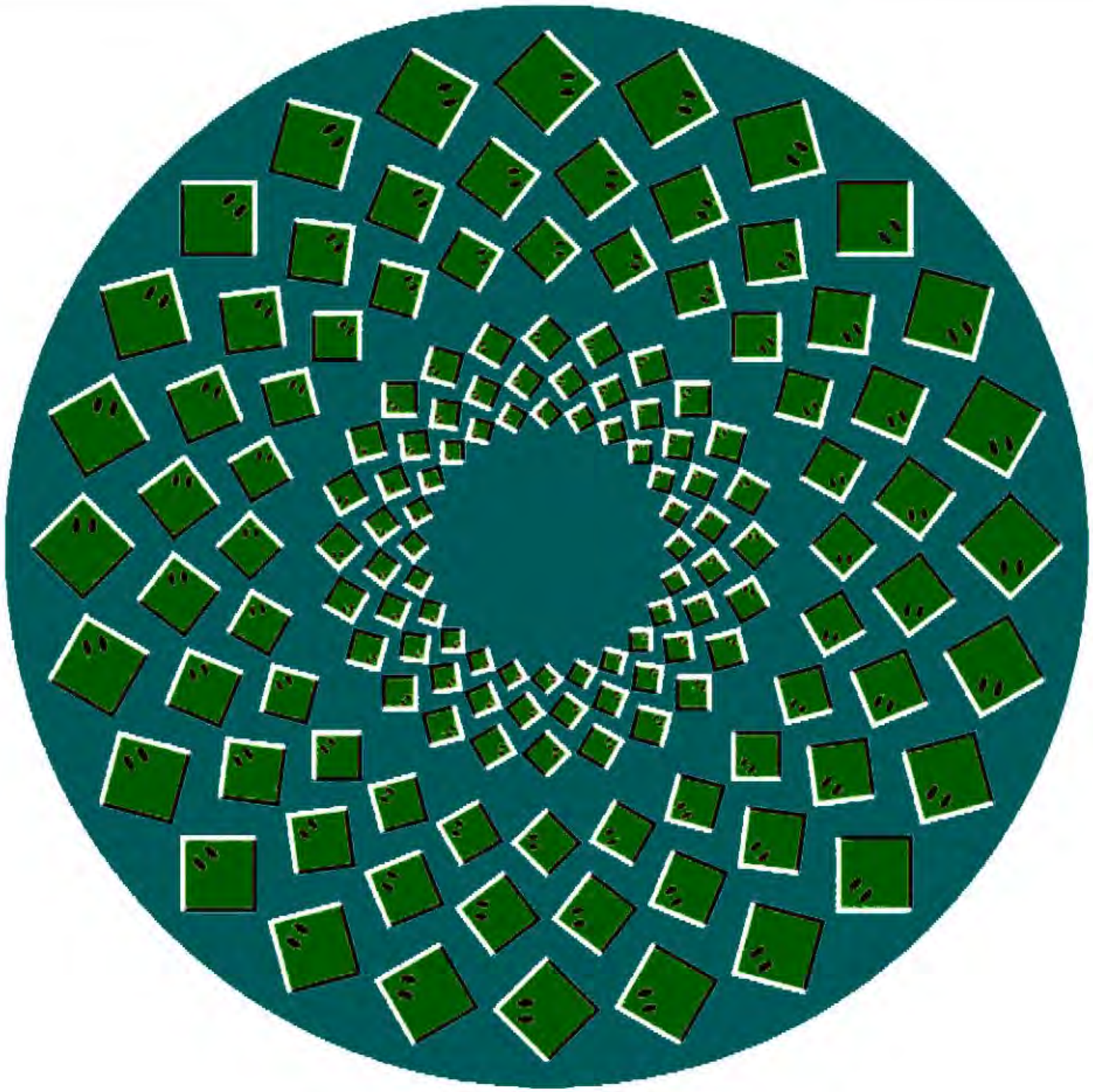


Image Processor

BackGround

50 120 50

0 80 0

Enabled

ForeGround

180 180 180

80 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

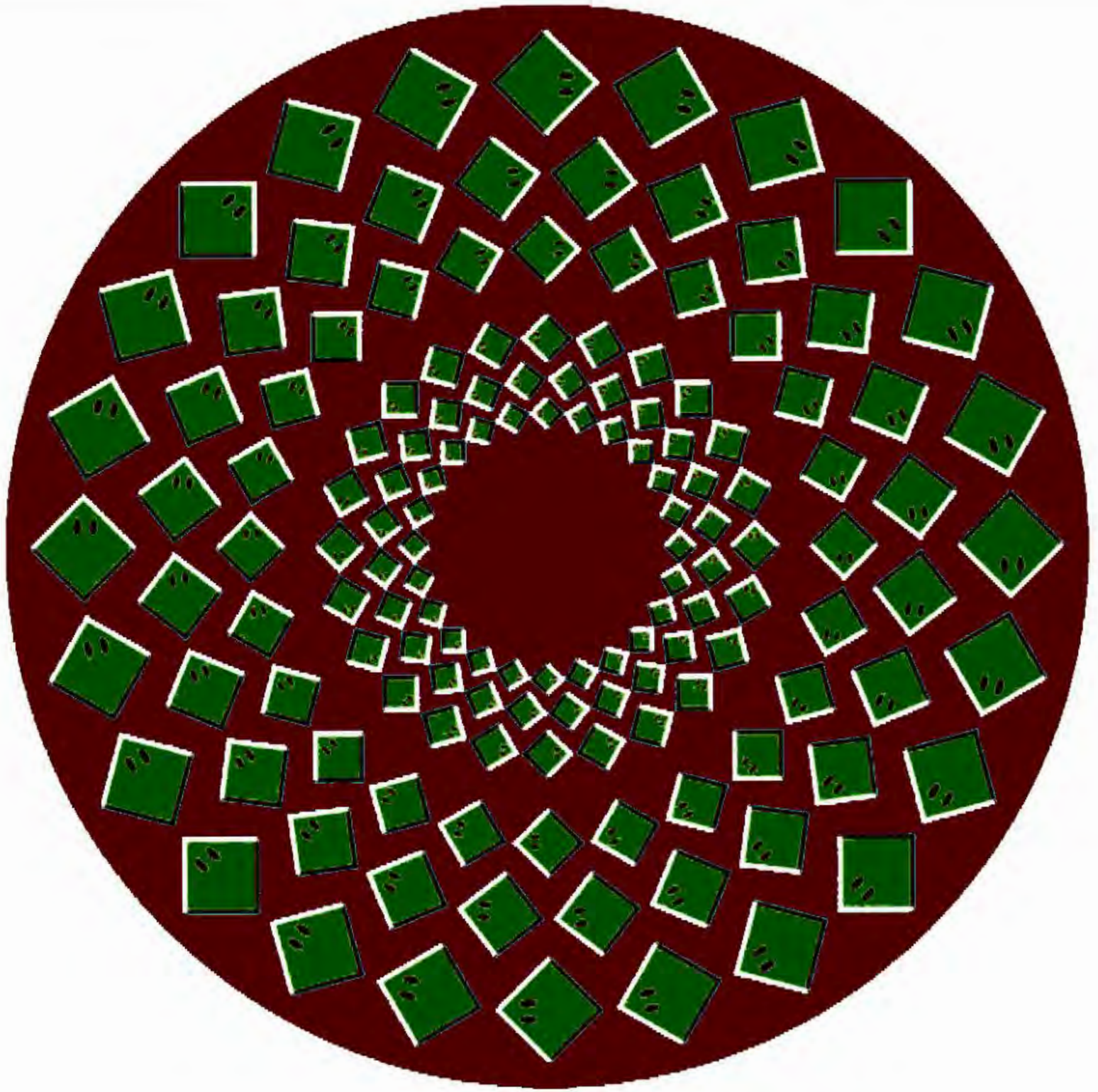


Image Processor

BackGround

50 | 120 | 50

0 | 100 | 0

Enabled

ForeGround

180 | 180 | 180

0 | 255 | 255

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

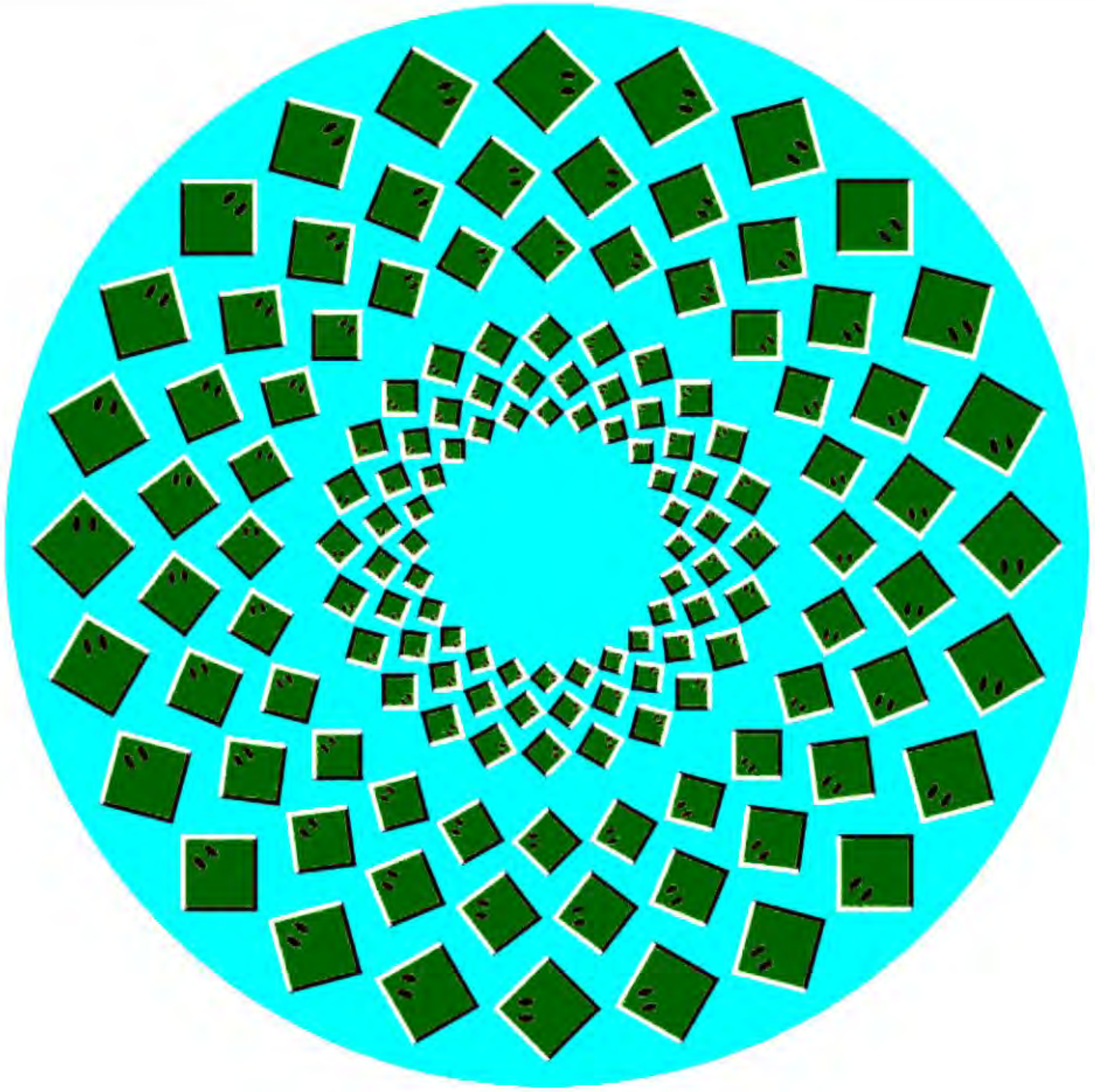


Image Processor

BackGround

50 120 50

0 150 0

Enabled

ForeGround

180 180 180

220 220 220

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

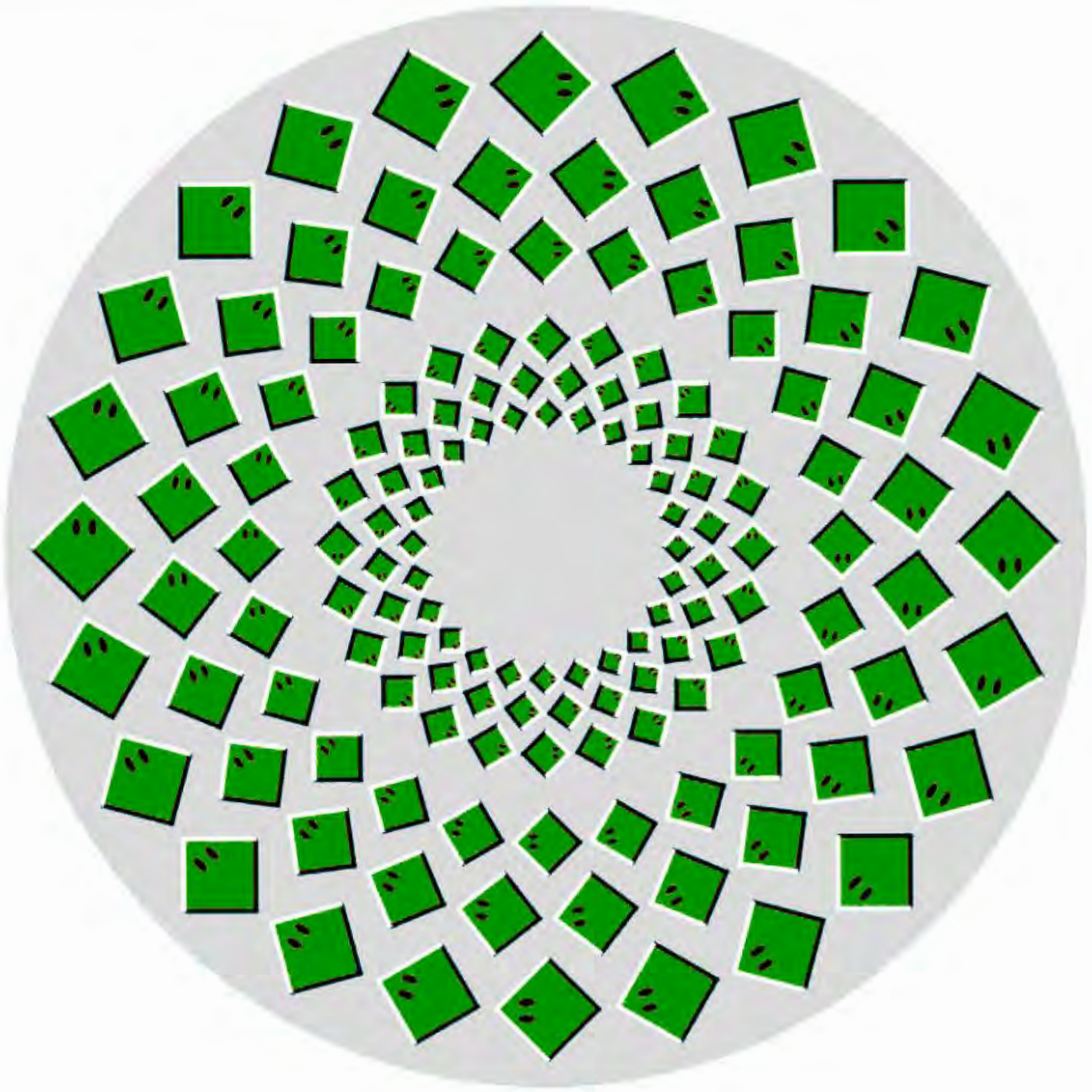


Image Processor

BackGround

50 120 50

0 100 0

Enabled

ForeGround

180 180 180

220 0 0

Enabled

Expander1

Apply

Reload and Apply

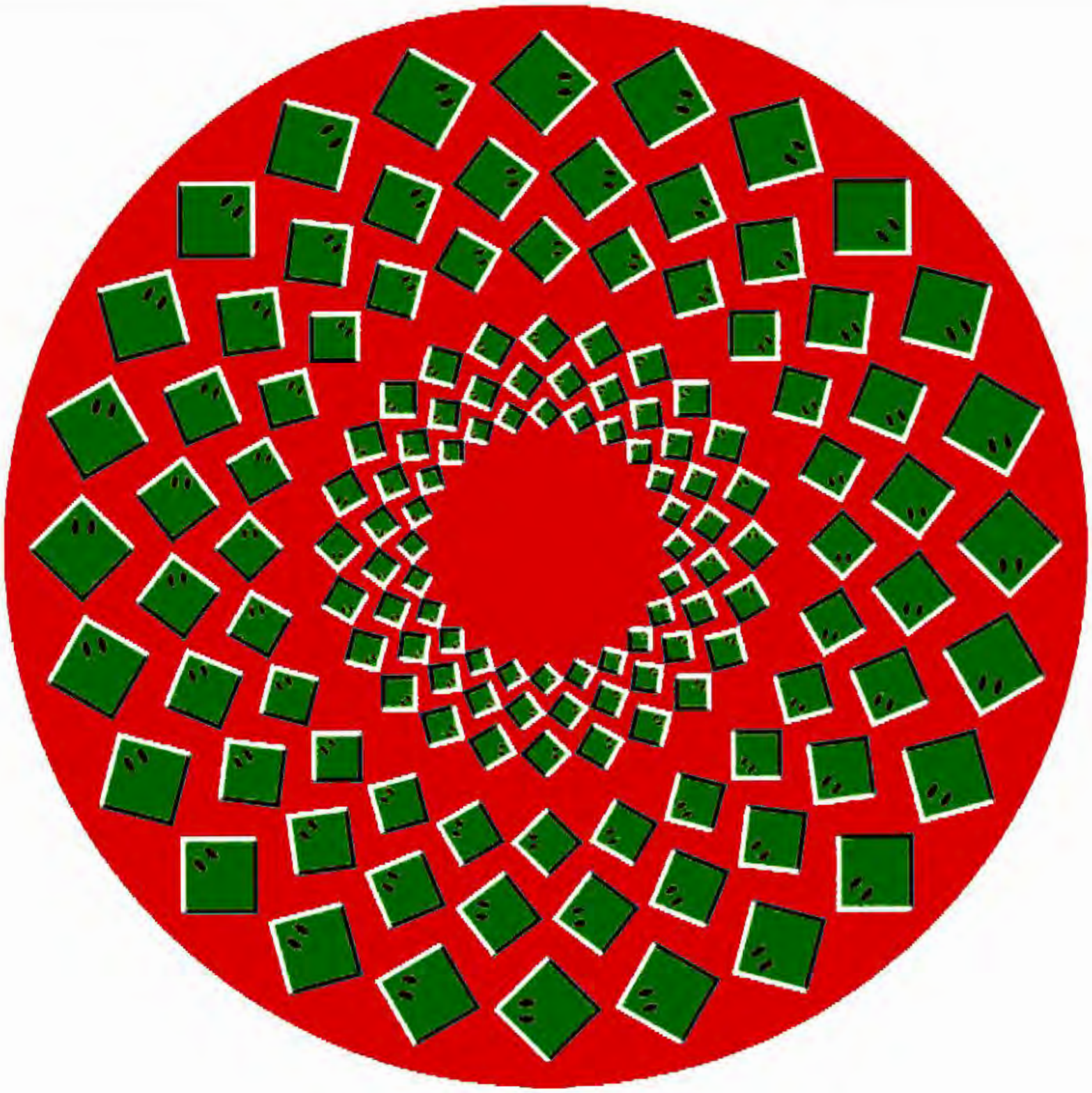
Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image



```
unit DataBaseFunctions;

interface
Uses Forms;
Procedure CreatePTable(NameDB: ShortString);
Procedure CreatePSummaryTable;
Procedure CreateCalTable(NameDB: ShortString);
Procedure CreateSumTableParvo;
Procedure CreateTableParticipants;
Procedure CreateMSummaryTable;
Procedure CreateMTable(NameDB: ShortString);

implementation
Uses CPT,FileFunctions,DB;

Procedure CreateTableParticipants;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`participants` (';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`ID` varchar(9) NOT NULL,';
  SString:=SString+'`Gender` varchar(1) NOT NULL,';
  SString:=SString+'`Age` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Visual Rating` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Colour Blind` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Date` datetime NOT NULL,';
  SString:=SString+'`V1` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V2` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V3` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V4` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V5` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V6` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V7` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V8` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V9` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`First Language` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form1.ADOCommand1.CommandText:=SString;
  Form1.ADOCommand1.Execute;
  Application.ProcessMessages;
End;

Procedure CreateSumTableParvo;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`ParvoCalibration` (';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`ID` varchar(9),';
  SString:=SString+'`Solution 1Hz` int(16) DEFAULT NULL,';
  SString:=SString+'`Solution 2Hz` int(16) DEFAULT NULL,';
```

```
SString:=SString+'`Solution 3Hz` int(16) DEFAULT NULL,';
SString:=SString+'`Solution 4Hz` int(16) DEFAULT NULL,';
SString:=SString+'`Solution 5Hz` int(16) DEFAULT NULL,';
SString:=SString+'`S1 Hz1 F` int(16) DEFAULT NULL,';
SString:=SString+'`S1 Hz2 F` int(16) DEFAULT NULL,';
SString:=SString+'`S1 Hz3 F` int(16) DEFAULT NULL,';
SString:=SString+'`S1 Hz4 F` int(16) DEFAULT NULL,';
SString:=SString+'`S1 Hz5 F` int(16) DEFAULT NULL,';
SString:=SString+'`S2 Hz1 F` int(16) DEFAULT NULL,';
SString:=SString+'`S2 Hz2 F` int(16) DEFAULT NULL,';
SString:=SString+'`S2 Hz3 F` int(16) DEFAULT NULL,';
SString:=SString+'`S2 Hz4 F` int(16) DEFAULT NULL,';
SString:=SString+'`S2 Hz5 F` int(16) DEFAULT NULL,';
SString:=SString+'`S3 Hz1 F` int(16) DEFAULT NULL,';
SString:=SString+'`S3 Hz2 F` int(16) DEFAULT NULL,';
SString:=SString+'`S3 Hz3 F` int(16) DEFAULT NULL,';
SString:=SString+'`S3 Hz4 F` int(16) DEFAULT NULL,';
SString:=SString+'`S3 Hz5 F` int(16) DEFAULT NULL,';
SString:=SString+'`S4 Hz1 F` int(16) DEFAULT NULL,';
SString:=SString+'`S4 Hz2 F` int(16) DEFAULT NULL,';
SString:=SString+'`S4 Hz3 F` int(16) DEFAULT NULL,';
SString:=SString+'`S4 Hz4 F` int(16) DEFAULT NULL,';
SString:=SString+'`S4 Hz5 F` int(16) DEFAULT NULL,';
SString:=SString+'`S5 Hz1 F` int(16) DEFAULT NULL,';
SString:=SString+'`S5 Hz2 F` int(16) DEFAULT NULL,';
SString:=SString+'`S5 Hz3 F` int(16) DEFAULT NULL,';
SString:=SString+'`S5 Hz4 F` int(16) DEFAULT NULL,';
SString:=SString+'`S5 Hz5 F` int(16) DEFAULT NULL,';
SString:=SString+'`S1 High` int(16) DEFAULT NULL,';
SString:=SString+'`S1 Avg` int(16) DEFAULT NULL,';
SString:=SString+'`S2 High` int(16) DEFAULT NULL,';
SString:=SString+'`S2 Avg` int(16) DEFAULT NULL,';
SString:=SString+'`S3 High` int(16) DEFAULT NULL,';
SString:=SString+'`S3 Avg` int(16) DEFAULT NULL,';
SString:=SString+'`S4 High` int(16) DEFAULT NULL,';
SString:=SString+'`S4 Avg` int(16) DEFAULT NULL,';
SString:=SString+'`S5 High` int(16) DEFAULT NULL,';
SString:=SString+'`S5 Avg` int(16) DEFAULT NULL,';
SString:=SString+'`Final Solution` int(16) DEFAULT NULL,';
SString:=SString+'`BG R` int(16) DEFAULT NULL,';
SString:=SString+'`BG G` int(16) DEFAULT NULL,';
SString:=SString+'`BG B` int(16) DEFAULT NULL,';
SString:=SString+'`FG R` int(16) DEFAULT NULL,';
SString:=SString+'`FG G` int(16) DEFAULT NULL,';
SString:=SString+'`FG B` int(16) DEFAULT NULL,';
SString:=SString+'`User Calibration` TINYINT(1) unsigned DEFAULT NULL,';
SString:=SString+'`Cal1R` int(10) unsigned DEFAULT NULL,';
SString:=SString+'`Cal1G` int(10) unsigned DEFAULT NULL,';
SString:=SString+'`Cal1B` int(10) unsigned DEFAULT NULL,';
SString:=SString+'`Cal2R` int(10) unsigned DEFAULT NULL,';
SString:=SString+'`Cal2G` int(10) unsigned DEFAULT NULL,';
SString:=SString+'`Cal2B` int(10) unsigned DEFAULT NULL,';
SString:=SString+'`Prescribed Solution` TINYINT(1) unsigned DEFAULT NULL,';
SString:=SString+'`PC1R` int(10) unsigned DEFAULT NULL,';
```

```
  SString:=SString+'`PC1G` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`PC1B` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`PC2R` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`PC2G` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`PC2B` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form1.ADOCommand1.CommandText:=SString;
  Form1.ADOCommand1.Execute;
  Application.ProcessMessages;
End;

Procedure CreateCalTable(NameDB: ShortString);
Var SString : WideString;
    NString : String;
Begin
  NString:=NameDB+'` ';
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`ParvoCal'+NString+'(';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`Timing` int(16) DEFAULT NULL,';
  SString:=SString+'`Colour Solution` int(16) DEFAULT NULL,';
  SString:=SString+'`C1 Red` int(16) DEFAULT NULL,';
  SString:=SString+'`C1 Green` int(16) DEFAULT NULL,';
  SString:=SString+'`C1 Blue` int(16) DEFAULT NULL,';
  SString:=SString+'`C2 Red` int(16) DEFAULT NULL,';
  SString:=SString+'`C2 Green` int(16) DEFAULT NULL,';
  SString:=SString+'`C2 Blue` int(16) DEFAULT NULL,';
  SString:=SString+'`Key Pressed` tinyInt(1) DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form1.ADOCommand1.CommandText:=SString;
  Form1.ADOCommand1.Execute;
  Application.ProcessMessages;
End;

Procedure CreatePTable(NameDB: ShortString);
Var SString : WideString;
    NString : String;
Begin
  NString:=NameDB+'` ';
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`Lum'+NString+'(';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`ID` varchar(9),';
  SString:=SString+'`TimeT` varchar(10) DEFAULT NULL,';
  SString:=SString+'`Stimulus Type` varchar(10) DEFAULT NULL,';
  SString:=SString+'`Stimulus Item` varchar(10)DEFAULT NULL,';
  SString:=SString+'`Response Latency` int(16) DEFAULT NULL,';
  SString:=SString+'`Error Latency` int(16) DEFAULT NULL,';
  SString:=SString+'`Non Target` varchar(10) DEFAULT NULL,';
  SString:=SString+'`Non Target Item` varchar(10) DEFAULT NULL,';
  SString:=SString+'`TimeE` varchar(10) DEFAULT NULL,';
  SString:=SString+'`Value` int(16) DEFAULT NULL,';
```

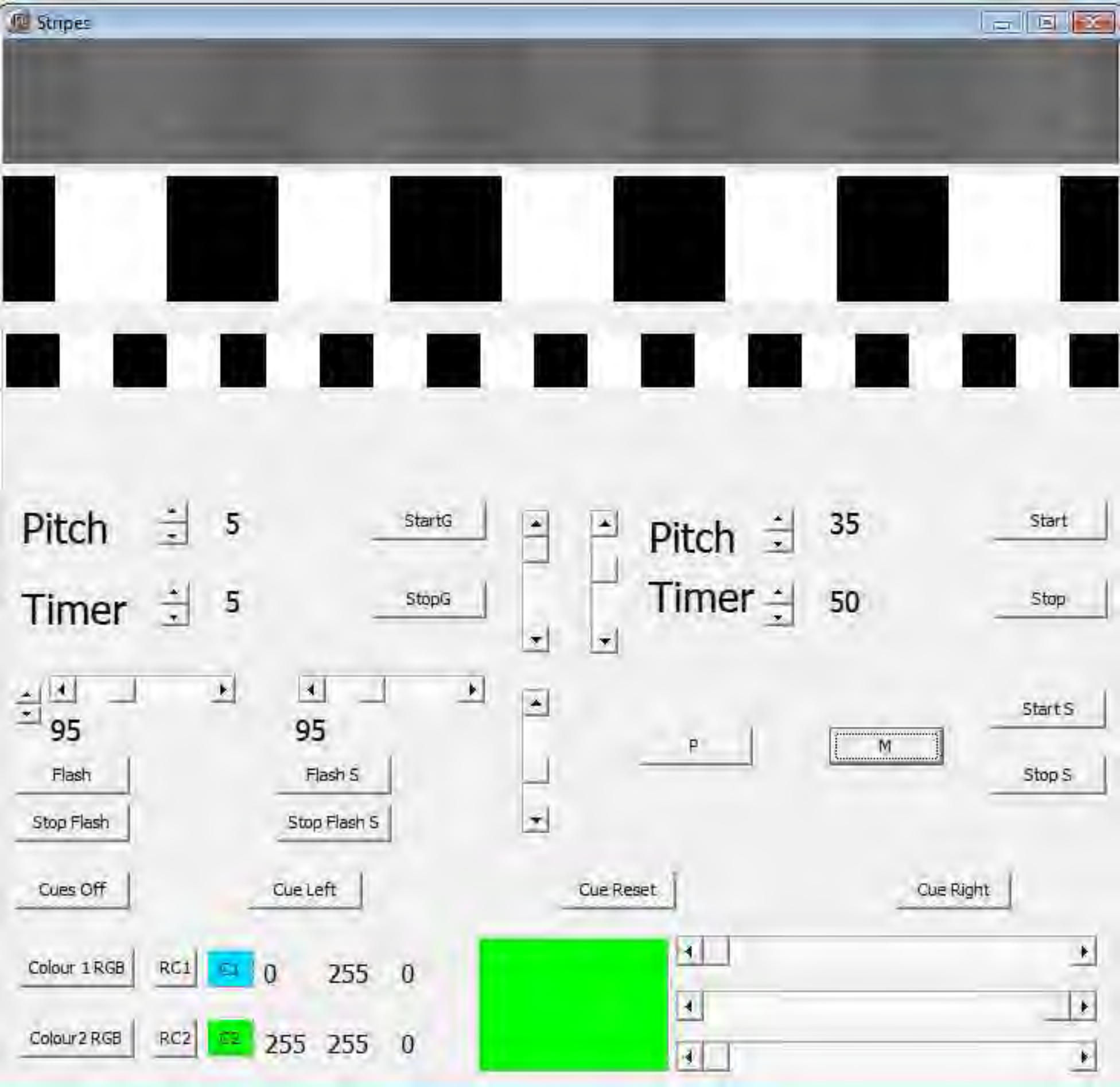
```
  SString:=SString+'`Report Seen` varchar(15) DEFAULT NULL,';
  SString:=SString+'`Key Pressed` int(16) DEFAULT NULL,';
  SString:=SString+'`R` int(16) DEFAULT NULL,';
  SString:=SString+'`G` int(16) DEFAULT NULL,';
  SString:=SString+'`B` int(16) DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form1.ADOCommand1.CommandText:=SString;
  Form1.ADOCommand1.Execute;
  Application.ProcessMessages;
End;

Procedure CreatePSummaryTable;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`LuminCalibration` (';
  SString:=SString+'`ID` varchar(9),';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`Score` int(16) DEFAULT NULL,';
  SString:=SString+'`Total Targets` int(16) DEFAULT NULL,';
  SString:=SString+'`Non Targets` int(16) DEFAULT NULL,';
  SString:=SString+'`Errors` int(16) DEFAULT NULL,';
  SString:=SString+'`Mean Target Latency` int(16) DEFAULT NULL,';
  SString:=SString+'`Mean Error Latency` int(16) DEFAULT NULL,';
  SString:=SString+'`STD Targets` float DEFAULT NULL,';
  SString:=SString+'`STD Errors` float DEFAULT NULL,';
  SString:=SString+'`Timer1` int(16) DEFAULT NULL,';
  SString:=SString+'`Timer2` int(16) DEFAULT NULL,';
  SString:=SString+'`Timer3` int(16) DEFAULT NULL,';
  SString:=SString+'`Hits` int(16) DEFAULT NULL,';
  SString:=SString+'`Targets` int(16) DEFAULT NULL,';
  SString:=SString+'`False Alarms` int(16) DEFAULT NULL,';
//  SString:=SString+'`Non Targets` int(16) DEFAULT NULL,';
  SString:=SString+'`Hit Rate Z` float DEFAULT NULL,';
  SString:=SString+'`False Alarm Z` float DEFAULT NULL,';
  SString:=SString+'`Hit Rate` float DEFAULT NULL,';
  SString:=SString+'`False Alarm Rate` float DEFAULT NULL,';
  SString:=SString+'`D Prime` float DEFAULT NULL,';
  SString:=SString+'`Adjustment` float DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form1.ADOCommand1.CommandText:=SString;
  Form1.ADOCommand1.Execute;
  Application.ProcessMessages;
End;

Procedure CreateMSummaryTable;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`MainStages` (';
  SString:=SString+'`ID` varchar(9),';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
```

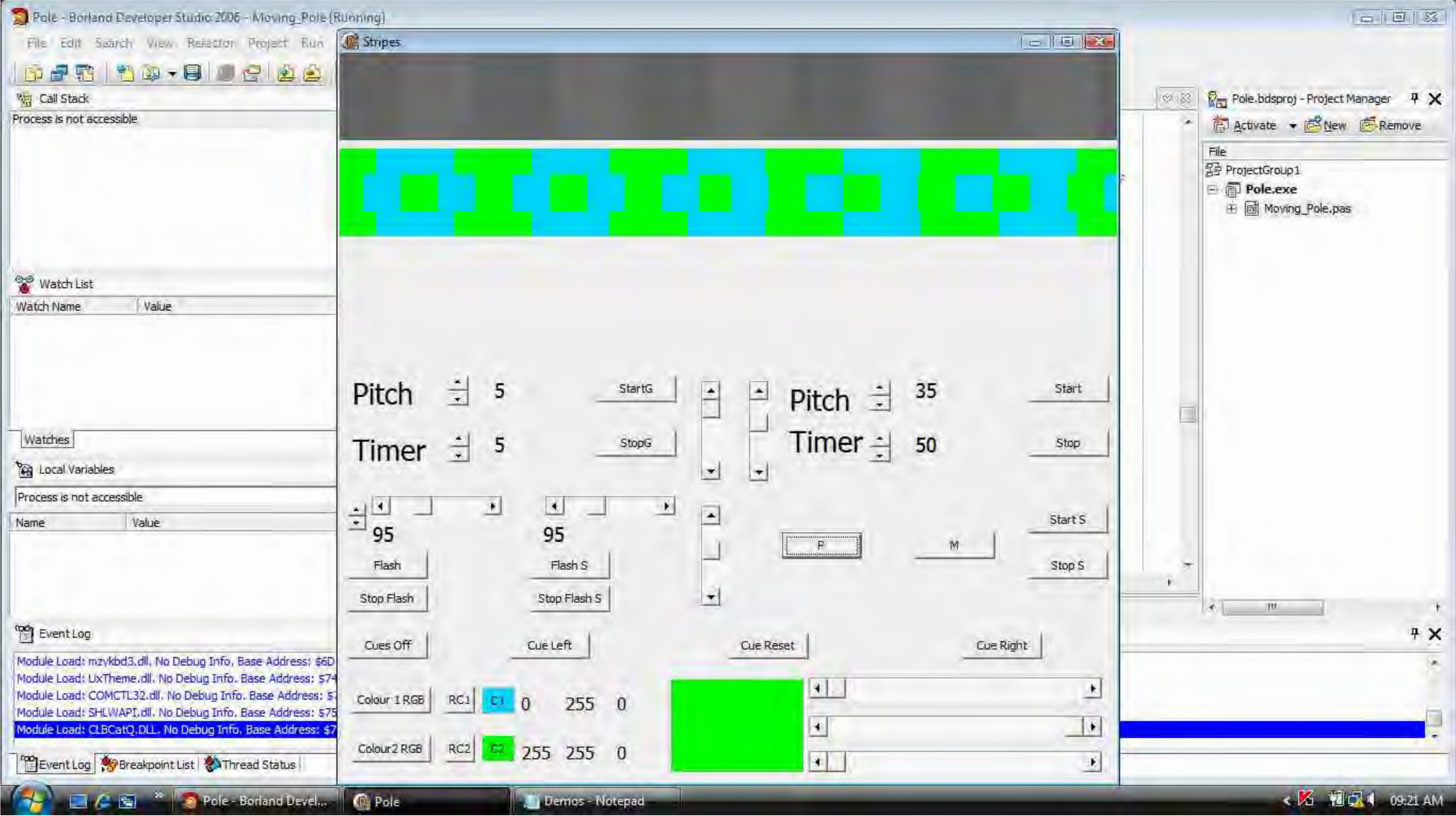
```
  SString:=SString+'`Stage` int(16) DEFAULT NULL,';
  SString:=SString+'`Score M` int(16) DEFAULT NULL,';    //MTotM
  SString:=SString+'`Total Targets` int(16) DEFAULT NULL,'; //MTargetsM
  SString:=SString+'`Total M NonTargets` int(16) DEFAULT NULL,';  //MNonTargetsM
  SString:=SString+'`Total M Targets` int(16) DEFAULT NULL,'; //MScoreM
  SString:=SString+'`Total M Errors` int(16) DEFAULT NULL,'; //MErrorsM
  SString:=SString+'`Total Null Errors M` int(16) DEFAULT NULL,'; //NullErrorsM
  SString:=SString+'`Mean Target Latency M` int(16) DEFAULT NULL,'; //ELatencyM
  SString:=SString+'`Mean Error Latency M` int(16) DEFAULT NULL,'; //TLatencyM
  SString:=SString+'`STD Targets` float DEFAULT NULL,';
  SString:=SString+'`STD Errors` float DEFAULT NULL,';
  SString:=SString+'`Score P` int(16) DEFAULT NULL,';    //MTotM
  SString:=SString+'`Total P Targets` int(16) DEFAULT NULL,'; //MTargetsM
  SString:=SString+'`Total P NonTargets` int(16) DEFAULT NULL,';  //MNonTargetsM
  SString:=SString+'`Total P Errors` int(16) DEFAULT NULL,'; //MErrorsM
  SString:=SString+'`Total Null Errors P` int(16) DEFAULT NULL,'; //NullErrorsM
  SString:=SString+'`Mean Target Latency P` int(16) DEFAULT NULL,'; //ELatencyM
  SString:=SString+'`Mean Error Latency P` int(16) DEFAULT NULL,'; //TLatencyM
//  SString:=SString+'`STD Targets P` float DEFAULT NULL,';
//  SString:=SString+'`STD Errors P` float DEFAULT NULL,';
  SString:=SString+'`Timer1` int(16) DEFAULT NULL,';
  SString:=SString+'`Timer2` int(16) DEFAULT NULL,';
  SString:=SString+'`Timer3` int(16) DEFAULT NULL,';
  SString:=SString+'`M Hits` int(16) DEFAULT NULL,';
  SString:=SString+'`Targets M` int(16) DEFAULT NULL,';
  SString:=SString+'`False Alarms M` int(16) DEFAULT NULL,';
  SString:=SString+'`M Hit Rate Z` float DEFAULT NULL,';
  SString:=SString+'`M False Alarm Z` float DEFAULT NULL,';
  SString:=SString+'`M Hit Rate` float DEFAULT NULL,';
  SString:=SString+'`M False Alarm Rate` float DEFAULT NULL,';
  SString:=SString+'`D Prime M` float DEFAULT NULL,';
  SString:=SString+'`P Hits` int(16) DEFAULT NULL,';
  SString:=SString+'`Targets P` int(16) DEFAULT NULL,';
  SString:=SString+'`False Alarms P` int(16) DEFAULT NULL,';
  SString:=SString+'`P Hit Rate Z` float DEFAULT NULL,';
  SString:=SString+'`P False Alarm Z` float DEFAULT NULL,';
  SString:=SString+'`P Hit Rate` float DEFAULT NULL,';
  SString:=SString+'`P False Alarm Rate` float DEFAULT NULL,';
  SString:=SString+'`D Prime P` float DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form1.ADOCommand1.CommandText:=SString;
  Form1.ADOCommand1.Execute;
  Application.ProcessMessages;
 End;

 Procedure CreateMTable(NameDB: ShortString);
 Var SString : WideString;
    NString : String;
 Begin
  NString:=NameDB+'` ';
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`Main'+NString+'(';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
```

```
  SString:=SString+'`Stage` int(16) DEFAULT NULL,';
  SString:=SString+'`ID` varchar(9),';
  SString:=SString+'`TimeT` varchar(10) DEFAULT NULL,';
  SString:=SString+'`Stimulus Type` varchar(10) DEFAULT NULL,';  //FirstWordT
  SString:=SString+'`Stimulus Item` varchar(10)DEFAULT NULL,';   //SecondWordT
  SString:=SString+'`Response Latency` int(16) DEFAULT NULL,';   //MLatency
  SString:=SString+'`Target Response Iteration  Latency` int(16) DEFAULT NULL,';   //M IT Latency
  SString:=SString+'`Error Latency` int(16) DEFAULT NULL,';      //MElatency
  SString:=SString+'`Error Response Iteration Latency` int(16) DEFAULT NULL,';     //M Error Latency
  SString:=SString+'`Non Target` varchar(10) DEFAULT NULL,';     //FirstWordF
  SString:=SString+'`Non Target Item` varchar(10) DEFAULT NULL,'; //SecondWordF
  SString:=SString+'`TimeE` varchar(10) DEFAULT NULL,';          //TimeE
  SString:=SString+'`Value` int(16) DEFAULT NULL,';              //Value
  SString:=SString+'`Report Seen` varchar(15) DEFAULT NULL,';     //ReportSeen
  SString:=SString+'`Target type` int(16) DEFAULT NULL,';        //TargetType
  SString:=SString+'`Key Pressed` int(16) DEFAULT NULL,';        //KeyPressed
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form1.ADOCommand1.CommandText:=SString;
  Form1.ADOCommand1.Execute;
  Application.ProcessMessages;
End;

end.
```

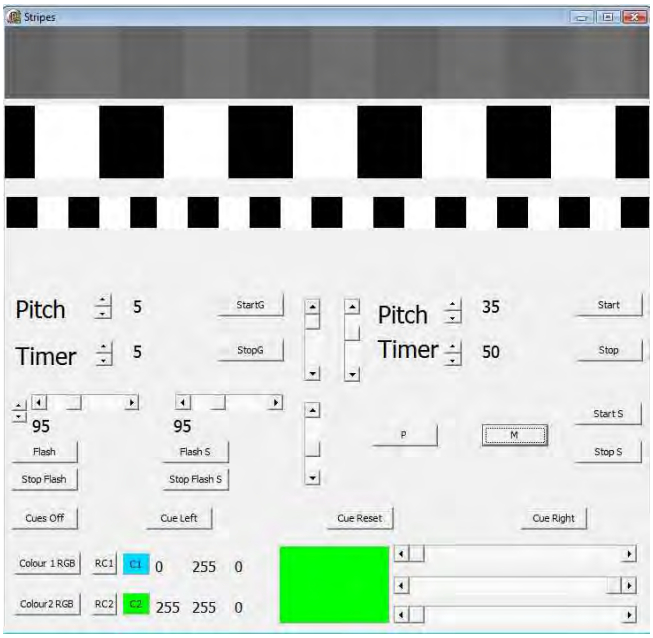
```
unit DBFunctions;

interface
Uses Forms;
Procedure CreateIndTable(NameDB: ShortString);
Procedure CreateSummaryTable;

implementation
Uses ArrayFunctions, FileFunctions, IATWinFormUnit, SysUtils,StrUtils;

Procedure CreateIndTable(NameDB: ShortString);
Var SString : WideString;
   NString : String;
Begin
  NString:=NameDB+'` ';
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`Ind'+NString+'(';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`ID` varchar(9),';
  SString:=SString+'`Block` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Date` varchar(11) DEFAULT NULL,';
  SString:=SString+'`Time` varchar(11) DEFAULT NULL,';
  SString:=SString+'`Gender` varchar(1) DEFAULT NULL,';
  SString:=SString+'`Age` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`QResponse` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Colour` varchar(1) DEFAULT NULL,';
  SString:=SString+'`SubTimer1` varchar(4) DEFAULT NULL,';
  SString:=SString+'`SubTimer2` varchar(4) DEFAULT NULL,';
  SString:=SString+'`Counter` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Diff` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Item` varchar(15) DEFAULT NULL,';
  SString:=SString+'`SubItem` varchar(25) DEFAULT NULL,';
  SString:=SString+'`Flag` TINYINT(1) unsigned DEFAULT NULL,';
  SString:=SString+'`BlockOrder` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  TWinForm1.ADOCommand1.CommandText:=SString;
  TWinForm1.ADOCommand1.Execute;
  Application.ProcessMessages;
End;

Procedure CreateSummaryTable;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`Summary` (';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`ID` varchar(9),';
  SString:=SString+'`Date` datetime NOT NULL,';
  SString:=SString+'`Age` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Gender` varchar(1) DEFAULT NULL,';
  SString:=SString+'`QResponse` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Colour` varchar(1) DEFAULT NULL,';
```

```
  SString:=SString+'`SubTimer1` varchar(4) DEFAULT NULL,';
  SString:=SString+'`SubTimer2` varchar(4) DEFAULT NULL,';
  SString:=SString+'`Block1 Average` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Block1 SD` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Block2 Average` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Block2 SD` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Block3 Average` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Block3 SD` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Block4 Average` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Block4 SD` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Block5 Average` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Block5 SD` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Block2 4 D` FLOAT DEFAULT NULL,';
  SString:=SString+'`D Score` FLOAT DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  TWinForm1.ADOCommand1.CommandText:=SString;
  TWinForm1.ADOCommand1.Execute;
  Application.ProcessMessages;
End;

end.
```

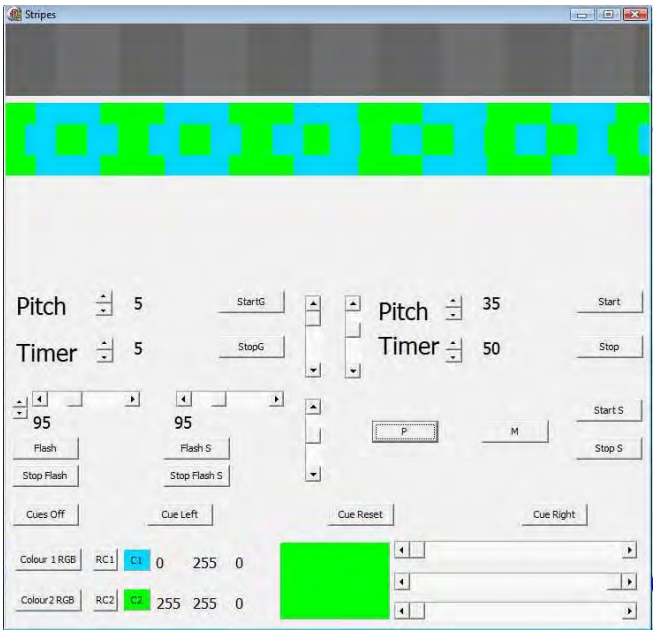

Stripes
Pitch
5
StartG
Timer
5
StopG
Pitch
35
Start
Timer
50
Stop
95
95
Flash
Stop Flash
Flash S
Stop Flash S
P
M
Start S
Stop S
Cues Off
Cue Left
Cue Reset
Cue Right
Colour 1 RGB
RC1
0
255
0
Colour2 RGB
RC2
255
255
0

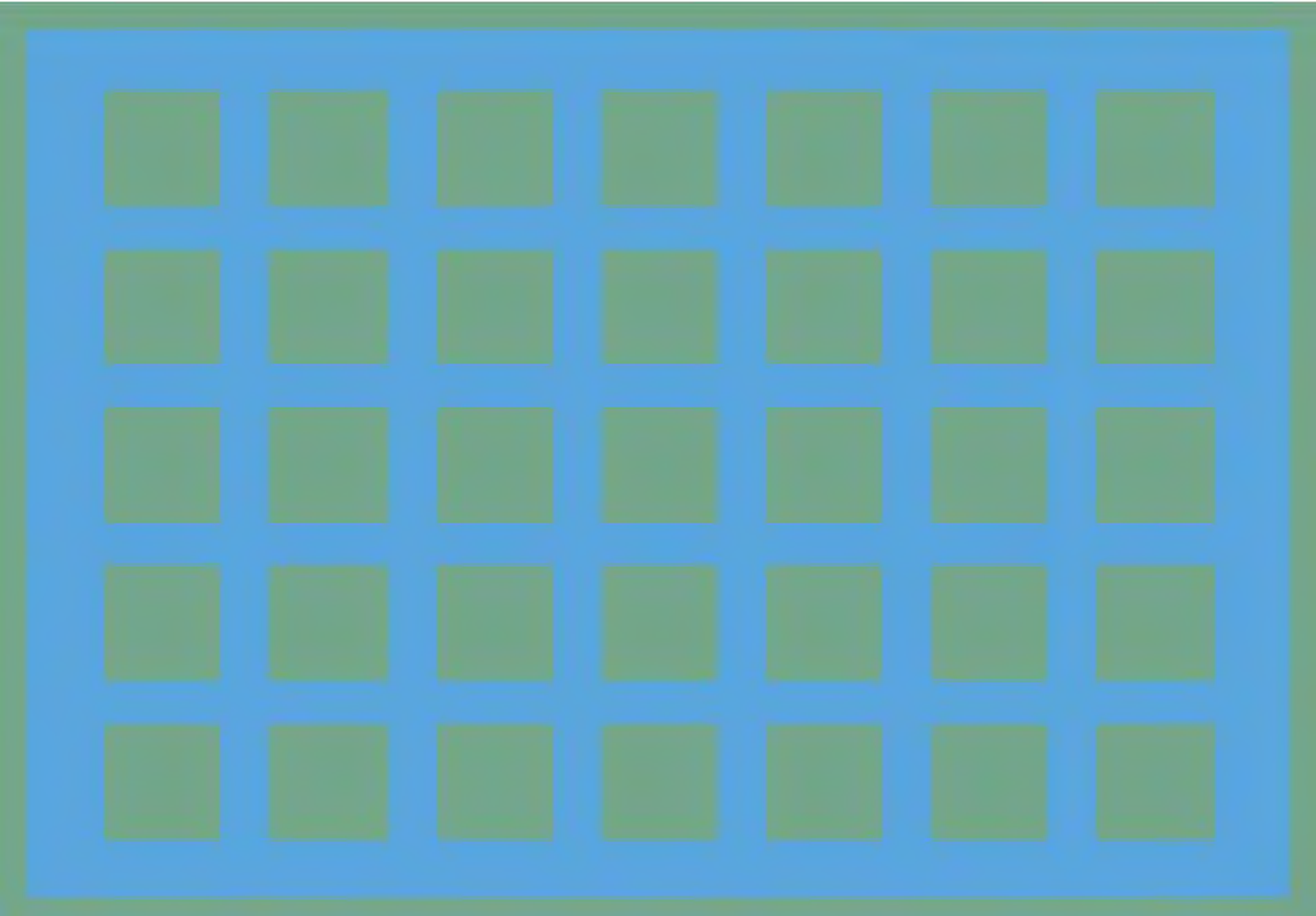

Pole - Borland Developer Studio 2006 - Moving_Pole [Running]
File Edit Search View Refactor Project Run
Call Stack
Process is not accessible
Watch List
Watch Name
Value
Watches
Local Variables
Process is not accessible
Name
Value
Event Log
Module Load: mzykbd3.dll. No Debug Info. Base Address: $6D
Module Load: UxTheme.dll. No Debug Info. Base Address: $74
Module Load: COMCTL32.dll. No Debug Info. Base Address: $7
Module Load: SHLWAPI.dll. No Debug Info. Base Address: $75
Module Load: CLBCatQ.DLL. No Debug Info. Base Address: $7
Event Log
Breakpoint List
Thread Status
Stripes
Pitch
5
StartG
Timer
5
StopG
95
Flash
Stop Flash
95
Flash S
Stop Flash S
Cues Off
Cue Left
Cue Reset
Cue Right
Colour 1 RGB
RC1
C1
0
255
0
Colour2 RGB
RC2
C2
255
255
0
Pitch
35
Start
Timer
50
Stop
P
M
Start S
Stop S
Pole.bdsproj - Project Manager
Activate
New
Remove
File
ProjectGroup1
Pole.exe
Moving_Pole.pas
Pole - Borland Devel...
Pole
Demos - Notepad
09:21 AM

Demo 1


Stripes
Pitch
5
StartG
Timer
5
StopG
95
95
Flash
Flash S
Stop Flash
Stop Flash S
Pitch
35
Start
Timer
50
Stop
P
M
Start S
Stop S
Cues Off
Cue Left
Cue Reset
Cue Right
Colour 1 RGB
RC1
C1
0
255
0
Colour 2 RGB
RC2
C2
255
255
0

Demo 2


Stripes
Pitch
5
StartG
Timer
5
StopG
95
95
Flash
Flash S
Stop Flash
Stop Flash S
Pitch
35
Start
Timer
50
Stop
P
M
Start S
Stop S
Cues Off
Cue Left
Cue Reset
Cue Right
Colour 1 RGB
RC1
C1
0
255
0
Colour 2 RGB
RC2
C2
255
255
0



```
unit FileFunctions;

interface
Uses Cpt,Sysutils,Sorting,Math,Classes,StrUtils;
Procedure InitScores;
Procedure OpenPracticeFile(VAR Finished : Boolean; Namecode: String);
Procedure OpenMainFile(VAR Finished: Boolean; Namecode: String);
Procedure OpenMainFile2(VAR Finished: Boolean; Namecode: String);
Procedure CloseM;
Procedure CloseP;
Procedure EndPracticeCM(Var PracticeD: Double; Var Adjustment : Real);
Procedure EndMain(Var MainDM,MainDP : Double; Stage: Integer);
//Procedure EndMain(Var Main1DM,MainDP : Double; Stage: Integer);
Procedure OpenInPutFiles(Var LTargets,LNon,LPracticeTargets,LPracNonT: Integer);
Procedure Checklist1;
Procedure Checklist2;
Procedure ChecklistP;
Procedure GetFileNames;
Procedure OpenItemMain(Namecode: String; Stage: Integer);
Procedure OpenSettings(Var T1,T2,T3,TurnAroundV,MaxRuns,CIterations : Integer; Var MaskCh,Colour1,Colour2:
String;
Var PEnabled,MEnabled,Cal: Boolean; Var Ban2 : String; Var RColour: Boolean);
Procedure DPrime(Var HR,FA,Hz,Fz,MainD : Double);
Procedure ExtractFileNames;
Procedure OpenMainFiles(Var LTargets,LNon: Integer; Phase: String);
procedure ExtractPFiles;


Type
  List = Record
  Item : String[10];
  Index : Integer;
End;

var
  ListArray: Array of List;

implementation
Uses DataBaseFunctions;
Procedure NormZ(Var P,PPP : Double); External
'NORMSINV.dll';

procedure ExtractPFiles;
Var
  ReturnCode :  Cardinal;
  Path       :  STRING;
  FileSpec   :  STRING;
  Line,Str   :  STRING;
  SearchRec  :  TSearchRec;
  IDX,NullS,NullL : Integer;
  FileList : TStringList;
  Smaller : TStringList;
  Bigger : TStringList;
```

```
begin
  FileList:=TStringList.Create;
  Smaller:=TStringList.Create;
  Bigger:=TStringList.Create;
FileSpec := '*.bmp';
Path:='C:\CPT\Files\ShoeBox\LumPics\';
ReturnCode := SysUtils.FindFirst(Path + Filespec, faAnyFile, SearchRec);
  WHILE ReturnCode = 0 DO
    Begin
      Line := SearchRec.Name;
      FileList.Add(Line);
      ReturnCode := SysUtils.FindNext(SearchRec);
    End;
   for IDX := 0 to FileList.Count - 1 do
  Begin
    Line:=FileList[IDX];
    Str:=MidStr(Line,1,1);
    if Str='S' then
    Begin
      Line:=AnsiReplaceText(Line,'.bmp','');
      Smaller.Add(Line);
    End;
    if Str='L' then
      Begin
      Line:=AnsiReplaceText(Line,'.bmp','');
      Bigger.Add(Line);
      End;
  End;
  NullS:=9;
  NullL:=11;
  if NullS>0 then
  Begin
    for IDX := 1 to NullS do
      Begin
        Smaller.Add('blank');
      End;
  End;
  if NullL>0 then
  Begin
    for IDX := 1 to NullL do
      Begin
        Bigger.Add('blank');
      End;
  End;
  Smaller.SaveToFile('C:\CPT\StFiles\SmallerCM.txt');
  Bigger.SaveToFile('C:\CPT\StFiles\BiggerCM.txt');
  FileList.SaveToFile('C:\CPT\StFiles\AllCM.txt');
  Smaller.Free;
  Bigger.Free;
  FileList.Clear;
  FileList.Free;
end;
```

```
Procedure OpenMainFiles(Var LTargets,LNon: Integer; Phase: String);
Var Counts : Integer;
Temp,FileNameML,FileNameMS : String;
Begin
  FileNameML:='C:\CPT\StFiles\Large'+Phase+'.txt';
  FileNameMS:='C:\CPT\StFiles\Small'+Phase+'.txt';
  Assign(InfileM,FileNameML);  //'c:\cpt\StFiles\Large.txt');
  Reset(InfileM);
  Counts:=0;
    While NOT Eof(InfileM) DO
      Begin
        Readln(InfileM,Temp);
        Counts:=Counts+1;
      End;
    LTargets:=Counts;
    SetLength(PracTargets,LTargets+1);
    Reset(InfileM);
    For Counts:= 1 to LTargets DO
      Begin
        Readln(InfileM,PracTargets[Counts].TargetStr);
        PracTargets[Counts].TargetValue:=0;
      End;
    Reset(InfileM);
  Assign(InfileM,FileNameMS); //'c:\cpt\StFiles\Small.txt');
  Reset(InfileM);
  Counts:=0;
    While NOT Eof(InfileM) DO
      Begin
        Counts:=Counts+1;
        Readln(InfileM,Temp);
      End;
    LNon:=Counts;
    SetLength(PracNonTargets,LNon+1);
    Reset(InfileM);
  For Counts:=1 to LNon DO
  Begin
    Readln(InfileM,PracNonTargets[Counts].NonTargetString);
    PracNonTargets[Counts].NonTargetValue:=1;
  End;
  Close(InfileM);
End;

Procedure SortArray(Len : Integer);
var
  I, J, T: Integer;
  TS : String[10];
Begin
  for I := 0 To Len-1 Do
    for J := Len-1 downto I+1 do
      if ListArray[I].Index > ListArray[J].Index then
      begin
        T:=ListArray[I].Index;
        TS:=ListArray[I].Item;
        ListArray[I].Item:=ListArray[J].Item;
```

```
        ListArray[I].Index:=ListArray[J].Index;
        ListArray[J].Item:=TS;
        ListArray[J].Index:=T;
      end;
End;

Procedure RandomOrderList(Var FileList: TStringList);
Var
  RR : Integer;
  I : Integer;
  Len : Integer;
  Itm : String[20];
Begin
  Len:=FileList.Count;
  SetLength(ListArray,Len);
  for I := 0 to FileList.Count - 1 do
  Begin
   RR:=Random(200000);
   Itm:=FileList[I];
   Itm:=AnsiReplaceText(Itm,'.bmp','');
   ListArray[I].Item:=Itm;
   ListArray[I].Index:=RR;
  End;
  SortArray(Len);
  FileList.Clear;
  for I:=0 To Len-1 do
    Begin
      FileList.Add(ListArray[I].Item);
    End;
End;

procedure ExtractFileNames;
Var
  ReturnCode :  Cardinal;
  Path       :  STRING;
  FileSpec   :  STRING;
  Line,Str   :  STRING;
  SearchRec  :  TSearchRec;
  IDX,NullS,NullL,SmallerLen,BiggerLen,BiggerIDX,SmallerIDX : Integer;
  FileList : TStringList;
  Smaller,Smaller1,Smaller2,Smaller3,Smaller4 : TStringList;
  Bigger,Bigger1,Bigger2,Bigger3,Bigger4 : TStringList;
begin
  FileList:=TStringList.Create;
FileSpec := '*.bmp';
Path:='C:\CPT\Files\ShoeBox\All Pictures\';
ReturnCode := SysUtils.FindFirst(Path + Filespec, faAnyFile, SearchRec);
  WHILE ReturnCode = 0 DO
    Begin
      Line := SearchRec.Name;
      FileList.Add(Line);
      ReturnCode := SysUtils.FindNext(SearchRec);
    End;
  FileList.SaveToFile('C:\CPT\StFiles\BMPListALL.txt');
```

```
RandomOrderList(FileList);
Smaller:=TStringList.Create;
Bigger:=TStringList.Create;
for IDX := 0 to FileList.Count - 1 do
Begin
  Line:=FileList[IDX];
  Str:=MidStr(Line,1,1);
  if Str='S' then
    Smaller.Add(Line);
  if Str='L' then
    Bigger.Add(Line);
End;
SmallerLen:=Smaller.Count Div 4;
BiggerLen:=Bigger.Count Div 4;
SmallerIDX:=SmallerLen;
BiggerIDX:=BiggerLen;
NullS:=SmallerLen Div 2;
NullL:=BiggerLen Div 2;

Bigger1:=TStringList.Create;
for IDX:=0 to BiggerIDX- 1 do
  Begin
    Bigger1.Add(Bigger[IDX]);
  End;
  if NullL>0 then
Begin
  for IDX := 1 to NullL do
    Begin
      Bigger1.Add('Null');
    End;
End;
Bigger1.SaveToFile('C:\CPT\StFiles\Large1.txt');
Bigger1.Free;

Bigger2:=TStringList.Create;
for IDX:= BiggerIDX to (BiggerIDX+BiggerLen)-1 do
  Begin
    Bigger2.Add(Bigger[IDX]);
  End;
  if NullL>0 then
Begin
  for IDX := 1 to NullL do
    Begin
      Bigger2.Add('Null');
    End;
End;
Bigger2.SaveToFile('C:\CPT\StFiles\Large2.txt');
Bigger2.Free;

BiggerIDX:=BiggerIDX+BiggerLen;
Bigger3:=TStringList.Create;
for IDX:= BiggerIDX to (BiggerIDX+BiggerLen)-1 do
  Begin
    Bigger3.Add(Bigger[IDX]);
```

```
  End;
  if NullL>0 then
Begin
  for IDX := 1 to NullL do
    Begin
      Bigger3.Add('Null');
    End;
End;
Bigger3.SaveToFile('C:\CPT\StFiles\Large3.txt');
Bigger3.Free;

BiggerIDX:=BiggerIDX+BiggerLen;
Bigger4:=TStringList.Create;
for IDX:= BiggerIDX to (BiggerIDX+BiggerLen)-1 do
  Begin
    Bigger4.Add(Bigger[IDX]);
  End;
  if NullL>0 then
Begin
  for IDX := 1 to NullL do
    Begin
      Bigger4.Add('Null');
    End;
End;
Bigger4.SaveToFile('C:\CPT\StFiles\Large4.txt');
Bigger4.Free;

Smaller1:=TStringList.Create;
for IDX:=0 to SmallerLen- 1 do
  Begin
    Smaller1.Add(Smaller[IDX]);
  End;
  if NullS>0 then
Begin
  for IDX := 1 to NullS do
    Begin
      Smaller1.Add('Null');
    End;
End;
Smaller1.SaveToFile('C:\CPT\StFiles\Small1.txt');
Smaller1.Free;

Smaller2:=TStringList.Create;
for IDX:= SmallerIDX to (SmallerIDX+SmallerLen)-1 do
  Begin
    Smaller2.Add(Smaller[IDX]);
  End;
  if NullS>0 then
Begin
  for IDX := 1 to NullS do
    Begin
      Smaller2.Add('Null');
    End;
End;
```

```
Smaller2.SaveToFile('C:\CPT\StFiles\Small2.txt');
Smaller2.Free;

SmallerIDX:=SmallerIDX+SmallerLen;
Smaller3:=TStringList.Create;
for IDX:= SmallerIDX to (SmallerIDX+SmallerLen)-1 do
  Begin
    Smaller3.Add(Smaller[IDX]);
  End;
  if NullS>0 then
Begin
  for IDX := 1 to NullS do
    Begin
      Smaller3.Add('Null');
    End;
End;
Smaller3.SaveToFile('C:\CPT\StFiles\Small3.txt');
Smaller3.Free;

SmallerIDX:=SmallerIDX+SmallerLen;
Smaller4:=TStringList.Create;
for IDX:= SmallerIDX to (SmallerIDX+SmallerLen)-1 do
  Begin
    Smaller4.Add(Smaller[IDX]);
  End;
  if NullS>0 then
Begin
  for IDX := 1 to NullS do
    Begin
      Smaller4.Add('Null');
    End;
End;
Smaller4.SaveToFile('C:\CPT\StFiles\Small4.txt');
Smaller4.Free;

NullS:=Smaller.Count Div 2;
NullL:=Bigger.Count Div 2;
if NullS>0 then
Begin
  for IDX := 1 to NullS do
    Begin
      Smaller.Add('Null');
    End;
End;
if NullL>0 then
Begin
  for IDX := 1 to NullL do
    Begin
      Bigger.Add('Null');
    End;
End;
Smaller.SaveToFile('C:\CPT\StFiles\Small.txt');
Bigger.SaveToFile('C:\CPT\StFiles\Large.txt');
Smaller.Free;
```

```
  Bigger.Free;
  FileList.Clear;
  FileList.Free;
end;

Procedure GetFileNames;
Var InputFile : Text; TempS : String;
Begin
  Assign(InputFile,'C:\CPT\StFiles\StartFiles.txt');
  Reset(InputFile);
  Readln(InputFile,TempS);
  if TempS='short' then
    Begin
      FLarge:='C:\CPT\StFiles\Large1.txt';
      FSmall:='C:\CPT\StFiles\Small1.txt';
      FSmallerCM:='C:\CPT\StFiles\SmallerCMShort.txt';
      FBiggerCM:='C:\CPT\StFiles\BiggerCMShort.txt';
    End
      Else
        Begin
          FLarge:='C:\CPT\StFiles\Large1.txt';
          FSmall:='C:\CPT\StFiles\Small1.txt';
          FSmallerCM:='C:\CPT\StFiles\SmallerCM.txt';
          FBiggerCM:='C:\CPT\StFiles\BiggerCM.txt';
        End;
End;

Procedure OpenSettings(Var T1,T2,T3,TurnAroundV,MaxRuns,CIterations : Integer; Var MaskCh,Colour1,Colour2 :
String;
Var PEnabled,MEnabled,Cal :Boolean; Var Ban2: String; Var RColour: Boolean);
Var InputFile : Text; TempS : String;
Begin
   Assign(InputFile,'c:\CPT\StFiles\SettingsUCT.txt');
   Reset(InputFile);
   Readln(InputFile,TempS); // timer 1 interval
   T1:=StrToInt(TempS);
   Readln(InputFile,TempS); // timer 2 interval
   T2:=StrToInt(TempS);
   Readln(InputFile,TempS); // timer 3 interval
   T3:=StrToInt(TempS);
   Readln(InputFile,MaskCh); // the mask characters
   if MaskCh='enabled-' Then Masking:=False;
   Readln(InputFile,Colour1);  // background colour 1
   Readln(InputFile,Colour2);  //background colour 2
   Readln(InputFile,TempS);
   SizerInt:=StrToInt(TempS);
   Overide:=False;
   Readln(InputFile,TempS);
   If TempS='overide' then Overide:=True;
   Readln(InputFile,TempS);
   If TempS='random' Then RColour:=True
     Else RColour:=False;
   REadln(InputFile,TempS);    // is the traffic light enabled or not
   If TempS='enabled' then PEnabled:=true
```

```
Else PEnabled:=False;
Readln(InputFile,TempS);   // is the traffic light enabled or not
If TempS='enabled' Then MEnabled:=True
Else MEnabled:=False;
Readln(InputFile,TempS);
If TempS='calibrate' Then Cal:=True;
Readln(InputFile,Ban2);
Readln(InputFile,TempS);
Panel:=False;
Readln(InputFile,TempS);
if TempS='Panel' then Panel:=True;
ReadLn(InputFile,TempS);
CIterations:=StrToInt(TempS);
ReadLn(InputFile,TempS);
TurnAroundV:=StrToInt(TempS);
Readln(InputFile,TempS);
MaxRuns:=StrToInt(TempS);
Readln(InputFile,TempS);
PTiming[1]:=StrToInt(TempS);
Readln(InputFile,TempS);
PTiming[2]:=StrToInt(TempS);
Readln(InputFile,TempS);
PTiming[3]:=StrToInt(TempS);
Readln(InputFile,TempS);
PTiming[4]:=StrToInt(TempS);
Readln(InputFile,TempS);
PTiming[5]:=StrToInt(TempS);
// set threshold values here
Readln(InputFile,TempS);
  MSR:=StrToInt(TempS); //119;
Readln(InputFile,TempS);
  MSG:=StrToInt(TempS);//119;
Readln(InputFile,TempS);
  MSB:=StrToInt(TempS);//119;
Readln(InputFile,TempS);
  MDR:=StrToInt(TempS);//100;
Readln(InputFile,TempS);
  MDG:=StrToInt(TempS);//100;
Readln(InputFile,TempS);
  MDB:=StrToInt(TempS);//100;
Readln(InputFile,TempS);
  MFSR:=StrToInt(TempS);//122;
Readln(InputFile,TempS);
  MFSG:=StrToInt(TempS);//122;
Readln(InputFile,TempS);
  MFSB:=StrToInt(TempS);//122;
Readln(InputFile,TempS);
  MFDR:=StrToInt(TempS);//119;
Readln(InputFile,TempS);
  MFDG:=StrToInt(TempS);//119;
Readln(InputFile,TempS);
  MFDB:=StrToInt(TempS);//119;
Readln(InputFile,TempS);
  Equi:=StrToInt(TempS);//119;
```

```
   Close(InputFile);
End;

Procedure ChecklistP;
Var Idex : Integer;
PracticeOutFile : Text;
Begin
  Assign(PracticeOutFile,'c:\CPT\StFiles\ItemlistP.txt');
  Rewrite(PracticeOutFile);
  For Idex:=1 to ArrayLength DO
  Begin
    Writeln(PracticeOutFile,Pract[Idex].Word:12,' ',Pract[Idex].Value);
  End;
  Close(PracticeOutFile);
  Assign(PracticeOutFile,'c:\CPT\StFiles\BMPList.txt');
  Rewrite(PracticeOutFile);
  For Idex:=1 to ArrayLength DO
  Begin
    if Pract[Idex].Word<>'blank' then
      Writeln(PracticeOutFile,Pract[Idex].Word+'.bmp');
  End;
  Close(PracticeOutFile);
End;

Procedure Checklist1;
Var Idx1 : Integer;
M1Outfile : Text;
Begin
  Assign(M1Outfile,'c:\CPT\StFiles\itemlistA.txt');
  Rewrite(M1Outfile);
  For Idx1:=1 to MainT DO
  Begin
    Writeln(M1Outfile,Both[Idx1].Word:12,' ',Both[Idx1].Value);
  End;
  Close(M1Outfile);
End;

Procedure Checklist2;
Var Idx2 : Integer;
M2OutFile : Text;
Begin
  Assign(M2OutFile,'c:\CPT\StFiles\itemlistB.txt');
  Rewrite(M2OutFile);
  For Idx2:=1 to MainT DO
  Begin
    Writeln(M2OutFile,Both[Idx2].Word:12,' ',Both[Idx2].Value);
  End;
  Close(M2OutFile);
End;

Procedure OpenItemMain(Namecode: String; Stage: Integer);
Var IdxM : Integer;
InPutFile: Text;
Filename : String;
```

```
begin
  if Stage=1 then
  Begin
    Filename:=Concat(Fileloc,Namecode,'Item1',Ext);
    Assign(InPutFile,Filename);
    Rewrite(InPutFile);
    Writeln(InPutFile,'Word list 1','  ',DateToStr(Date),'  ',TimeToStr(Time));
    Writeln(InPutFile,'ID: ',Namecode,' Gender: ',Gend,' Age: ',Age);
    Writeln(InPutFile);
  For IdxM:=1 to GlobalCounter-1 DO
   Begin
    Writeln(InPutFile,Both[IdxM].Word:12,' ',Both[IdxM].Value,'  ',Both[IdxM].TargetType);
   End;
   Close(InPutFile);
  End;
  if Stage=2 then
  Begin
    Filename:=Concat(Fileloc,Namecode,'Item2',Ext);
    Assign(InPutFile,Filename);
    Rewrite(InPutFile);
    Writeln(InPutFile,'Word list 2','  ',DateToStr(Date),'  ',TimeToStr(Time));
    Writeln(InPutFile,'ID: ',Namecode,' Gender: ',Gend,' Age: ',Age);
    Writeln(InPutFile);
  For IdxM:=1 to GlobalCounter-1 DO
   Begin
    Writeln(InPutFile,Both[IdxM].Word:12,' ',Both[IdxM].Value,'  ',Both[IdxM].TargetType);
   End;
   Close(InPutFile);
  End;
  if Stage=3 then
  Begin
    Filename:=Concat(Fileloc,Namecode,'Item3',Ext);
    Assign(InPutFile,Filename);
    Rewrite(InPutFile);
    Writeln(InPutFile,'Word list 3','  ',DateToStr(Date),'  ',TimeToStr(Time));
    Writeln(InPutFile,'ID: ',Namecode,' Gender: ',Gend,' Age: ',Age);
    Writeln(InPutFile);
  For IdxM:=1 to GlobalCounter-1 DO
   Begin
    Writeln(InPutFile,Both[IdxM].Word:12,' ',Both[IdxM].Value,'  ',Both[IdxM].TargetType);
   End;
   Close(InPutFile);
  End;
  if Stage=4 then
  Begin
    Filename:=Concat(Fileloc,Namecode,'Item4',Ext);
    Assign(InPutFile,Filename);
    Rewrite(InPutFile);
    Writeln(InPutFile,'Word list 4','  ',DateToStr(Date),'  ',TimeToStr(Time));
    Writeln(InPutFile,'ID: ',Namecode,' Gender: ',Gend,' Age: ',Age);
    Writeln(InPutFile);
  For IdxM:=1 to GlobalCounter-1 DO
   Begin
    Writeln(InPutFile,Both[IdxM].Word:12,' ',Both[IdxM].Value,'  ',Both[IdxM].TargetType);
```

```
    End;
    Close(InPutFile);
   End;
End;

Procedure DPrime(Var HR,FA,Hz,Fz,MainD : Double);
Var HitZ,FAZ: Double;
Begin
  IF HR<0.01 Then HR:=0.000001;
  IF HR>0.99 Then HR:=0.999999;
  IF FA<0.01 Then FA:=0.000001;
  IF FA>0.99 Then FA:=0.999999;
  NormZ(HR,HitZ);
  Hz:=HitZ;
  NormZ(FA,FAZ);
  Fz:=FAZ;
  MainD:=HitZ-FAZ;
End;

Procedure EndMain(Var MainDM,MainDP : Double; Stage: Integer);
Var Indx,IndxL,CountPlus,CountMinus : Integer;
    HRM,FAM,HzM,FzM,HRP,FAP,HzP,FzP: Double;
    TMain1,TMain2,TMain3,TMain4 : Array of Double;
    NameDB: ShortString;
    MeanM1,MeanME1,StdM1,StdE1,MeanM2,MeanME2,StdM2,StdE2,MeanM3,MeanME3,StdM3,StdE3,
    MeanM4,MeanME4,StdM4,StdE4 : Extended;
    MeanRPlus,MeanRMinus: Real;
BEGIN
    NameDB:=CPT.Namecode;
    StdM1:=0;
    StdE1:=0;
    StdM2:=0;
    StdE2:=0;
    StdM3:=0;
    StdE3:=0;
    StdM4:=0;
    StdE4:=0;
    if Stage=1 Then
    Begin
      M1Tot:=GlobalCounter-1;
      M1TargetsM:=MScores[Stage].MTargetsM;
      M1NonTargetsM:=MScores[Stage].MNonTargetsM;
      M1ScoreM:=MScores[Stage].MScoreM;
      If (M1ScoreM>0) AND (M1TargetsM>0) Then
        M1TLatencyM:=MScores[Stage].MTotM DIV M1ScoreM;
      M1ErrorsM:=MScores[Stage].MErrorsM;
       if (M1NonTargetsM>0) AND (M1ErrorsM>0) then
        M1ELatencyM:=MScores[Stage].ELatencyM DIV M1ErrorsM;
      M1NullErrors:=MScores[Stage].NullErrorsM+MScores[Stage].NullErrorsP;
      if (M1ScoreM>0) AND (M1TargetsM>0) then HRM:=M1ScoreM/M1TargetsM
        Else
          HRM:=0.001;
      if (M1ErrorsM>0) AND (M1NonTargetsM>0) then FAM:=M1ErrorsM/M1NonTargetsM
        Else
```

```
    FAM:=0.001;
  Dprime(HrM,FAM,HzM,FzM,MainDM);
  M1TargetsP:=MScores[Stage].MTargetsP;
  M1NonTargetsP:=MScores[Stage].MNonTargetsP;
  M1ScoreP:=MScores[Stage].MScoreP;
  If (M1ScoreP>0) AND (M1TargetsP>0) Then
    M1TLatencyP:=MScores[Stage].MTotP DIV M1ScoreP;
  M1ErrorsP:=MScores[Stage].MErrorsP;
  if (M1NonTargetsP>0) AND (M1ErrorsP>0) then
  M1ELatencyP:=MScores[Stage].ELatencyP DIV M1ErrorsP;
  if (M1ScoreP>0) AND (M1TargetsP>0) then HRP:=M1ScoreP/M1TargetsP
   Else
    HRP:=0.001;
  if (M1ErrorsP>0) AND (M1NonTargetsP>0) then FAP:=M1ErrorsP/M1NonTargetsP
   Else
    FAP:=0.001;
   Dprime(HrP,FAP,HzP,FzP,MainDP);

  if M1ScoreM+M1ScoreP>=2 then
    Begin
      Setlength(TMain1,M1ScoreM+M1ScoreP);
      IndxL:=0;
      for Indx:=1 to M1Tot do
        Begin
          if Scores1[Indx].ReportSeen='True Positive' then
          Begin
            TMain1[IndxL]:=Scores1[Indx].MLatency;
            Inc(IndxL);
          End;
        End;
        MeanAndStdDev(TMain1,MeanM1,StdM1);
    End
      Else
        Begin
          MeanM1:=0;
          StdM1:=0;
        End;
 if (M1ErrorsM+M1ErrorsP>=2) then
    Begin
      SetLength(TMain1,M1ErrorsM+M1ErrorsP);
      IndxL:=0;
      for Indx := 1 to M1Tot do
        Begin
          if Scores1[Indx].ReportSeen='False Positive' then
            Begin
              TMain1[IndxL]:=Scores1[Indx].MElatency;
              Inc(IndxL);
            End;
         End;
        MeanAndStdDev(TMain1,MeanME1,StdE1);
    End
      Else
        Begin
          MeanME1:=0;
```

```
        StdE1:=0;
       End;
  Form1.ADOSummaryMain.TableName:='MainStages';
  Form1.ADOSummaryMain.Open;
  Form1.ADOSummaryMain.Active:=true;
  With Form1.ADOSummaryMain Do
  Begin
    Append;
    Fields.FieldByName('ID').Value:=NameDB;
    Fields.FieldByName('Stage').Value:=Stage;
    Fields.FieldByName('Score M').Value:=M1ScoreM;
    Fields.FieldByName('Total Targets').Value:=M1Tot;
    Fields.FieldByName('Total M NonTargets').Value:=M1NonTargetsM;
    Fields.FieldByName('Total M Targets').Value:=M1TargetsM;
    Fields.FieldByName('Total M Errors').Value:= M1ErrorsM;
    Fields.FieldByName('Total Null Errors M').Value:=MScores[Stage].NullErrorsM;
    Fields.FieldByName('Mean Target Latency M').Value:=M1TLatencyM;   //MeanM1
    Fields.FieldByName('Mean Error Latency M').Value:=M1ELatencyM;
    Fields.FieldByName('STD Targets').Value:=StdM1;
    Fields.FieldByName('STD Errors').Value:=StdE1;
    Fields.FieldByName('Score P').Value:=M1ScoreP;
    Fields.FieldByName('Total P Targets').Value:=M1TargetsP;
    Fields.FieldByName('Total P NonTargets').Value:=M1NonTargetsP;
    Fields.FieldByName('Total P Errors').Value:=M1ErrorsP;
    Fields.FieldByName('Total Null Errors P').Value:=MScores[Stage].NullErrorsP;
    Fields.FieldByName('Mean Target Latency P').Value:=M1TLatencyP;
    Fields.FieldByName('Mean Error Latency P').Value:=M1ELatencyP;
    Fields.FieldByName('Timer1').Value:=T1;
    Fields.FieldByName('Timer2').Value:=T2;
    Fields.FieldByName('Timer3').Value:=T3;
    Fields.FieldByName('M Hits').Value:=M1ScoreM;
    Fields.FieldByName('Targets M').Value:=M1TargetsM;
    Fields.FieldByName('False Alarms M').Value:=M1ErrorsM;
    Fields.FieldByName('M Hit Rate Z').Value:=HzM;
    Fields.FieldByName('M False Alarm Z').Value:=FzM;
    Fields.FieldByName('M Hit Rate').Value:=HRM;
    Fields.FieldByName('M False Alarm Rate').Value:=FAM;
    Fields.FieldByName('D Prime M').Value:=MainDM;
    Fields.FieldByName('P Hits').Value:=M1ScoreP;
    Fields.FieldByName('Targets P').Value:=M1TargetsP;
    Fields.FieldByName('False Alarms P').Value:=M1ErrorsP;
    Fields.FieldByName('P Hit Rate Z').Value:=HzP;
    Fields.FieldByName('P False Alarm Z').Value:=FzP;
    Fields.FieldByName('P Hit Rate').Value:=HRP;
    Fields.FieldByName('P False Alarm Rate').Value:=FAP;
    Fields.FieldByName('D Prime P').Value:=MainDP;
    UpdateRecord;
    Post;
   End;
End;
if Stage=2 Then
Begin
  M2Tot:=GlobalCounter-1;
  M2TargetsM:=MScores[Stage].MTargetsM;
```

```
M2NonTargetsM:=MScores[Stage].MNonTargetsM;
M2ScoreM:=MScores[Stage].MScoreM;
If (M2ScoreM>0) AND (M2TargetsM>0) Then
   M2TLatencyM:=MScores[Stage].MTotM DIV M2ScoreM;
M2ErrorsM:=MScores[Stage].MErrorsM;
 if (M2NonTargetsM>0) AND (M2ErrorsM>0) then
   M2ELatencyM:=MScores[Stage].ELatencyM DIV M2ErrorsM;
M2NullErrors:=MScores[Stage].NullErrorsM+MScores[Stage].NullErrorsP;
if (M2ScoreM>0) AND (M2TargetsM>0) then HRM:=M2ScoreM/M2TargetsM
  Else
    HRM:=0.001;
if (M2ErrorsM>0) AND (M2NonTargetsM>0) then FAM:=M2ErrorsM/M2NonTargetsM
  Else
    FAM:=0.001;
 Dprime(HrM,FAM,HzM,FzM,MainDM);
 M2TargetsP:=MScores[Stage].MTargetsP;
 M2NonTargetsP:=MScores[Stage].MNonTargetsP;
 M2ScoreP:=MScores[Stage].MScoreP;
 If (M2ScoreP>0) AND (M2TargetsP>0) Then
   M2TLatencyP:=MScores[Stage].MTotP DIV M2ScoreP;
 M2ErrorsP:=MScores[Stage].MErrorsP;
 if (M2NonTargetsP>0) AND (M2ErrorsP>0) then
 M2ELatencyP:=MScores[Stage].ELatencyP DIV M2ErrorsP;
 if (M2ScoreP>0) AND (M2TargetsP>0) then HRP:=M2ScoreP/M1TargetsP
  Else
    HRP:=0.001;
 if (M2ErrorsP>0) AND (M2NonTargetsP>0) then FAP:=M2ErrorsP/M2NonTargetsP
  Else
    FAP:=0.001;
  Dprime(HrP,FAP,HzP,FzP,MainDP);

 if M2ScoreM+M2ScoreP>=2 then
   Begin
     Setlength(TMain2,M2ScoreM+M2ScoreP);
     IndxL:=0;
     for Indx:=1 to M2Tot do
       Begin
         if Scores2[Indx].ReportSeen='True Positive' then
         Begin
           TMain2[IndxL]:=Scores2[Indx].MLatency;
           Inc(IndxL);
         End;
       End;
       MeanAndStdDev(TMain2,MeanM2,StdM2);
   End
     Else
       Begin
         MeanM2:=0;
         StdM2:=0;
       End;
if (M2ErrorsM+M2ErrorsP>=2) then
   Begin
     SetLength(TMain2,M2ErrorsM+M2ErrorsP);
     IndxL:=0;
```

```
    for Indx := 1 to M2Tot do
      Begin
       if Scores2[Indx].ReportSeen='False Positive' then
         Begin
           TMain2[IndxL]:=Scores2[Indx].MElatency;
           Inc(IndxL);
         End;
       End;
       MeanAndStdDev(TMain2,MeanME2,StdE2);
  End
    Else
      Begin
       MeanME2:=0;
       StdE2:=0;
      End;
With Form1.ADOSummaryMain Do
Begin
  Append;
  Fields.FieldByName('ID').Value:=NameDB;
  Fields.FieldByName('Stage').Value:=Stage;
  Fields.FieldByName('Score M').Value:=M2ScoreM;
  Fields.FieldByName('Total Targets').Value:=M2Tot;
  Fields.FieldByName('Total M NonTargets').Value:=M2NonTargetsM;
  Fields.FieldByName('Total M Targets').Value:=M2TargetsM;
  Fields.FieldByName('Total M Errors').Value:= M2ErrorsM;
  Fields.FieldByName('Total Null Errors M').Value:=MScores[Stage].NullErrorsM;
  Fields.FieldByName('Mean Target Latency M').Value:=M2TLatencyM;   //MeanM1
  Fields.FieldByName('Mean Error Latency M').Value:=M2ELatencyM;
  Fields.FieldByName('STD Targets').Value:=StdM2;
  Fields.FieldByName('STD Errors').Value:=StdE2;
  Fields.FieldByName('Score P').Value:=M2ScoreP;
  Fields.FieldByName('Total P Targets').Value:=M2TargetsP;
  Fields.FieldByName('Total P NonTargets').Value:=M2NonTargetsP;
  Fields.FieldByName('Total P Errors').Value:=M2ErrorsP;
  Fields.FieldByName('Total Null Errors P').Value:=MScores[Stage].NullErrorsP;
  Fields.FieldByName('Mean Target Latency P').Value:=M2TLatencyP;
  Fields.FieldByName('Mean Error Latency P').Value:=M2ELatencyP;
  Fields.FieldByName('Timer1').Value:=T1;
  Fields.FieldByName('Timer2').Value:=T2;
  Fields.FieldByName('Timer3').Value:=T3;
  Fields.FieldByName('M Hits').Value:=M2ScoreM;
  Fields.FieldByName('Targets M').Value:=M2TargetsM;
  Fields.FieldByName('False Alarms M').Value:=M2ErrorsM;
  Fields.FieldByName('M Hit Rate Z').Value:=HzM;
  Fields.FieldByName('M False Alarm Z').Value:=FzM;
  Fields.FieldByName('M Hit Rate').Value:=HRM;
  Fields.FieldByName('M False Alarm Rate').Value:=FAM;
  Fields.FieldByName('D Prime M').Value:=MainDM;
  Fields.FieldByName('P Hits').Value:=M2ScoreP;
  Fields.FieldByName('Targets P').Value:=M2TargetsP;
  Fields.FieldByName('False Alarms P').Value:=M2ErrorsP;
  Fields.FieldByName('P Hit Rate Z').Value:=HzP;
  Fields.FieldByName('P False Alarm Z').Value:=FzP;
  Fields.FieldByName('P Hit Rate').Value:=HRP;
```

```
    Fields.FieldByName('P False Alarm Rate').Value:=FAP;
    Fields.FieldByName('D Prime P').Value:=MainDP;
    UpdateRecord;
    Post;
   End;
End;
if Stage=3 Then
Begin
 M3Tot:=GlobalCounter-1;
 M3TargetsM:=MScores[Stage].MTargetsM;
 M3NonTargetsM:=MScores[Stage].MNonTargetsM;
 M3ScoreM:=MScores[Stage].MScoreM;
 If (M3ScoreM>0) AND (M3TargetsM>0) Then
   M3TLatencyM:=MScores[Stage].MTotM DIV M3ScoreM;
 M3ErrorsM:=MScores[Stage].MErrorsM;
  if (M3NonTargetsM>0) AND (M3ErrorsM>0) then
   M3ELatencyM:=MScores[Stage].ELatencyM DIV M3ErrorsM;
 M3NullErrors:=MScores[Stage].NullErrorsM+MScores[Stage].NullErrorsP;
 if (M3ScoreM>0) AND (M3TargetsM>0) then HRM:=M3ScoreM/M3TargetsM
  Else
    HRM:=0.001;
 if (M3ErrorsM>0) AND (M3NonTargetsM>0) then FAM:=M3ErrorsM/M3NonTargetsM
  Else
    FAM:=0.001;
 Dprime(HrM,FAM,HzM,FzM,MainDM);
 M3TargetsP:=MScores[Stage].MTargetsP;
 M3NonTargetsP:=MScores[Stage].MNonTargetsP;
 M3ScoreP:=MScores[Stage].MScoreP;
 If (M3ScoreP>0) AND (M3TargetsP>0) Then
   M3TLatencyP:=MScores[Stage].MTotP DIV M3ScoreP;
 M3ErrorsP:=MScores[Stage].MErrorsP;
 if (M3NonTargetsP>0) AND (M3ErrorsP>0) then
 M3ELatencyP:=MScores[Stage].ELatencyP DIV M3ErrorsP;
 if (M3ScoreP>0) AND (M3TargetsP>0) then HRP:=M3ScoreP/M3TargetsP
  Else
    HRP:=0.001;
 if (M3ErrorsP>0) AND (M3NonTargetsP>0) then FAP:=M3ErrorsP/M3NonTargetsP
  Else
    FAP:=0.001;
  Dprime(HrP,FAP,HzP,FzP,MainDP);

 if M3ScoreM+M3ScoreP>=2 then
   Begin
    Setlength(TMain3,M3ScoreM+M3ScoreP);
    IndxL:=0;
    for Indx:=1 to M3Tot do
      Begin
       if Scores3[Indx].ReportSeen='True Positive' then
       Begin
         TMain3[IndxL]:=Scores3[Indx].MLatency;
         Inc(IndxL);
       End;
      End;
      MeanAndStdDev(TMain3,MeanM3,StdM3);
```

```
  End
   Else
    Begin
     MeanM3:=0;
     StdM3:=0;
    End;
if (M3ErrorsM+M3ErrorsP>=2) then
  Begin
   SetLength(TMain3,M3ErrorsM+M3ErrorsP);
   IndxL:=0;
   for Indx := 1 to M3Tot do
    Begin
     if Scores3[Indx].ReportSeen='False Positive' then
      Begin
       TMain3[IndxL]:=Scores3[Indx].MElatency;
       Inc(IndxL);
      End;
     End;
     MeanAndStdDev(TMain3,MeanME3,StdE3);
  End
   Else
    Begin
     MeanME3:=0;
     StdE3:=0;
    End;
 With Form1.ADOSummaryMain Do
 Begin
  Append;
  Fields.FieldByName('ID').Value:=NameDB;
  Fields.FieldByName('Stage').Value:=Stage;
  Fields.FieldByName('Score M').Value:=M3ScoreM;
  Fields.FieldByName('Total Targets').Value:=M3Tot;
  Fields.FieldByName('Total M NonTargets').Value:=M3NonTargetsM;
  Fields.FieldByName('Total M Targets').Value:=M3TargetsM;
  Fields.FieldByName('Total M Errors').Value:= M3ErrorsM;
  Fields.FieldByName('Total Null Errors M').Value:=MScores[Stage].NullErrorsM;
  Fields.FieldByName('Mean Target Latency M').Value:=M3TLatencyM;   //MeanM1
  Fields.FieldByName('Mean Error Latency M').Value:=M3ELatencyM;
  Fields.FieldByName('STD Targets').Value:=StdM3;
  Fields.FieldByName('STD Errors').Value:=StdE3;
  Fields.FieldByName('Score P').Value:=M3ScoreP;
  Fields.FieldByName('Total P Targets').Value:=M3TargetsP;
  Fields.FieldByName('Total P NonTargets').Value:=M3NonTargetsP;
  Fields.FieldByName('Total P Errors').Value:=M3ErrorsP;
  Fields.FieldByName('Total Null Errors P').Value:=MScores[Stage].NullErrorsP;
  Fields.FieldByName('Mean Target Latency P').Value:=M3TLatencyP;
  Fields.FieldByName('Mean Error Latency P').Value:=M3ELatencyP;
  Fields.FieldByName('Timer1').Value:=T1;
  Fields.FieldByName('Timer2').Value:=T2;
  Fields.FieldByName('Timer3').Value:=T3;
  Fields.FieldByName('M Hits').Value:=M3ScoreM;
  Fields.FieldByName('Targets M').Value:=M3TargetsM;
  Fields.FieldByName('False Alarms M').Value:=M3ErrorsM;
  Fields.FieldByName('M Hit Rate Z').Value:=HzM;
```

```
    Fields.FieldByName('M False Alarm Z').Value:=FzM;
    Fields.FieldByName('M Hit Rate').Value:=HRM;
    Fields.FieldByName('M False Alarm Rate').Value:=FAM;
    Fields.FieldByName('D Prime M').Value:=MainDM;
    Fields.FieldByName('P Hits').Value:=M3ScoreP;
    Fields.FieldByName('Targets P').Value:=M3TargetsP;
    Fields.FieldByName('False Alarms P').Value:=M3ErrorsP;
    Fields.FieldByName('P Hit Rate Z').Value:=HzP;
    Fields.FieldByName('P False Alarm Z').Value:=FzP;
    Fields.FieldByName('P Hit Rate').Value:=HRP;
    Fields.FieldByName('P False Alarm Rate').Value:=FAP;
    Fields.FieldByName('D Prime P').Value:=MainDP;
    UpdateRecord;
    Post;
  End;
End;

if Stage=4 Then
Begin
  M4Tot:=GlobalCounter-1;
  M4TargetsM:=MScores[Stage].MTargetsM;
  M4NonTargetsM:=MScores[Stage].MNonTargetsM;
  M4ScoreM:=MScores[Stage].MScoreM;
  If (M4ScoreM>0) AND (M4TargetsM>0) Then
     M4TLatencyM:=MScores[Stage].MTotM DIV M4ScoreM;
  M4ErrorsM:=MScores[Stage].MErrorsM;
   if (M4NonTargetsM>0) AND (M4ErrorsM>0) then
     M4ELatencyM:=MScores[Stage].ELatencyM DIV M4ErrorsM;
  M4NullErrors:=MScores[Stage].NullErrorsM+MScores[Stage].NullErrorsP;
  if (M4ScoreM>0) AND (M4TargetsM>0) then HRM:=M4ScoreM/M4TargetsM
    Else
      HRM:=0.001;
  if (M4ErrorsM>0) AND (M4NonTargetsM>0) then FAM:=M4ErrorsM/M4NonTargetsM
    Else
      FAM:=0.001;
   Dprime(HrM,FAM,HzM,FzM,MainDM);
   M4TargetsP:=MScores[Stage].MTargetsP;
   M4NonTargetsP:=MScores[Stage].MNonTargetsP;
   M4ScoreP:=MScores[Stage].MScoreP;
   If (M4ScoreP>0) AND (M4TargetsP>0) Then
     M4TLatencyP:=MScores[Stage].MTotP DIV M4ScoreP;
   M4ErrorsP:=MScores[Stage].MErrorsP;
   if (M4NonTargetsP>0) AND (M4ErrorsP>0) then
   M4ELatencyP:=MScores[Stage].ELatencyP DIV M4ErrorsP;
   if (M4ScoreP>0) AND (M4TargetsP>0) then HRP:=M4ScoreP/M4TargetsP
    Else
      HRP:=0.001;
   if (M4ErrorsP>0) AND (M4NonTargetsP>0) then FAP:=M4ErrorsP/M4NonTargetsP
    Else
      FAP:=0.001;
    Dprime(HrP,FAP,HzP,FzP,MainDP);

   if M4ScoreM+M4ScoreP>=2 then
     Begin
```

```
    Setlength(TMain4,M4ScoreM+M4ScoreP);
    IndxL:=0;
    for Indx:=1 to M4Tot do
      Begin
        if Scores4[Indx].ReportSeen='True Positive' then
        Begin
          TMain4[IndxL]:=Scores4[Indx].MLatency;
          Inc(IndxL);
        End;
      End;
      MeanAndStdDev(TMain4,MeanM4,StdM4);
  End
    Else
      Begin
        MeanM4:=0;
        StdM4:=0;
      End;
if (M4ErrorsM+M4ErrorsP>=2) then
  Begin
    SetLength(TMain4,M4ErrorsM+M4ErrorsP);
    IndxL:=0;
    for Indx := 1 to M4Tot do
      Begin
       if Scores4[Indx].ReportSeen='False Positive' then
         Begin
           TMain4[IndxL]:=Scores4[Indx].MElatency;
           Inc(IndxL);
         End;

       End;
       MeanAndStdDev(TMain4,MeanME4,StdE4);
  End
    Else
      Begin
        MeanME4:=0;
        StdE4:=0;
      End;
 With Form1.ADOSummaryMain Do
 Begin
  Append;
  Fields.FieldByName('ID').Value:=NameDB;
  Fields.FieldByName('Stage').Value:=Stage;
  Fields.FieldByName('Score M').Value:=M4ScoreM;
  Fields.FieldByName('Total Targets').Value:=M4Tot;
  Fields.FieldByName('Total M NonTargets').Value:=M4NonTargetsM;
  Fields.FieldByName('Total M Targets').Value:=M4TargetsM;
  Fields.FieldByName('Total M Errors').Value:= M4ErrorsM;
  Fields.FieldByName('Total Null Errors M').Value:=MScores[Stage].NullErrorsM;
  Fields.FieldByName('Mean Target Latency M').Value:=M4TLatencyM;   //MeanM1
  Fields.FieldByName('Mean Error Latency M').Value:=M4ELatencyM;
  Fields.FieldByName('STD Targets').Value:=StdM4;
  Fields.FieldByName('STD Errors').Value:=StdE4;
  Fields.FieldByName('Score P').Value:=M4ScoreP;
  Fields.FieldByName('Total P Targets').Value:=M4TargetsP;
```

```
      Fields.FieldByName('Total P NonTargets').Value:=M4NonTargetsP;
      Fields.FieldByName('Total P Errors').Value:=M4ErrorsP;
      Fields.FieldByName('Total Null Errors P').Value:=MScores[Stage].NullErrorsP;
      Fields.FieldByName('Mean Target Latency P').Value:=M4TLatencyP;
      Fields.FieldByName('Mean Error Latency P').Value:=M4ELatencyP;
      Fields.FieldByName('Timer1').Value:=T1;
      Fields.FieldByName('Timer2').Value:=T2;
      Fields.FieldByName('Timer3').Value:=T3;
      Fields.FieldByName('M Hits').Value:=M4ScoreM;
      Fields.FieldByName('Targets M').Value:=M4TargetsM;
      Fields.FieldByName('False Alarms M').Value:=M4ErrorsM;
      Fields.FieldByName('M Hit Rate Z').Value:=HzM;
      Fields.FieldByName('M False Alarm Z').Value:=FzM;
      Fields.FieldByName('M Hit Rate').Value:=HRM;
      Fields.FieldByName('M False Alarm Rate').Value:=FAM;
      Fields.FieldByName('D Prime M').Value:=MainDM;
      Fields.FieldByName('P Hits').Value:=M3ScoreP;
      Fields.FieldByName('Targets P').Value:=M3TargetsP;
      Fields.FieldByName('False Alarms P').Value:=M3ErrorsP;
      Fields.FieldByName('P Hit Rate Z').Value:=HzP;
      Fields.FieldByName('P False Alarm Z').Value:=FzP;
      Fields.FieldByName('P Hit Rate').Value:=HRP;
      Fields.FieldByName('P False Alarm Rate').Value:=FAP;
      Fields.FieldByName('D Prime P').Value:=MainDP;
      UpdateRecord;
      Post;
    End;
  End;
End;

Procedure EndPracticeCM(Var PracticeD: Double; Var Adjustment : Real);
Var Indx,IndxL,CountPlus,CountMinus : Integer;
  HR,FA,Hz,Fz: Double;
  TPrac1 : Array of Double;
  NameDB: ShortString;
  MeanP,MeanPE,StdP1,StdE1 : Extended;
  MeanRPlus,MeanRMinus: Real;
BEGIN
   Finished:=True;
   StdP1:=0;
   StdE1:=0;
   PTargets:=Valid;
   PScore:=Counter;
   PErrs:=errors;
   PRTrials:=GlobalCounter-1;
   IF (Counter>0) AND (Valid>0) Then HR:=PScore/PTargets
    Else HR:=0.001;
   IF (Errors>0) And  (PracticeNonTargets>0) Then FA:=Errors/PracticeNonTargets
   Else FA:=0.001;
   DPrime(HR,FA,Hz,Fz,PracticeD);

   If Counter>=2 Then
    Begin
     SetLength(TPrac1,Counter);
```

```
      IndxL:=0;
      For Indx:=1 to PRTrials DO
       Begin
        if ScoresP[Indx].ReportSeen='True Positive' then
         Begin
          TPrac1[IndxL]:=ScoresP[Indx].M1Latency;
          Inc(IndxL);
         End;
       End;
       MeanAndStdDev(TPrac1,MeanP,StdP1);
  //    StdP1:=StdDev(TPrac1);
  //    MeanP:=Mean(TPrac1);
  End
    Else
    Begin
     StdP1:=0;
     MeanP:=0;
    End;

  If Errors>=2 Then
   Begin
    SetLength(TPrac1,Errors);
    IndxL:=0;
    For Indx:=1 to PRTrials Do
     Begin
      if ScoresP[Indx].ReportSeen='False Positive' Then
       Begin
        TPrac1[IndxL]:=ScoresP[Indx].M1Elatency;
        Inc(IndxL);
       End;
     End;
     MeanAndStdDev(TPrac1,MeanPE,StdE1);
//      StdE1:=StdDev(TPrac1);
//      MeanPE:=Mean(TPrac1);
   End
    Else
    Begin
     StdE1:=0;
     MeanPE:=0;
    End;
  NameDB:=CPT.Namecode;

  MeanRPlus:=0;
  MeanRMinus:=0;
  CountPlus:=0;
  CountMinus:=0;
  For Indx:=1 TO PRTrials DO
  Begin
   if (ScoresP[Indx].ReportSeen='True Positive') then
    Begin
     if ScoresP[Indx].R>0 then
      Begin
       MeanRPlus:=MeanRPlus+ScoresP[Indx].R;
       Inc(CountPlus);
```

```
          End;
        if ScoresP[Indx].R<0 then
          Begin
            MeanRMinus:=MeanRMinus+ScoresP[Indx].R;
            Inc(CountMinus);
          End;
      End;
    End;
    if CountPlus>0 then
      MeanRPlus:=MeanRPlus/CountPlus;
    if CountMinus>0 then
      MeanRMinus:=MeanRMinus/CountMinus;
    MeanRMinus:=Abs(MeanRMinus);
    Adjustment:=((MeanRMinus+MeanRPlus) /2);

  //  CreatePSummaryTable;
    Form1.ADOSummaryTable.TableName:='LuminCalibration';
    Form1.ADOSummaryTable.Open;
    Form1.ADOSummaryTable.Active:=True;
      With Form1.ADOSummaryTable Do
      Begin
        Append;
        Fields.FieldByName('ID').Value:=NameDB;
        Fields.FieldByName('Score').Value:=Counter;
        Fields.FieldByName('Total Targets').Value:=Valid;
        Fields.FieldByName('Non Targets').Value:=PracticeNonTargets;
        Fields.FieldByName('Errors').Value:=Errors;
        Fields.FieldByName('Mean Target Latency').Value:=P1Mean;
        Fields.FieldByName('Mean Error Latency').Value:=P1EMean;
        Fields.FieldByName('STD Targets').Value:=STDP1;
        Fields.FieldByName('STD Errors').Value:=StdE1;
        Fields.FieldByName('Timer1').Value:=T1;
        Fields.FieldByName('Timer2').Value:=T2;
        Fields.FieldByName('Timer3').Value:=T3;
        Fields.FieldByName('Hits').Value:=Counter;
        Fields.FieldByName('Targets').Value:=Valid;
        Fields.FieldByName('False Alarms').Value:=P1Errors;
        Fields.FieldByName('Hit Rate Z').Value:=Hz;
        Fields.FieldByName('False Alarm Z').Value:=Fz;
        Fields.FieldByName('Hit Rate').Value:=HR;
        Fields.FieldByName('False Alarm Rate').Value:=FA;
        Fields.FieldByName('D Prime').Value:=PracticeD;
        Fields.FieldByName('Adjustment').Value:=Adjustment;
        UpdateRecord;
        Post;
      End;
    Writeln(OutfileP,Ban2);
    Writeln(OutfileP,'Score:  ',Counter,'  ',' Targets: ', Valid,' Non Targets: ',PracticeNonTargets,'  Errors: ',errors);
    Writeln(OutFileP,'Mean Target Latency:  ',P1Mean,' Std Targets:  ',StdP1:8:3,'  Std Errors:  ',StdE1:8:3,'   Mean
Error Latency:  ',P1EMean);
    Writeln(OutfileP,'END TEST  ','  ',DateToStr(Date),'  ',TimeToStr(Time));
    Writeln(OutfileP,'Actual Practice Time: ',(FormatDateTime('hh:nn:ss',StopTime-StartTime)));
    Writeln(OutfileP,'Timer 1: ',T1,' Timer 2: ',T2,' Timer 3: ',T3);
    Writeln(OutfileP,'Hits: ',Counter,'  Targets: ',Valid,' Hit Rate Z: ',Hz:6:4,'  False Alarms: ',P1Errors,'  Non-Targets:
```

```
',PracticeNonTargets,' False Alarms Z: ',Fz:6:4);
    Writeln(OutfileP,'Hit Rate: ',HR:8:3,' False Alarm Rate: ',FA:8:3,' D Prime for Practice: ',PracticeD:6:4,'Adjustment:
',Adjustment:2:5);
    Writeln(OutfileP);
    Writeln(OutfileP);

    For Indx:= 1 to PRTrials Do
    begin
      Writeln(OutfileP,'Timer value trial ', Indx,' ',Pract[Indx].Timing);
    end;
    Writeln(OutfileP);
    FileFunctions.CloseP;
    Writeln(OutFileP2,'True Positives');
    Writeln(OutfileP2);
    For Indx:=1 TO PRTrials DO
    Begin
      if (ScoresP[Indx].ReportSeen='True Positive') then
        Begin
          Write(OutfileP2,'LumCalib '+NameDB:20,'  ');
          Write(OutfileP2,ScoresP[Indx].TimeT :12);
          Write(OutfileP2,ScoresP[Indx].FirstWordT :12);
          Write(OutfileP2,ScoresP[Indx].SecondWordT :12);
          Write(OutfileP2,ScoresP[Indx].M1Latency :10);
          Write(OutfileP2,ScoresP[Indx].Value :2);
          Write(OutfileP2,ScoresP[Indx].ReportSeen : 16);
          Write(OutfileP2,ScoresP[Indx].KeyPressed : 2);
          Write(OutfileP2,ScoresP[Indx].R : 3);
          Write(OutfileP2,ScoresP[Indx].G : 3);
          Write(OutfileP2,ScoresP[Indx].B : 3);
          Writeln(OutFileP2);
        End;
    End;
    Writeln(OutfileP2);
    Writeln(OutfileP2,'False Positives');
    Writeln(OutfileP2);
    For Indx:=1 TO PRTrials DO
    Begin
      if (ScoresP[Indx].ReportSeen='False Positive') then
        Begin
          Write(OutfileP2,'LumCalib '+NameDB:20,'  ');
          Write(OutfileP2,ScoresP[Indx].TimeT :12);
          Write(OutfileP2,ScoresP[Indx].FirstWordT :12);
          Write(OutfileP2,ScoresP[Indx].SecondWordT :12);
          Write(OutfileP2,ScoresP[Indx].M1ELatency :10);
          Write(OutfileP2,ScoresP[Indx].Value :2);
          Write(OutfileP2,ScoresP[Indx].ReportSeen : 16);
          Write(OutfileP2,ScoresP[Indx].KeyPressed : 2);
          Write(OutfileP2,ScoresP[Indx].R : 3);
          Write(OutfileP2,ScoresP[Indx].G : 3);
          Write(OutfileP2,ScoresP[Indx].B : 3);
          Writeln(OutFileP2);
        End;
    End;
    Writeln(OutfileP2);
```

```
    Writeln(OutfileP2,'True Negatives');
    Writeln(OutfileP2);
    For Indx:=1 TO PRTrials DO
    Begin
      if (ScoresP[Indx].ReportSeen='True Negative') then
        Begin
          Write(OutfileP2,'LumCalib '+NameDB:20,'  ');
          Write(OutfileP2,ScoresP[Indx].TimeE :12);
          Write(OutfileP2,ScoresP[Indx].FirstWordT :12);
          Write(OutfileP2,ScoresP[Indx].SecondWordT :12);
          Write(OutfileP2,ScoresP[Indx].M1ELatency :10);
          Write(OutfileP2,ScoresP[Indx].Value :2);
          Write(OutfileP2,ScoresP[Indx].ReportSeen : 16);
          Write(OutfileP2,ScoresP[Indx].KeyPressed : 2);
          Write(OutfileP2,ScoresP[Indx].R : 3);
          Write(OutfileP2,ScoresP[Indx].G : 3);
          Write(OutfileP2,ScoresP[Indx].B : 3);
          Writeln(OutFileP2);
        End;
    End;
    Writeln(OutfileP2);
    Writeln(OutfileP2,'False Negatives');
    Writeln(OutfileP2);
    For Indx:=1 TO PRTrials DO
    Begin
      if (ScoresP[Indx].ReportSeen='False Negative') then
        Begin
          Write(OutfileP2,'LumCalib '+NameDB:20,'  ');
          Write(OutfileP2,ScoresP[Indx].TimeE :12);
          Write(OutfileP2,ScoresP[Indx].FirstWordT :12);
          Write(OutfileP2,ScoresP[Indx].SecondWordT :12);
          Write(OutfileP2,ScoresP[Indx].M1ELatency :10);
          Write(OutfileP2,ScoresP[Indx].Value :2);
          Write(OutfileP2,ScoresP[Indx].ReportSeen : 16);
          Write(OutfileP2,ScoresP[Indx].KeyPressed : 2);
          Write(OutfileP2,ScoresP[Indx].R : 3);
          Write(OutfileP2,ScoresP[Indx].G : 3);
          Write(OutfileP2,ScoresP[Indx].B : 3);
          Writeln(OutFileP2);
        End;
    End;
    Close(OutfileP2);
End;

Procedure CloseM;
BEGIN
//  Close(InfileM);
  Close(OutfileM);
End;

Procedure CloseP;
BEGIN
//  Close(InfileP);
  Close(OutfileP);
```

```
END;


Procedure OpenInputFiles(Var LTargets,LNon,LPracticeTargets,LPracNonT: Integer);
Var Counts : Integer;
Temp : String;
Begin
Assign(InfileM,FLarge);  //'c:\cpt\StFiles\Large.txt');
   Reset(InfileM);
   Counts:=0;
   While NOT Eof(InfileM) DO
     Begin
       Readln(InfileM,Temp);
       Counts:=Counts+1;
     End;
     LTargets:=Counts;
     SetLength(PracTargets,LTargets+1);
     Reset(InfileM);
   For Counts:= 1 to LTargets DO
     Begin
       Readln(InfileM,PracTargets[Counts].TargetStr);
       PracTargets[Counts].TargetValue:=0;
     End;
   Reset(InfileM);
Assign(InfileM,FSmall); //'c:\cpt\StFiles\Small.txt');
Reset(InfileM);
  COunts:=0;
While NOT Eof(InfileM) DO
     Begin
       Counts:=Counts+1;
       Readln(InfileM,Temp);
     End;
     LNon:=Counts;
     SetLength(PracNonTargets,LNon+1);
     Reset(InfileM);
 For Counts:=1 to LNon DO
 Begin
   Readln(InfileM,PracNonTargets[Counts].NonTargetString);
   PracNonTargets[Counts].NonTargetValue:=1;
 End;
   Reset(InfileM);
   Assign(InfileP,FSmallerCM); //'c:\cpt\Stfiles\SmallerCM.txt');
   Reset(InfileP);
   Counts:=0;
   While NOT Eof(InfileP) DO
     Begin
       Counts:=Counts+1;
       Readln(InfileP,Temp);
     End;
     LPracticeTargets:=Counts;
     SetLength(PracTargetArr,LPracticeTargets+1);
     Reset(InfileP);
   For Counts:= 1 TO LPracticeTargets DO
   Begin
```

```
     Readln(InfileP,PracTargetArr[Counts]);
    End;
    Reset(InfileP);
    Assign(InfileP,FBiggerCM); //'c:\cpt\Stfiles\BiggerCM.txt');
    Reset(InfileP);
    Counts:=0;
    WHile Not EOF(InfileP) Do
     Begin
      Counts:=Counts+1;
      Readln(InfileP,Temp);
     End;
     LPracNonT:=Counts;
     SetLength(PracNonT,LPracNonT+1);
     Reset(InfileP);
     For Counts:= 1 To LPracNonT DO
     Begin
      Readln(InfileP,PracNonT[Counts]);
     End;
     Close(InfileP);
End;

// PROCEDURES CALLED BY THE ENDPRACTICE ENDMAIN AND ENDMAIN2 PROCEDURES
// THEY SIMPLY CLOSE ALL THE OPEN FILES
// ENDMAIN AND ENDMAIN2 BOTH CALL CloseM
Procedure OpenMainFile(VAR Finished: Boolean; Namecode: String);
VAR
Filename,FilenameD: String;
begin
   Filename:=Concat(Fileloc,Namecode+'X1','M',IntToStr(Stage),Ext);
   FilenameD:=Concat(Fileloc,Namecode+'X1','D',IntToStr(Stage),Ext);

if FileExists(Filename) then
   Begin
    Assign(OutfileM,Filename);
    Append(OutfileM);
    Writeln(OutfileM,'BEGIN TEST Phase 1','  ',DateToStr(Date),'  ',TimeToStr(Time));
    Writeln(OutfileM,'ID: ',Namecode,' Gender: ',Gend,' Age: ',Age);
    Finished:=False;
   End;
If NOT FileExists(Filename) then
   Begin
    ReWrite(OutfileM,Filename);
    Writeln(OutfileM,'BEGIN TEST Phase 1','  ',DateToStr(Date),'  ',TimeToStr(Time));
    Writeln(OutfileM,'ID: ',Namecode,' Gender: ',Gend,' Age: ',Age);
   end;
   if Stage<5 Then
    Finished:=False;
//   Phase:='M';
   Rewrite(OutfileC1,FilenameD);
end;

Procedure OpenMainFile2(VAR Finished: Boolean; Namecode: String);
VAR
Filename,FilenameD2: String;
```

```
begin
   Filename:=Concat(Fileloc,Namecode+'X2','M',Ext);
   FilenameD2:=Concat(Fileloc,Namecode+'X2','D2',Ext);
   if FileExists(Filename) then
   Begin
     Assign(OutfileM,Filename);
     Append(OutfileM);
     Writeln(OutfileM,'BEGIN TEST phase 2','  ',DateToStr(Date),'  ',TimeToStr(Time));
     Writeln(OutfileM,'ID: ',Namecode,' Gender: ',Gend,' Age: ',Age);
     Finished:=false;
   End;
If NOT FileExists(Filename) then
   Begin
     Assign(OutfileM,Filename);
     ReWrite(OutfileM);
     Writeln(OutfileM,'BEGIN TEST phase 2','  ',DateToStr(Date),'  ',TimeToStr(Time));
   end;
     Finished:=false;
     Phase:='M2';
     Rewrite(OutfileC2,FilenameD2);
end;

Procedure OpenPracticeFile(VAR Finished : Boolean; Namecode: String);
VAR
filename,FilenameP2 : string;
BEGIN
  filename:=Concat(Fileloc,Namecode+'X2','P',Ext);
  FilenameP2:=Concat(Fileloc,Namecode+'X2','P2',Ext);
  if FileExists(Filename) then
begin
   Assign(OutfileP,Filename);
   Append(OutfileP);
   Writeln(OutfileP,'BEGIN TEST','  ',DateToStr(Date),'  ',TimeToStr(Time));
   Finished:=False;
end;
If NOT FileExists(Filename) then
begin
    ReWrite(OutfileP,Filename);
    Writeln(OutfileP,'BEGIN TEST','  ',DateToStr(Date),'  ',TimeToStr(Time));
    Finished:=False;
end;
  if FileExists(FilenameP2) then
begin
   Assign(OutfileP2,FilenameP2);
   Append(OutfileP2);
   Writeln(OutfileP2,'BEGIN TEST','  ',DateToStr(Date),'  ',TimeToStr(Time));
   Finished:=False;
end;
If NOT FileExists(FilenameP2) then
begin
    ReWrite(OutfileP2,FilenameP2);
    Writeln(OutfileP2,'BEGIN TEST','  ',DateToStr(Date),'  ',TimeToStr(Time));
    Finished:=False;
end;
```

```
END;

Procedure InitScores;
Begin
  Counter:=0;
  Last:=' ';
  valid:=0;
  errors:=0;
  GraphShown:=False;
  Cal:=false;
End;



end.
```

```
unit FileFunctions;


interface
Uses Sysutils,Windows;
Type
  TPleasant = ARRAY OF STRING[13];

Var Qfile,Outfile, Infile, B1Out,B2Out,B3Out,B4Out,B5Out: TextFile;
B1Subliminal, B2Subliminal, B3Subliminal, B4Subliminal, B5Subliminal,
  CCyan, COR, Counterbalance, B2Second, B3Second, B1Text, B2Text,B3Text,B4Text,
  B5Text,Priming,MaskFlag,DMask : Boolean;
  Output,questions,Pleasantfile,Unpleasantfile,Flowerfile,Condomfile,
  Conceptfile,SubFileU, SubFileP,SubFileWhite,SubFileBlack,
  ss1,ss2,ss3,ss4,ss5,ss6,ss7,ssA,SPN,SPNE,
  SPSS,SPSSE,SPSPR,SPSPRE,SPSTRS,SPSTRSE,SPSTRR,SPSTRRE,SetupType  : String;
  TS2,TS3,GS2,GS3,DoubleMask,PL,Ul,FL,CL,FNL: Integer;
  SubBlack,SubWhite: ARRAY[1..10] OF STRING[12];
  SubU,SubP : ARRAY[1..10] OF STRING[13];
  Pleasant: TPleasant;
  Unpleasant: TPleasant;
  Blacks : TPleasant;
  Whites : TPleasant;
  Conceptarr: ARRAY[1..10] OF STRING[25];
  FinalArray: Array[1..140] OF String[120];
  Block1Arr,Block2Arr,Block4Arr,TempB2,TempB4 : Array[1..20] OF Double;
  Block3Arr,Block5Arr,TempB3,TempB5 : Array[1..40] OF Double;



Procedure OpenFiles;
PROCEDURE Readarrays;
Procedure InitArrays;
Procedure CloseFiles;
Procedure OpenB1;
Procedure CloseB1;
Procedure OpenB2;
Procedure CloseB2;
Procedure OpenB3;
Procedure CloseB3;
Procedure OpenB4;
Procedure CloseB4;
Procedure OpenB5;
Procedure CloseB5;
Procedure JoinOutPutFiles;

implementation

uses ControlLogic, IATWinFormUnit;
Procedure TrueFalse(Var Flag: Boolean); External
'TF.dll';

Procedure OpenB1;
```

```
Begin
 AssignFile(B1Out,FileLoc+'b1.txt');
 Rewrite(B1Out);
End;

Procedure CLoseB1;
Begin
  Close(B1Out);
End;

Procedure OpenB2;
Begin
 AssignFile(B2Out,FileLoc+'b2.txt');
 Rewrite(B2Out);
End;

Procedure CLoseB2;
Begin
  Close(B2Out);
End;

Procedure OpenB3;
Begin
  AssignFile(B3Out,FileLoc+'b3.txt');
  Rewrite(B3Out);
End;

Procedure CloseB3;
Begin
  Close(B3Out);
End;

Procedure OpenB4;
Begin
  AssignFile(B4Out,FileLoc+'b4.txt');
  Rewrite(B4Out);
End;

Procedure CloseB4;
Begin
  Close(B4Out);
End;

Procedure OpenB5;
Begin
  AssignFile(B5Out,FileLoc+'b5.txt');
  Rewrite(B5Out);
End;

Procedure CloseB5;
Begin
  Close(B5Out);
End;
```

```
Procedure JoinOutPutFiles;
Var Dex : Integer;  TempString : String;
Begin
 AssignFile(B1Out,FileLoc+'b1.txt');
 Reset(B1Out);
 FOR Dex:= 1 TO  BT1 DO
   Begin
   ReadLn(B1Out,TempString);
   FinalArray[Dex]:=TempString;
   End;
 Close(B1Out);
 AssignFile(B2Out,FileLoc+'b2.txt');
 Reset(B2Out);
 FOR Dex:= BT1+1 TO BT1+BT2 DO
   Begin
     Readln(B2Out,TempString);
     FinalArray[Dex]:=Tempstring;
   End;
   Close(B2Out);
 AssignFile(B3Out,FileLoc+'b3.txt');
 Reset(B3Out);
 For Dex:=(BT1+BT2+1) TO BT1+BT2+BT3 DO
   Begin
     Readln(B3Out,TempString);
     FinalArray[Dex]:=Tempstring;
   End;
 Close(B3Out);
 AssignFile(B4Out,FileLoc+'b4.txt');
 Reset(B4Out);
 For Dex:=(BT1+BT2+BT3+1) TO (BT1+BT2+BT3+BT4) DO
   Begin
     Readln(B4Out,TempString);
     FinalArray[Dex]:=TempString;
   End;
 Close(B4Out);
 AssignFile(B5Out,FileLoc+'b5.txt');
 Reset(B5Out);
 For Dex:=(BT1+BT2+BT3+BT4+1) To BT1+BT2+BT3+BT4+BT5 Do
   Begin
     Readln(B5Out,TempString);
     FinalArray[Dex]:=Tempstring;
   End;
 Close(B5Out);
 For Dex:= 1 TO BT1+BT2+BT3+BT4+BT5 DO
   BEgin
     Writeln(Outfile,FinalArray[Dex]);
   End;
End;


Procedure CloseFiles;
BEGIN
  Close(Outfile);
END;
```

```
Procedure ReadParameters;
Var TempString,M,P : ShortString;
Var R,G,B : Byte;
BEGIN
  COR:=False;
  Readln(Infile,TempString);
  Readln(Infile,Output);              //1
  Readln(Infile,Pleasantfile);        //2
  Readln(Infile,Unpleasantfile);      //3
  Readln(Infile,Flowerfile);          //4
  Readln(Infile,Condomfile);          //5
  Readln(Infile,Conceptfile);         //6
  Readln(Infile,SubFileU);            //7
  Readln(Infile,SubFileP);            //8
  Readln(Infile,SubFileWhite);        //9
  Readln(Infile,SubFileBlack);        //10
  Readln(Infile,FileLocation);        //11
  Readln(Infile,M);
  Readln(Infile,P);
  Readln(Infile,SSA);                 //12
  Readln(Infile,SS1);                 //13
  Readln(Infile,SS2);                 //14
  Readln(Infile,SS3);                 //15
  Readln(Infile,SS4);                 //16
  Readln(Infile,SS5);                 //17
  Readln(Infile,SS6);                 //18
  Readln(Infile,SS7);                 //19
  Readln(Infile,TempString);    // B1Subliminal
    IF (TempString='B1YES') THEN
      B1Subliminal := True ELSE B1Subliminal:=false;
  Readln(Infile,Tempstring);      // B2Subliminal
    IF (Tempstring='B2YES') THEN
      B2Subliminal :=True  ELSE B2Subliminal:=false;
  Readln(Infile,TempString);      // B3Subliminal
    IF (Tempstring='B3YES') THEN
      B3Subliminal:=True   ELSE B3Subliminal:=false;
  Readln(Infile,Tempstring);      // B4Subliminal
    IF (Tempstring='B4YES')  THEN
      B4Subliminal:=True   ELSE B4Subliminal:=false;
  Readln(Infile,TempString);    // B5Subliminal
    IF (Tempstring='B5YES') THEN
      B5Subliminal:=True   ELSE B5Subliminal:=false;
  Readln(Infile,TempString); // using colour or not
    IF (TempString='RANDOM') THEN
    BEGIN
     COR:=True;
     CyanOrRed;
    End;
    IF (TempString='CYAN') THEN
      BEGIN
        COR:=TRUE;
        CCyan :=True;
      END;
```

```
   IF (TempString='RED') THEN
     BEGIN
       COR:=True;
       CCyan :=False;
     End;
   IF COR THEN
     BEGIN
       IF CCyan THEN FileLocationP:=M;//    Concat(FileLocation,'cyan\');
       IF NOT(CCyan) THEN FilelocationP:=P;//Concat(Filelocation,'red\');
      End;
   Counterbalance:= false;
   Readln(Infile,TempString); // COUNTERBALANCE
   If (TempString='COUNTER') THEN
     SetOrder(B2Second,B3Second);
   Readln(Infile,TempString); // B1TEXT B1PIC
     If (Tempstring='B1TEXT') Then B1Text:=True
       Else B1Text:=False;
     Readln(Infile,TempString); // B2TEXT B2PIC
     If (Tempstring='B2TEXT') Then B2Text:=True
       Else B2Text:=False;
     Readln(Infile,TempString); // B3TEXT B3PIC
     If (Tempstring='B3TEXT') Then B3Text:=True
       Else B3Text:=False;
     Readln(Infile,TempString); // B4TEXT B4PIC
     If (Tempstring='B4TEXT') Then B4Text:=True
       Else B4Text:=False;
     Readln(Infile,TempString); // B5TEXT B5PIC
     If (Tempstring='B5TEXT') Then B5Text:=True
       Else B5Text:=False;
     Readln(Infile,TempString); // NOTPRIME  - MEANS THAT IMAGE 2 IS NOT DISPLAYED
     if (TempString='PRIME') then
       Priming:=True
       Else Priming:=false;
     if (TempString='PRIME+R') then
       Begin
         Priming:=True;
         CyanOrRed;
         IF CCyan THEN FileLocationP:=M;//    Concat(FileLocation,'cyan\');
         IF NOT(CCyan) THEN FilelocationP:=P;//Concat(Filelocation,'red\');
       End;
     Readln(Infile,TempString); // Is mask enabled or not
     If(TempString='MASK') Then MaskFlag:=True
     Else MaskFlag:=False;
     Readln(Infile,TempString);
     if (TempString='DOUBLE') then Dmask:=True
     Else DMask:=False;
     Readln(Infile,TempString);
     R:=StrToInt(TempString);
     Readln(Infile,TempString);
     G:=StrToInt(TempString);
     Readln(Infile,TempString);
     B:=StrToInt(TempString);
     TWinform1.Color:=RGB(R,G,B);
End;
```

```
Procedure ReadTimerValues;
VAR TempString : String;
BEGIN
  Readln(Infile,TempString);
  TS2:=StrToInt(TempString);
  Readln(Infile,TempString);
  TS3:=StrToInt(TempString);
  Readln(Infile,Tempstring);
  GS2:=StrToInt(TempString);
  Readln(Infile,Tempstring);
  GS3:=StrToInt(TempString);
  Readln(Infile,TempString);
  DoubleMask:=StrToInt(TempString);
END;

Procedure OpenFiles;
  BEGIN
    Assign(Infile,Parameters);
    Reset(Infile);
    ReadParameters;
    Close(Infile);
    Assign(Infile,TimerValues);
    Reset(Infile);
    ReadTimerValues;
    Close(Infile);
    if FileExists(Output) then
    Begin
      AssignFile(Outfile,Output);
      Append(Outfile);
    End
      Else
        Begin
          AssignFile(Outfile,Output);
          ReWrite(Outfile);
          Reset(Outfile);
          Append(Outfile);
        End;
  END;

  {This procedure reads in the data from the files and stores it in arrays}
PROCEDURE Readarrays;
VAR
Loopten : Integer;
TTStr: String;
//open sub  Graphics White stimulus name-file, read words into an array then close it
BEGIN

// read in the subliminal graphics WHITE stimuli
  Assign(Infile,SubFileWhite);
  Reset(Infile);
  For Loopten:=1 to 10 DO
  BEGIN
    Readln(Infile,TTStr);
```

```
    SubWhite[Loopten]:=TTstr;
  End;
  Close(Infile);

// read in the subliminal graphics BLACK stimuli
  Assign(Infile,SubFileBlack);
  Reset(Infile);
  For Loopten:=1 to 10 DO
  BEGIN
  Readln(infile,TTStr);
  SubBlack[Loopten]:=TTStr;
  End;
  Close(Infile);

// read in the subliminal pleasant text stimuli
    Assign(Infile,SubFileP);
    Reset(Infile);
    For Loopten:= 1 to 10 DO
    BEGIN
      Readln(Infile,TTStr);
      SubP[Loopten]:=TTStr;
    End;
    Close(Infile);

// open sub unpleasant file, read words into an array then close it
    Assign(Infile,SubfileU);
    Reset(Infile);
    For Loopten:= 1 To 10 DO
    BEGIN
      Readln(Infile,TTStr);
      SubU[Loopten]:=TTStr;
    End;
    Close(Infile);

// open pleasant file and read the words into an array then close it
    Assign(Infile,Pleasantfile);
    Reset(Infile);
    Pl:=0;
    REPEAT
      Readln(Infile,TTStr);
      Pl:=Pl+1;
    UNTIL EOF(Infile);
    Reset(Infile);
    SetLength(Pleasant,Pl+1);
    FOR Loopten:=  1 TO Pl DO
    BEGIN
      Readln(Infile,TTStr);
      Pleasant[Loopten]:=TTStr;
    END;
    Close(Infile);

// unpleasant
    Assign(Infile,Unpleasantfile);
    Reset(Infile);
```

```
    Ul:=0;
    REPEAT
      Readln(Infile,TTstr);
      Ul:=Ul+1;
    UNTIL EOF(Infile);
    Reset(Infile);
    SetLength(Unpleasant,Ul+1);
    FOR Loopten:= 1 TO Ul DO
    BEGIN
      Readln(Infile,TTStr);
      Unpleasant[Loopten]:=TTStr;
    END;
    Close(Infile);

// flowerfile
    Assign(Infile,Flowerfile);
    Reset(Infile);
    Fl:=0;
    REPEAT
      Readln(Infile,TTStr);
      Fl:=Fl+1;
    UNTIL EOF(Infile);
    Reset(Infile);
    SetLength(Whites,Fl+1);
FOR Loopten:= 1 TO Fl DO
    BEGIN
      Readln(Infile,TTStr);
      Whites[Loopten]:=TTStr;
    END;
    Close(Infile);

// condomfile
    Assign(Infile,Condomfile);
    Reset(Infile);
    Cl:=0;
    REPEAT
      Readln(Infile,TTStr);
      Cl:=Cl+1;
    UNTIL EOF(Infile);
    Reset(Infile);
    SetLength(Blacks,Cl+1);
FOR Loopten:= 1 TO Cl DO
    BEGIN
      Readln(Infile,TTStr);
      Blacks[Loopten]:=TTStr;
    //  Whites[Loopten]:=TTStr;
    END;
    Close(Infile);

//concept array
    Assign(Infile,Conceptfile);
    Reset(Infile);
FOR Loopten:=1 to 10 DO
    BEGIN
```

```
      Readln(Infile,TTStr);
      Conceptarr[Loopten]:=TTStr;
    END;
    Close(Infile);
END;

Procedure InitArrays;
Var Loopstreet : Integer;
BEGIN
  For Loopstreet:= 1 to 20 DO
  BEGIN
    Block1Arr[Loopstreet]:=0;
    Block2Arr[Loopstreet]:=0;
    Block4Arr[Loopstreet]:=0;
    TempB2[Loopstreet]:=0;
    TempB4[Loopstreet]:=0;
  End;
  For Loopstreet:= 1 to 40 DO
  BEGIN
    Block3Arr[Loopstreet]:=0;
    Block5Arr[Loopstreet]:=0;
    TempB3[Loopstreet]:=0;
    TempB5[Loopstreet]:=0;
  End;
END;


end.
```

```
unit Flicker;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, ComCtrls;

type
  TForm3 = class(TForm)
    Stimulus: TLabel;
    RedBlue: TTimer;
    Button1: TButton;
    CyanYellow: TTimer;
    Stop: TButton;
    UpDown1: TUpDown;
    StimulusM: TLabel;
    Light: TTimer;
    Dark: TTimer;
    Button2: TButton;
    StopM: TButton;
    UpDown2: TUpDown;
    Label1: TLabel;
    Label2: TLabel;
    R: TLabel;
    G: TLabel;
    LblHz: TLabel;
    Hz: TLabel;
    Panel1: TPanel;
    RedBar: TScrollBar;
    GreenBar: TScrollBar;
    BlueBar: TScrollBar;
    Button4: TButton;
    Button5: TButton;
    RedRGB: TLabel;
    GreenRGB: TLabel;
    BlueRGB: TLabel;
    RedRGB1: TLabel;
    BlueRGB1: TLabel;
    GreenRGB1: TLabel;
    Recall: TButton;
    Recall1: TButton;
    REdit: TEdit;
    GEdit: TEdit;
    BEdit: TEdit;
    Enter: TButton;
    Defaults: TButton;
    Reset: TButton;
    Solution: TComboBox;
    Label3: TLabel;
    HzM: TLabel;
    BlueBG: TScrollBar;
    GreenBG: TScrollBar;
```

```
    RedBG: TScrollBar;
    Label4: TLabel;
    All: TScrollBar;
    Shape1: TShape;
    Shape2: TShape;
    Shape3: TShape;
    Shape4: TShape;
    Shape5: TShape;
    Shape6: TShape;
    Shape7: TShape;
    Shape8: TShape;
    Shape9: TShape;
    Label5: TLabel;
    Shape10: TShape;
    Label6: TLabel;
    ForeGround: TScrollBar;
    Shape14: TShape;
    Label7: TLabel;
    BackGround: TScrollBar;
    procedure Button1Click(Sender: TObject);
    procedure RedBlueTimer(Sender: TObject);
    procedure CyanYellowTimer(Sender: TObject);
    procedure StopClick(Sender: TObject);
    procedure UpDown1ChangingEx(Sender: TObject; var AllowChange: Boolean;
      NewValue: Smallint; Direction: TUpDownDirection);
    procedure Button2Click(Sender: TObject);
    procedure LightTimer(Sender: TObject);
    procedure DarkTimer(Sender: TObject);
    procedure UpDown2ChangingEx(Sender: TObject; var AllowChange: Boolean;
      NewValue: Smallint; Direction: TUpDownDirection);
    procedure StopMClick(Sender: TObject);
    procedure RChangeChangingEx(Sender: TObject; var AllowChange: Boolean;
      NewValue: Smallint; Direction: TUpDownDirection);

    procedure ScrollChange(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button5Click(Sender: TObject);
    procedure RecallClick(Sender: TObject);
    procedure Recall1Click(Sender: TObject);
    procedure EnterClick(Sender: TObject);
    procedure DefaultsClick(Sender: TObject);
    procedure ResetClick(Sender: TObject);
    procedure SolutionChange(Sender: TObject);
    procedure AllChange(Sender: TObject);
    procedure ForeGroundChange(Sender: TObject);
    procedure BackGroundChange(Sender: TObject);

  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
```

```
  Form3: TForm3;
 CValue1,CValue2,Red,Green,Blue,Red1,Green1,Blue1: Integer;
implementation

{$R *.dfm}

procedure TForm3.AllChange(Sender: TObject);
begin
  RedBG.Position:=All.Position;
  GreenBG.Position:=All.Position;
  BlueBG.Position:=All.Position;
  Panel1.Color := RGB(RedBar.Position,GreenBar.Position, BlueBar.Position);
end;

procedure TForm3.BackGroundChange(Sender: TObject);
begin
StimulusM.Color:=RGB(Background.Position,Background.Position,Background.Position);
end;

procedure TForm3.Button1Click(Sender: TObject);
begin
RedBlue.Enabled:=true;
Hz.Caption:=IntToStr(1000 Div CyanYellow.Interval);
end;

procedure TForm3.Button2Click(Sender: TObject);
begin
  Light.Enabled:=True;
  HzM.Caption:=IntToStr(1000 Div Dark.Interval);
end;

procedure TForm3.ResetClick(Sender: TObject);
begin
  Light.Interval:=10;
  Dark.Interval:=10;
  RedBlue.Interval:=10;
  CyanYellow.Interval:=10;
  Hz.Caption:=IntToStr(1000 Div RedBlue.Interval);
end;

procedure TForm3.ForeGroundChange(Sender: TObject);
begin
  StimulusM.Font.Color:=RGB(Foreground.Position,Foreground.Position,Foreground.Position);
end;

procedure TForm3.ScrollChange(Sender: TObject);
begin
  Panel1.Color := RGB(RedBar.Position,GreenBar.Position, BlueBar.Position);
end;


procedure TForm3.SolutionChange(Sender: TObject);
begin
If Solution.ItemIndex=0 then
```

```
 Begin
  Red:=0;
  RedRgb.Caption:=IntToStr(Red);
  Green:=44;
  GreenRgb.Caption:=IntToStr(Green);
  Blue:=255;
  BlueRgb.Caption:=IntToStr(Blue);
  R.Color:=RGB(Red,Green,Blue);
  Red1:=200;
  RedRgb1.Caption:=IntToStr(Red1);
  Green1:=0;
  GreenRgb1.Caption:=IntToStr(Green1);
  Blue1:=0;
  BlueRgb1.Caption:=IntToStr(Blue1);
  G.Color:=RGB(Red1,Green1,Blue1);
 End;
 If Solution.ItemIndex=1 then
 Begin
  Red:=0;
  RedRgb.Caption:=IntToStr(Red);
  Green:=255;
  GreenRgb.Caption:=IntToStr(Green);
  Blue:=255;
  BlueRgb.Caption:=IntToStr(Blue);
  R.Color:=RGB(Red,Green,Blue);
  Red1:=255;
  RedRgb1.Caption:=IntToStr(Red1);
  Green1:=255;
  GreenRgb1.Caption:=IntToStr(Green1);
  Blue1:=0;
  BlueRgb1.Caption:=IntToStr(Blue1);
  G.Color:=RGB(Red1,Green1,Blue1);
 End;
If Solution.ItemIndex=2 then
Begin
  Red:=0;
  RedRgb.Caption:=IntToStr(Red);
  Green:=255;
  GreenRgb.Caption:=IntToStr(Green);
  Blue:=255;
  BlueRgb.Caption:=IntToStr(Blue);
  R.Color:=RGB(Red,Green,Blue);
  Red1:=128;
  RedRgb1.Caption:=IntToStr(Red1);
  Green1:=255;
  GreenRgb1.Caption:=IntToStr(Green1);
  Blue1:=0;
  BlueRgb1.Caption:=IntToStr(Blue1);
  G.Color:=RGB(Red1,Green1,Blue1);
End;
 If Solution.ItemIndex=3 then
Begin
  Red:=255;
  RedRgb.Caption:=IntToStr(Red);
```

```
   Green:=0;
   GreenRgb.Caption:=IntToStr(Green);
   Blue:=0;
   BlueRgb.Caption:=IntToStr(Blue);
   R.Color:=RGB(Red,Green,Blue);
   Red1:=0;
   RedRgb1.Caption:=IntToStr(Red1);
   Green1:=255;
   GreenRgb1.Caption:=IntToStr(Green1);
   Blue1:=0;
   BlueRgb1.Caption:=IntToStr(Blue1);
   G.Color:=RGB(Red1,Green1,Blue1);
 End;
end;

procedure TForm3.Button4Click(Sender: TObject);
begin
    Red:=(RedBar.Position);
    Green:=GreenBar.Position;
    Blue:=BlueBar.Position;
    RedRGB.Caption:=IntToStr(Red);
    GreenRGB.Caption:=IntToStr(Green);
    BlueRGB.Caption:=IntToStr(Blue);
    R.Color:=RGB(Red,Green,Blue);
end;

procedure TForm3.Button5Click(Sender: TObject);
begin
    Red1:=(RedBar.Position);
    Green1:=GreenBar.Position;
    Blue1:=BlueBar.Position;
    RedRGB1.Caption:=IntToStr(Red1);
    GreenRGB1.Caption:=IntToStr(Green1);
    BlueRGB1.Caption:=IntToStr(Blue1);
    G.Color:=RGB(Red1,Green1,Blue1);
end;



procedure TForm3.CyanYellowTimer(Sender: TObject);
begin
  Stimulus.Color:=RGB(RedBG.Position,GreenBG.Position,BlueBG.Position);
  Stimulus.Font.Color:=RGB(Red,Green,Blue);
  CyanYellow.Enabled:=false;
  RedBlue.Enabled:=True;
end;

procedure TForm3.DarkTimer(Sender: TObject);
begin
  StimulusM.Color:=RGB(Background.Position,Background.Position,Background.Position);
  StimulusM.Font.Color:=RGB(Foreground.Position,Foreground.Position,Foreground.Position);
  Dark.Enabled:=False;
  Light.Enabled:=True;
end;
```

```
procedure TForm3.DefaultsClick(Sender: TObject);
begin
  Light.Interval:=70;
  Dark.Interval:=70;
  RedBlue.Interval:=70;
  CyanYellow.Interval:=70;
  Hz.Caption:=IntToStr(1000 Div RedBlue.Interval);
end;

procedure TForm3.EnterClick(Sender: TObject);
begin
   RedBar.Position:=StrToInt(REdit.Text);
   GreenBar.Position:=StrToInt(GEdit.Text);
   BlueBar.Position:=StrToInt(BEdit.Text);
end;

procedure TForm3.LightTimer(Sender: TObject);
begin
  StimulusM.Color :=RGB(Foreground.Position,Foreground.Position,Foreground.Position);
  StimulusM.Font.Color:=RGB(Background.Position,Background.Position,Background.Position);
  Light.Enabled:=False;
  Dark.Enabled:=True;
end;

procedure TForm3.RChangeChangingEx(Sender: TObject; var AllowChange: Boolean;
  NewValue: Smallint; Direction: TUpDownDirection);
begin
  If Direction=updUp Then
  Begin
    CValue1:=CValue1+1;
  End;
  If Direction=updDown Then
  Begin
    CValue1:=CValue1-1;
  End;
  R.Font.Color:=RGB(CValue1,0,0);
end;

procedure TForm3.Recall1Click(Sender: TObject);
begin
  RedBar.Position:=Red1;
  GreenBar.Position:=Green1;
  BlueBar.Position:=Blue1;
end;

procedure TForm3.RecallClick(Sender: TObject);
begin
  RedBar.Position:=Red;
  GreenBar.Position:=Green;
  BlueBar.Position:=Blue;
end;

procedure TForm3.RedBlueTimer(Sender: TObject);
```

```
begin
Stimulus.Color:=RGB(RedBG.Position,GreenBG.Position,BlueBG.Position);//ClGray;     ////ActiveBorder;//Aqua;
Stimulus.Font.Color:=RGB(Red1,Green1,Blue1);
RedBlue.Enabled:=false;
CyanYellow.Enabled:=true;
end;


procedure TForm3.StopClick(Sender: TObject);
begin
CyanYellow.Enabled:=false;
RedBlue.Enabled:=False;
end;

procedure TForm3.StopMClick(Sender: TObject);
begin
  Light.Enabled:=False;
  Dark.Enabled:=False;
end;

procedure TForm3.UpDown1ChangingEx(Sender: TObject; var AllowChange: Boolean;
  NewValue: Smallint; Direction: TUpDownDirection);
  Var
    TimeP : Integer;
begin
  If Direction=updUp Then
  Begin
     CyanYellow.Interval:=CyanYellow.Interval+10;
     RedBlue.Interval:=RedBlue.Interval+10;
  End;
  If Direction=updDown Then
  Begin
     CyanYellow.Interval:=CyanYellow.Interval-10;
     RedBlue.Interval:=RedBlue.Interval-10;
  End;
  TimeP:=CyanYellow.Interval;
  Label1.Caption:=IntToStr(TimeP);
  Hz.Caption:=IntToStr(1000 Div TimeP);
end;

procedure TForm3.UpDown2ChangingEx(Sender: TObject; var AllowChange: Boolean;
  NewValue: Smallint; Direction: TUpDownDirection);
  Var TimeM : Integer;
begin
  If Direction=updUp Then
  Begin
    Light.Interval:=Light.Interval+10;
    Dark.Interval:=Dark.Interval+10;
  End;
  If Direction=updDown Then
  Begin
    Light.Interval:=Light.Interval-10;
    Dark.Interval:=Dark.Interval-10;
  End;
```

```
  TimeM:=Light.Interval;
  Label2.Caption:=IntToStr(TimeM);
  HzM.Caption:=IntToStr(1000 Div Dark.Interval);
end;

Begin
Red:=0;
Red1:=0;
Green:=255;
Green1:=255;
Blue:=255;
Blue1:=0;
CValue1:=172;
CValue2:=255;//29952;


end.
```

```
unit Graph;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Series, TeEngine, ExtCtrls, TeeProcs, Chart, StdCtrls, TeeFunci, DB,
  ADODB, Grids, DBGrids, DBCtrls, Mask, DBChart;

type
  TForm2 = class(TForm)
    Dprime1: TLabel;
    SaveDialog1: TSaveDialog;
    DBGrid1: TDBGrid;
    ADOConnection1: TADOConnection;
    ADOTable1: TADOTable;
    DataSource1: TDataSource;
    DBNavigator1: TDBNavigator;
    DBEdit1: TDBEdit;
    DBGrid3: TDBGrid;
    DataSource2: TDataSource;
    ADOQuery1: TADOQuery;
    DBChart1: TDBChart;
    Button1: TButton;
    Button2: TButton;
    Series1: TBarSeries;
    Series2: TBarSeries;
    DBChart2: TDBChart;
    BarSeries1: TBarSeries;
    BarSeries2: TBarSeries;
    Series3: TBarSeries;
    Series4: TBarSeries;
    Button3: TButton;
    Button4: TButton;
    Procedure PlotDPrime(Participant: String);
    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form2: TForm2;
  Participant : String;

implementation
Uses CPT,Login;
{$R *.dfm}
```

```
procedure TForm2.Button1Click(Sender: TObject);
begin
  Form2.ADOConnection1.Connected:=False;
  Form2.ADOConnection1.ConnectionString:=Login.CString;
  Form2.ADOConnection1.Connected:=True;
  Form2.ADOTable1.Connection:=Form2.ADOConnection1;
  Form2.ADOTable1.Open;
  Form2.ADOTable1.Active:=True;
  Form2.ADOQuery1.Connection:=Form2.ADOConnection1;
  Participant:=CPT.Namecode;
  PlotDPrime(Participant);
end;

procedure TForm2.Button2Click(Sender: TObject);
Var ID : ShortString;
   QString : ShortString;
begin
   ID:=DbEdit1.Text;
   QString:=('Select * from mainstages WHERE `ID`='+ ID +' ORDER BY `Stage` ');
   ADOQuery1.SQL.Text:=QString;
   ADOQuery1.Active:=True;
   ADOQuery1.Locate('ID',DbEdit1.Text,[]);
end;

procedure TForm2.Button3Click(Sender: TObject);
begin
 if SaveDialog1.Execute() then
   DbChart2.SaveToBitmapFile(SaveDialog1.FileName);
end;

procedure TForm2.Button4Click(Sender: TObject);
begin
 if SaveDialog1.Execute() then
   DbChart1.SaveToBitmapFile(SaveDialog1.FileName);
end;

procedure TForm2.PlotDPrime(Participant: String);
Var ID : ShortString;
   QString : ShortString;
begin
   ID:=Participant;
   QString:=('Select * from mainstages WHERE `ID`='+ ID +' ORDER BY `Stage` ');
   ADOQuery1.SQL.Text:=QString;      //('Select * from mainstages WHERE `ID`= ID  ORDER BY `Stage` ');
   ADOQuery1.Active:=True;
   ADOQuery1.Locate('ID',DbEdit1.Text,[]);
end;

end.
```

Image Processor

BackGround

0 | 201 | 254

0 | 90 | 220

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

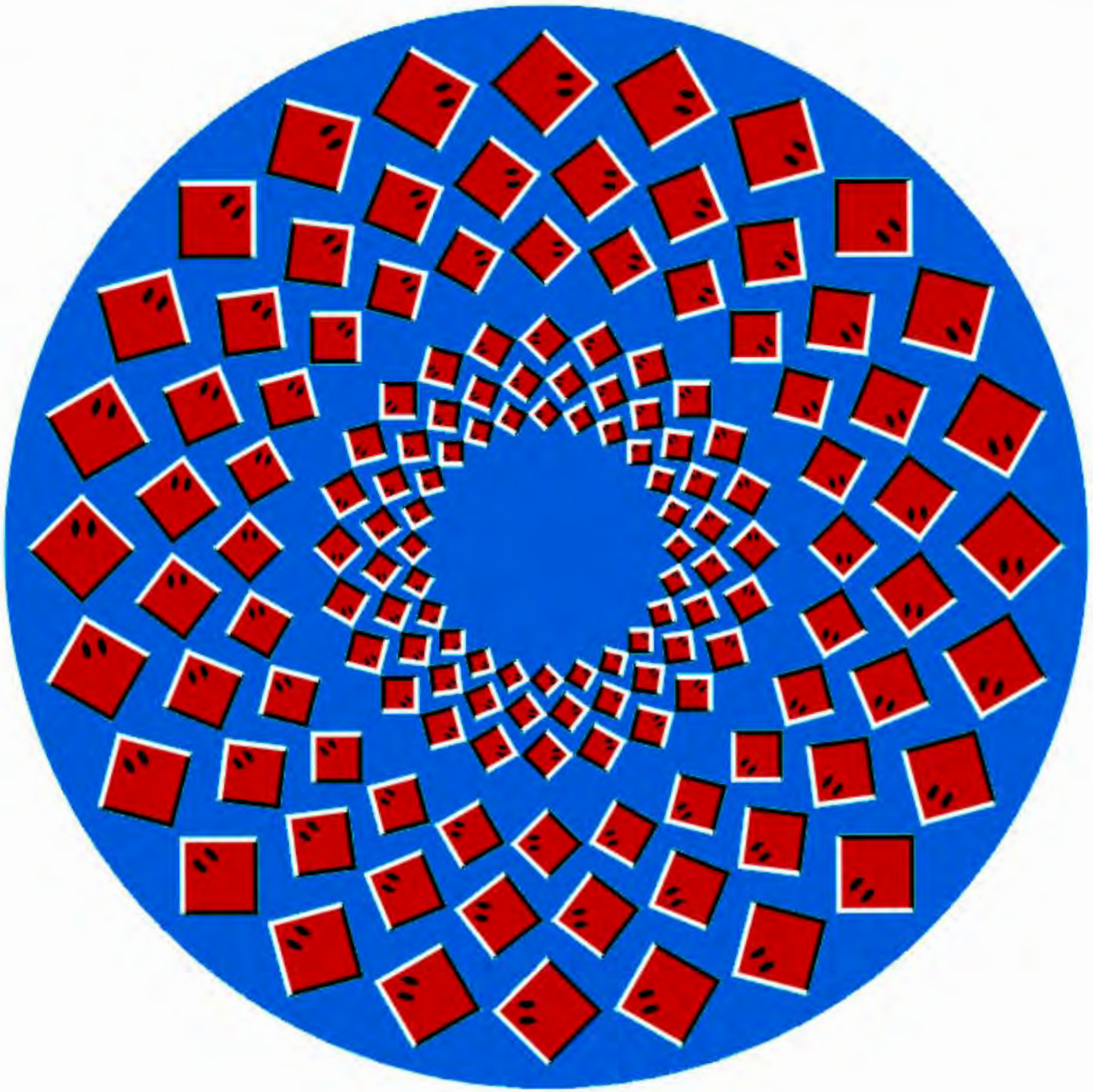


Image Processor

BackGround

0 | 201 | 254

0 | 90 | 255

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

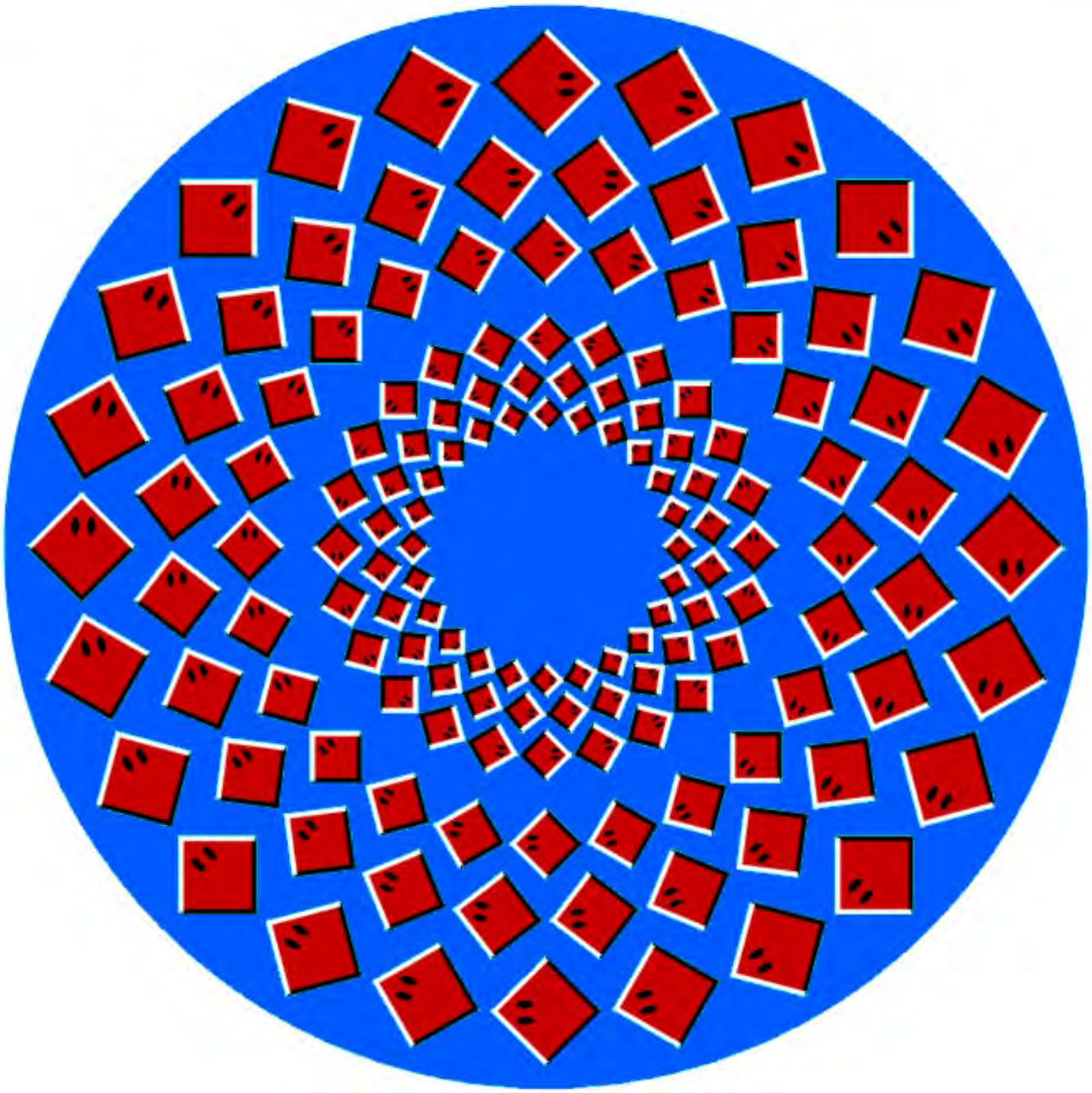


Image Processor

BackGround

0 | 201 | 254

0 | 100 | 255

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

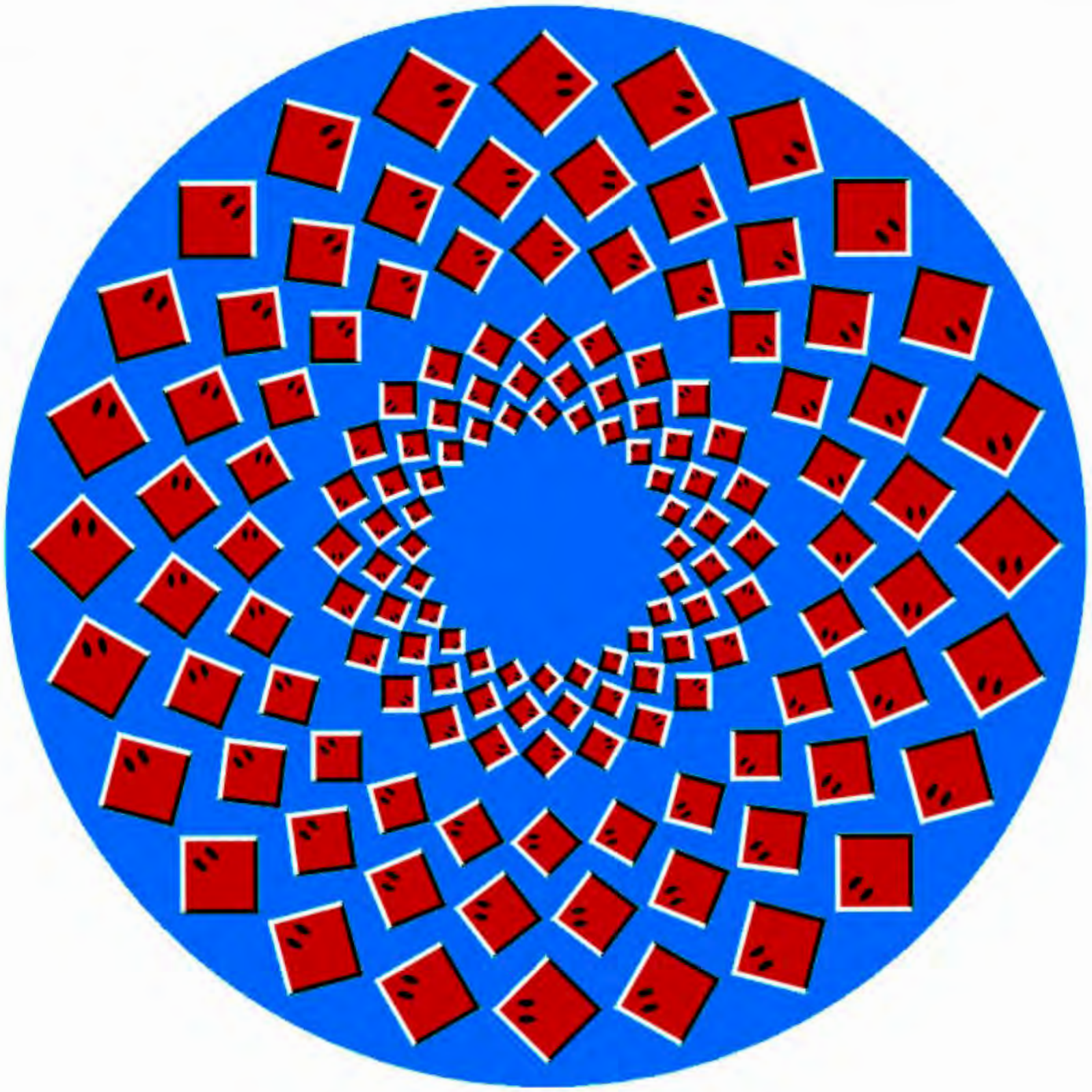


Image Processor

BackGround

50 120 50

0 100 0

Enabled

ForeGround

180 180 180

0 200 200

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

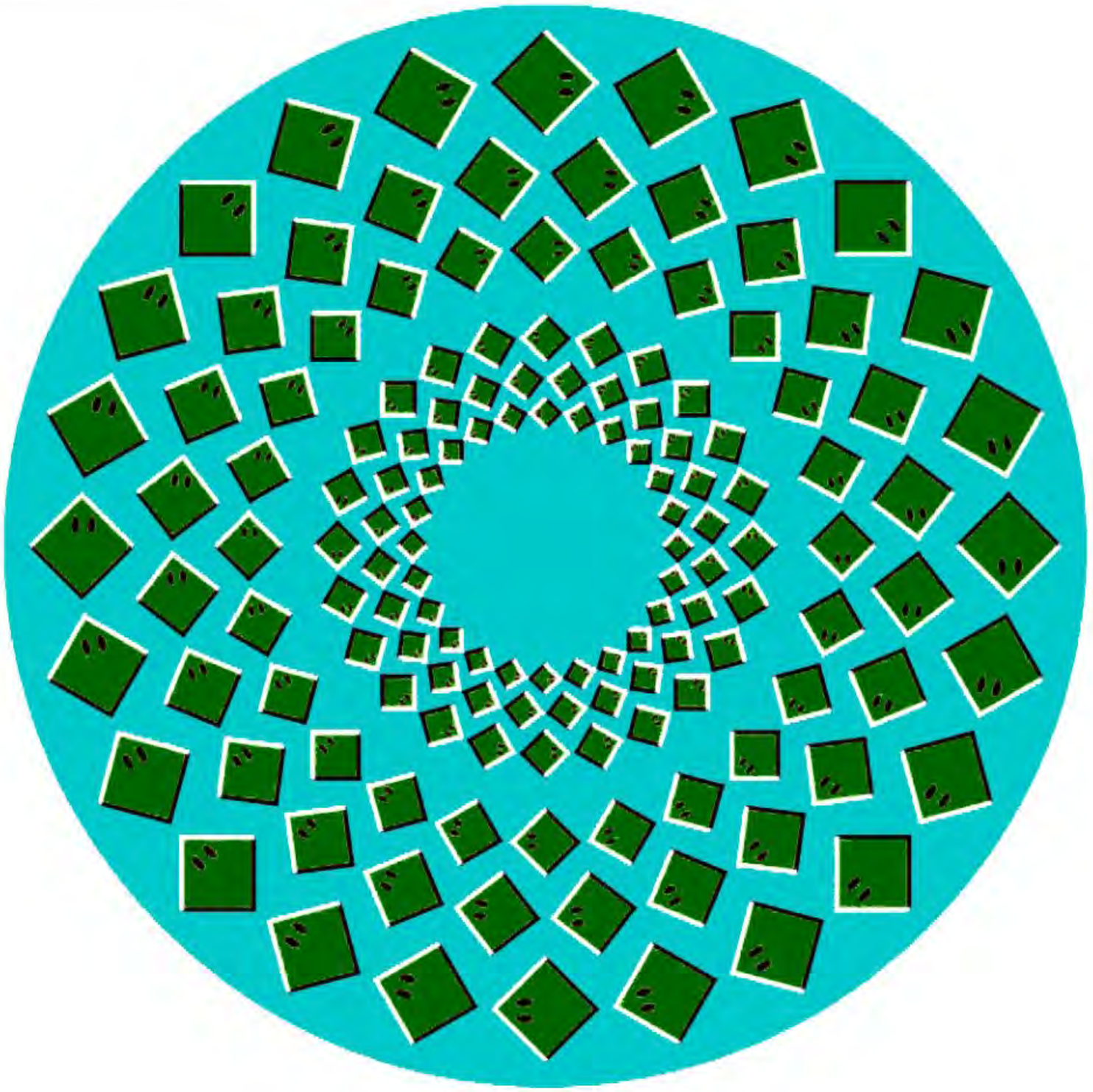


Image Processor

BackGround

0 | 201 | 254

0 | 130 | 255

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

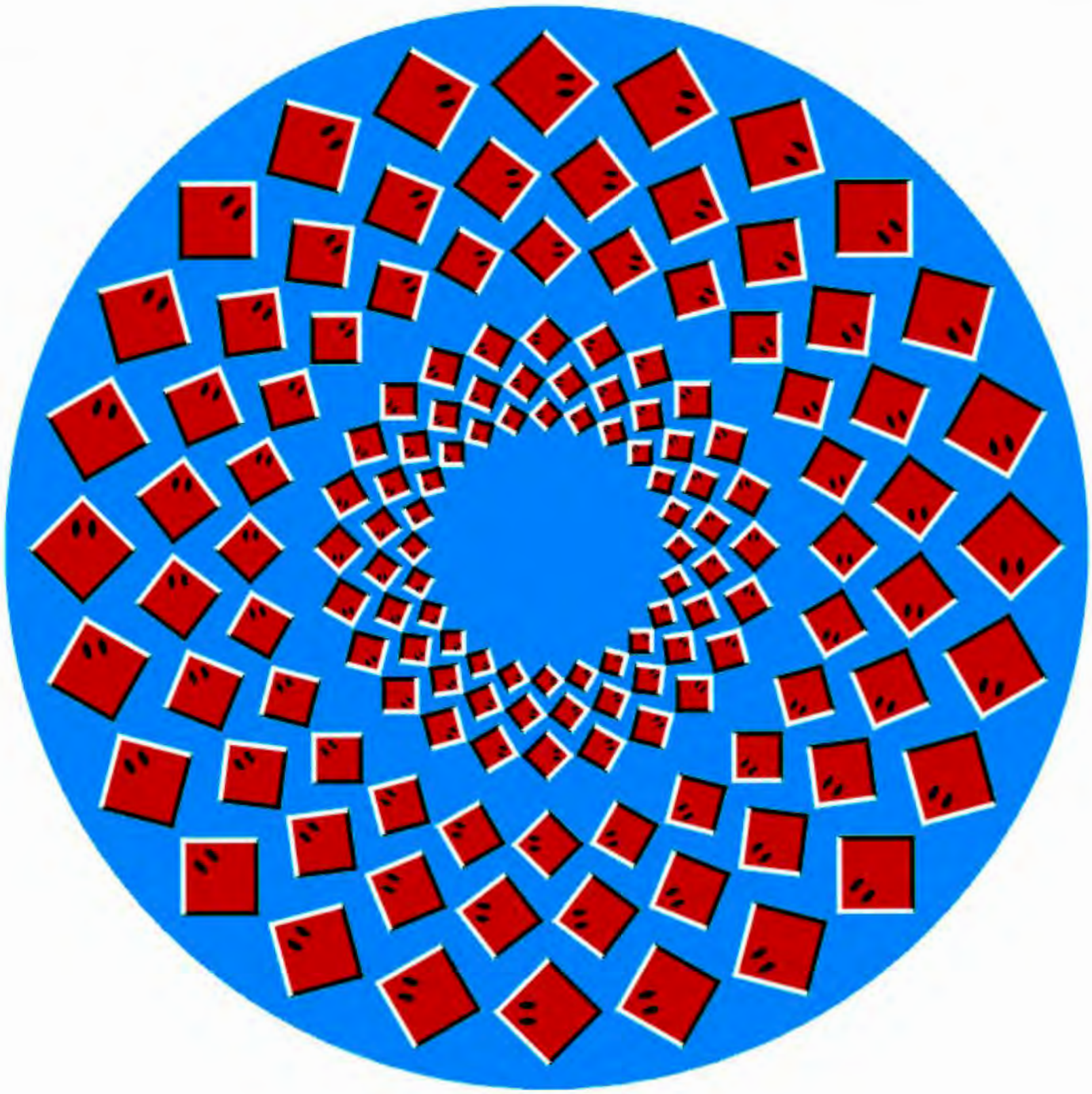


Image Processor

BackGround

0 | 201 | 254

0 | 150 | 255

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

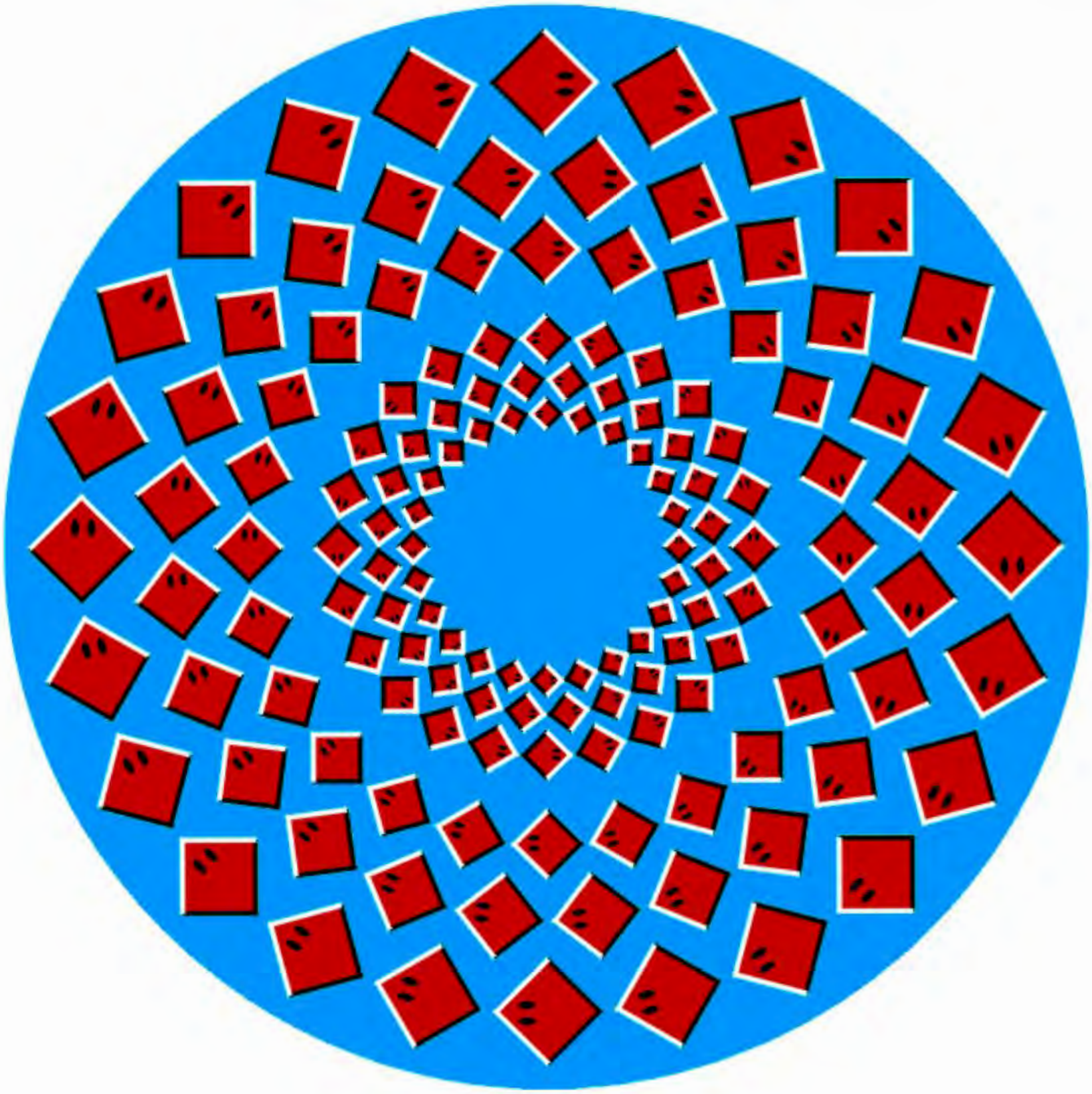


Image Processor

BackGround

0 | 201 | 254

0 | 230 | 0

Enabled

ForeGround

255 | 255 | 255

180 | 0 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

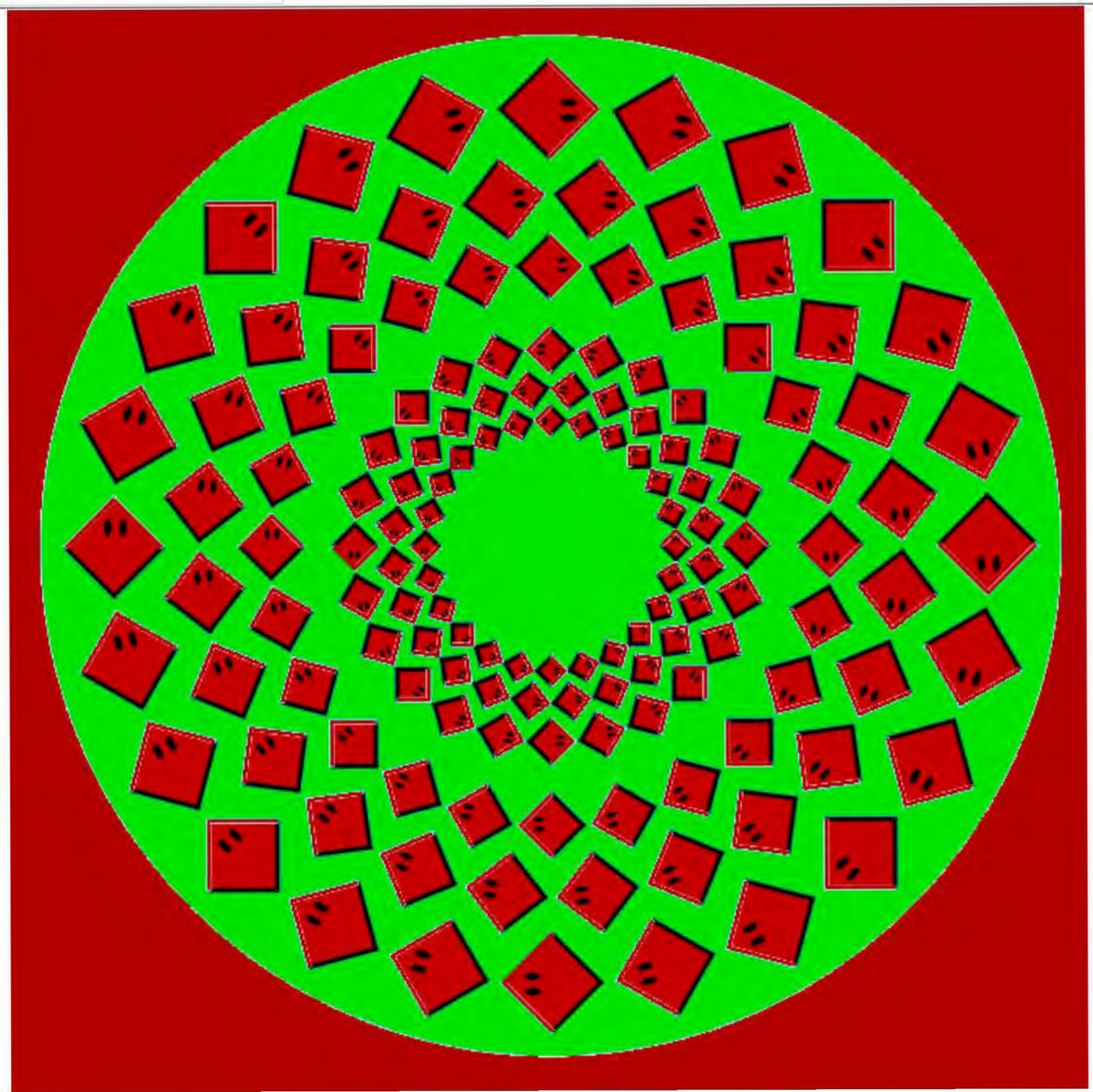


Image Processor

BackGround

50 120 50

0 200 0

Enabled

ForeGround

180 180 180

0 200 200

Enabled

Expander1

Apply

Reload and Apply

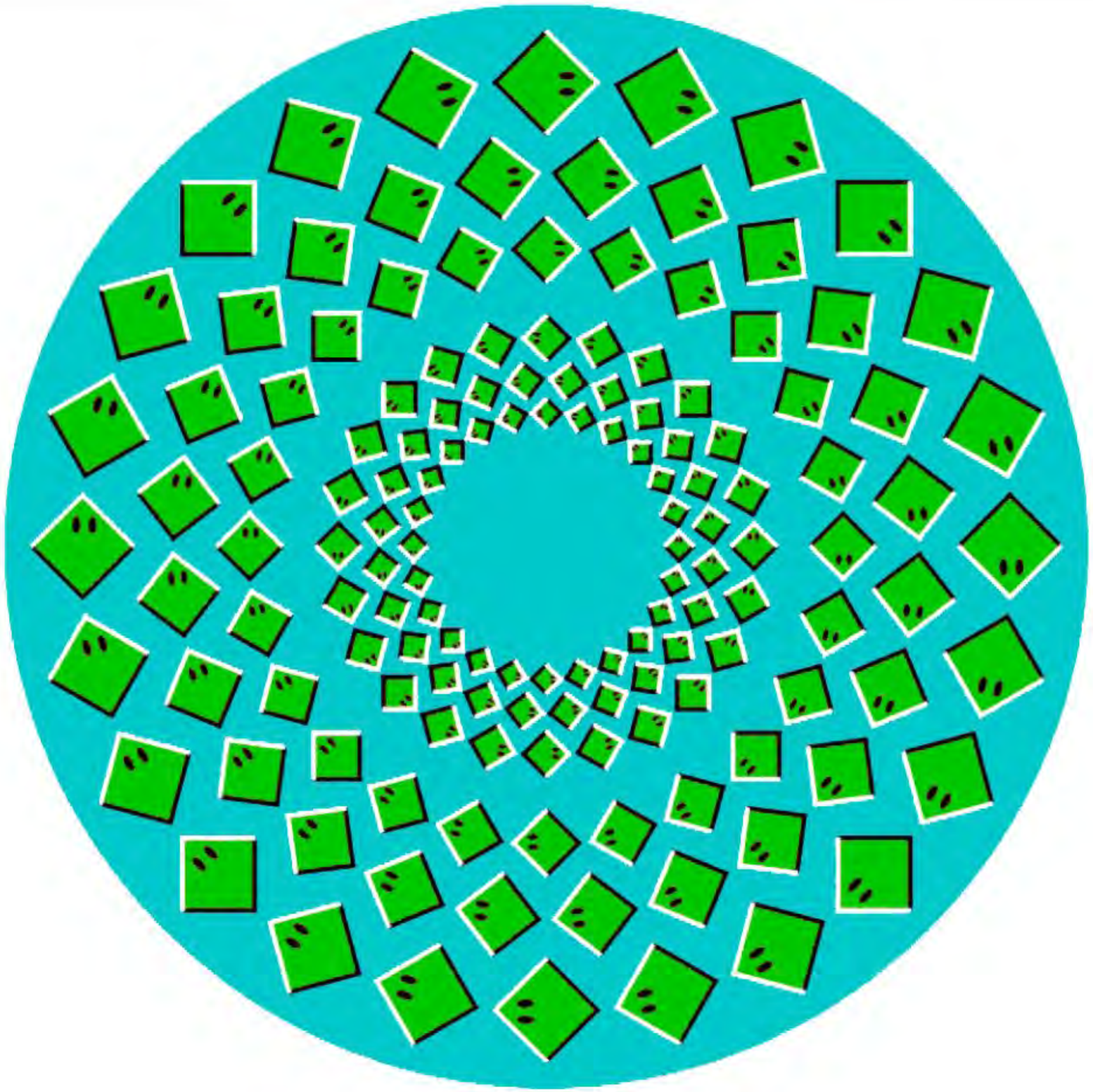
Rotate CW Rotate CCW

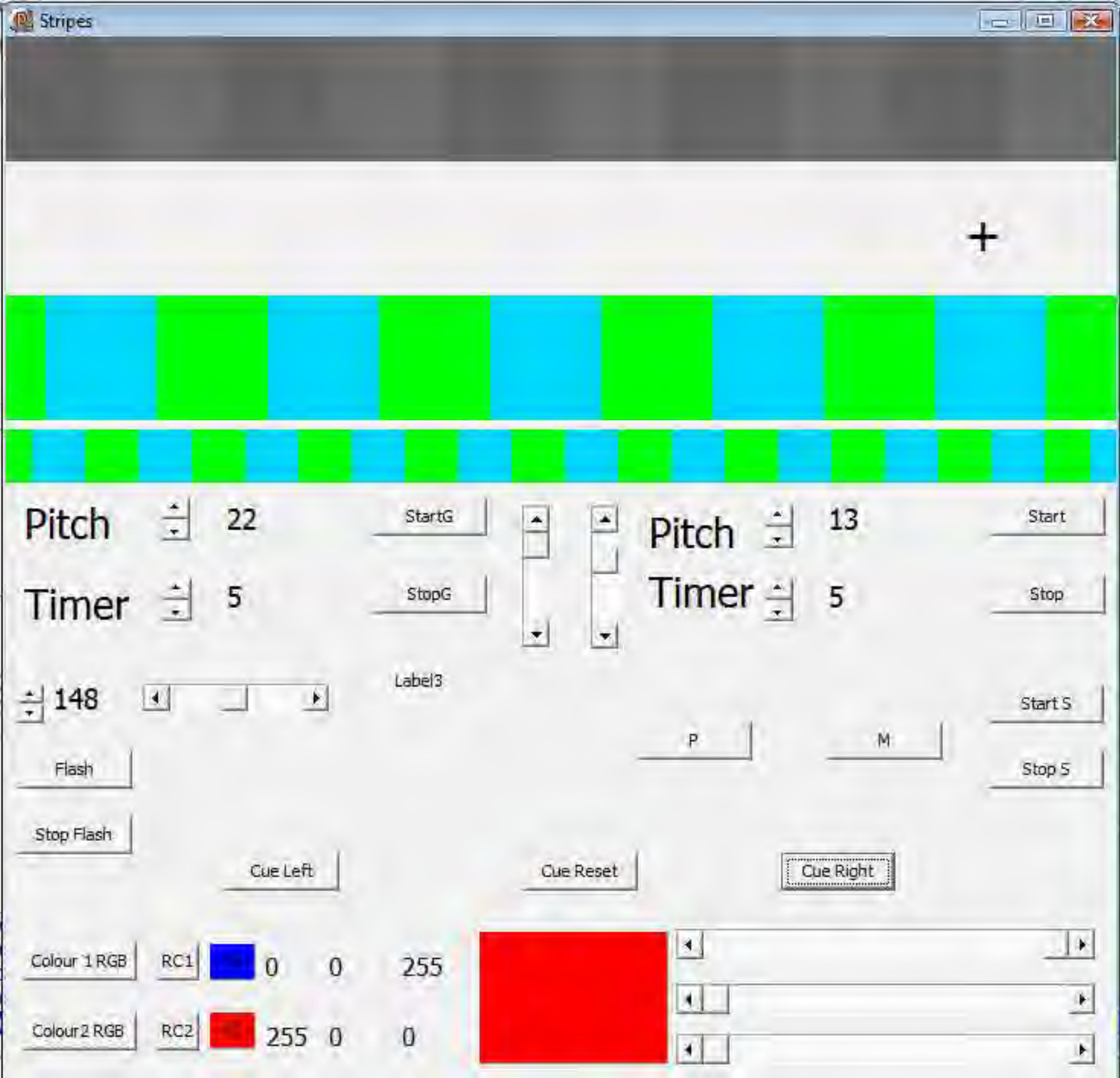
0°

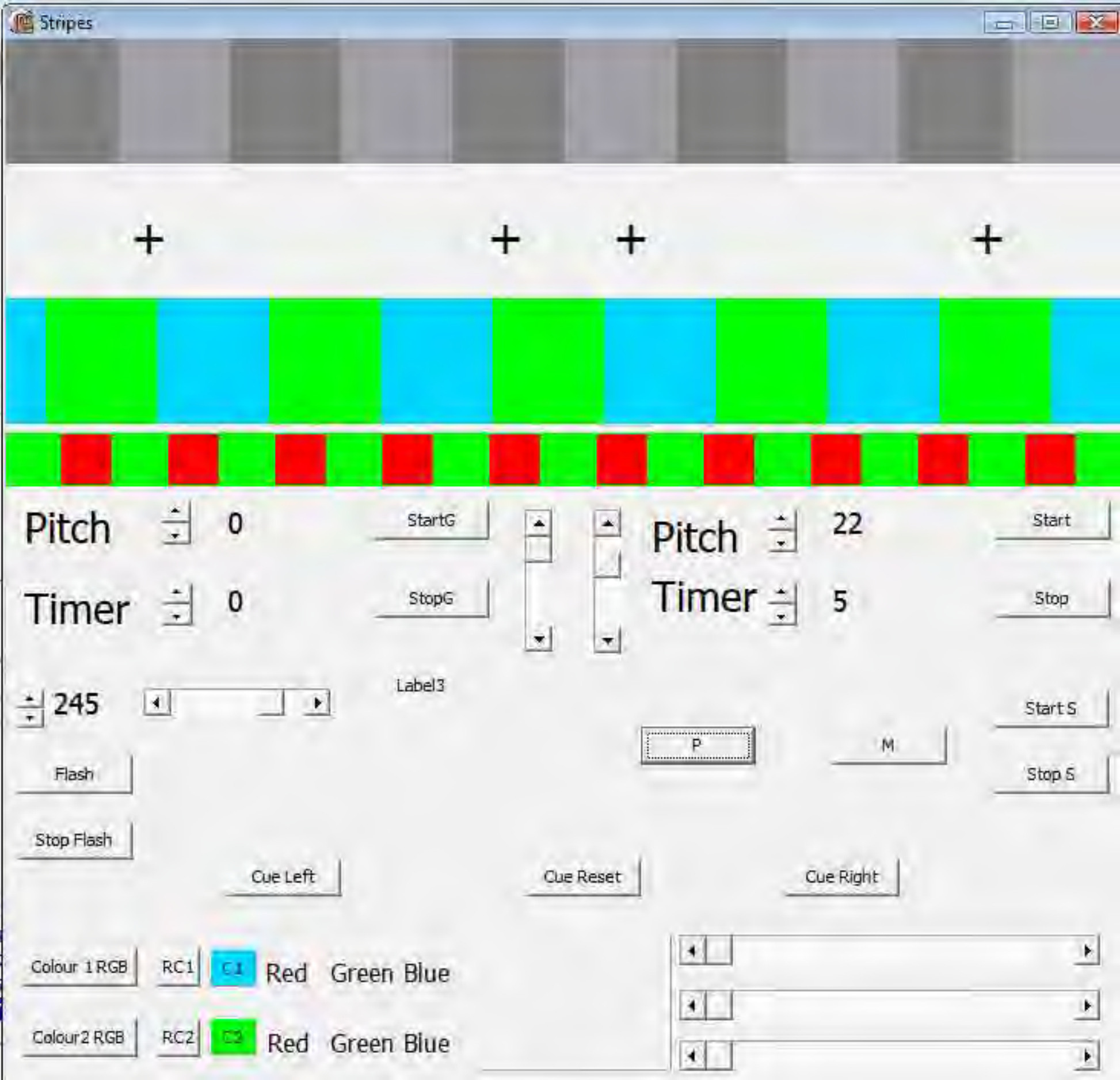
Save Image

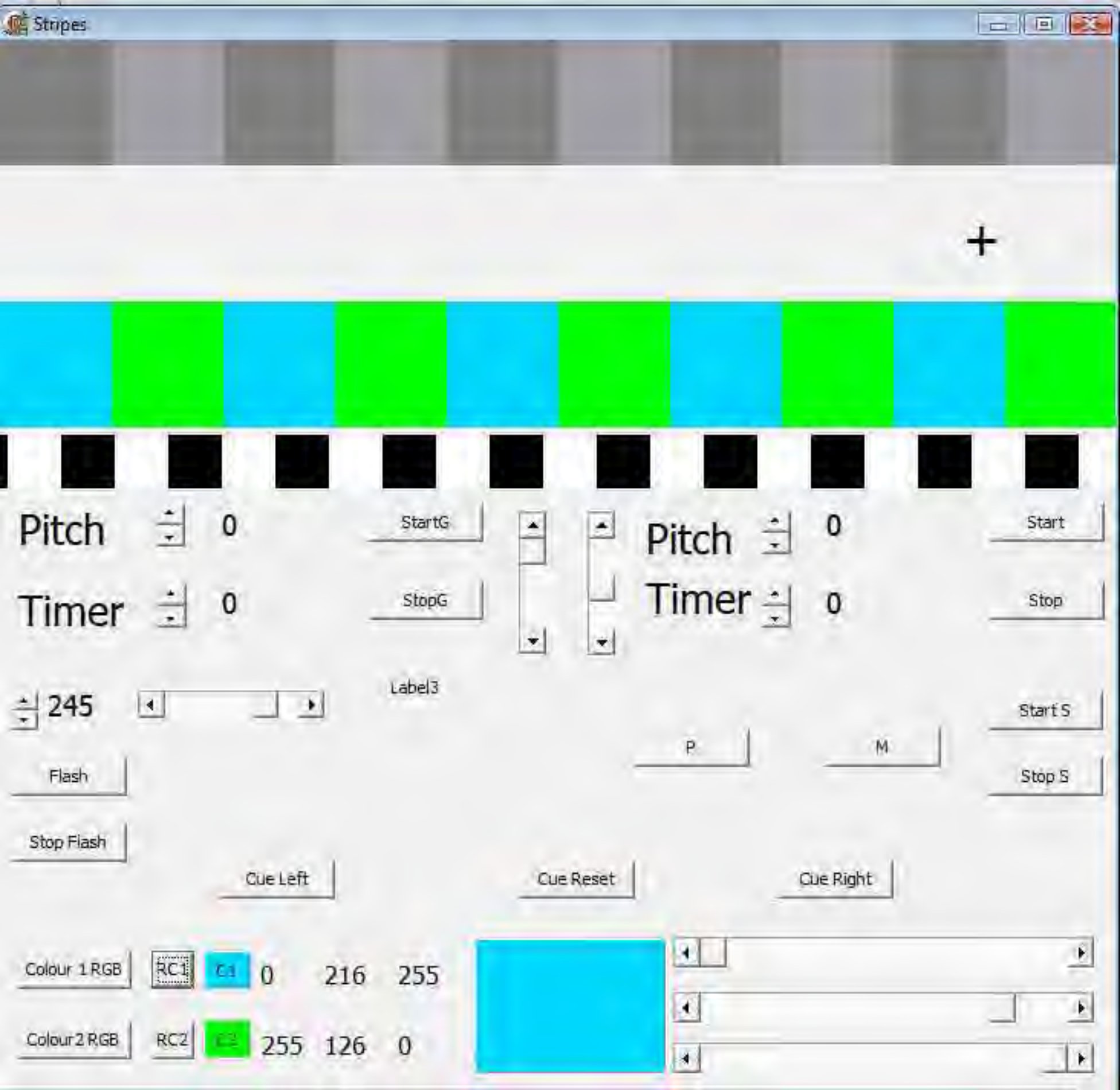
Load Image

Reload Image




Stripes
Pitch
22
StartG
Timer
5
StopG
Label3
148
Flash
Stop Flash
Cue Left
Cue Reset
Cue Right
Colour 1 RGB
RC1
0
0
255
Colour 2 RGB
RC2
255
0
0
Pitch
13
Start
Timer
5
Stop
P
M
Start S
Stop S


Stripes
Pitch
0
Timer
0
StartG
StopG
Pitch
22
Timer
5
Start
Stop
Label3
245
Start S
P
M
Stop S
Flash
Stop Flash
Cue Left
Cue Reset
Cue Right
Colour 1 RGB
RC1
C1
Red
Green
Blue
Colour2 RGB
RC2
C2
Red
Green
Blue


Stripes
Pitch
0
StartG
Pitch
0
Start
Timer
0
StopG
Timer
0
Stop
Label3
245
Start S
P
M
Flash
Stop S
Stop Flash
Cue Left
Cue Reset
Cue Right
Colour 1 RGB
RC1
C1
0
216
255
Colour 2 RGB
RC2
C2
255
126
0

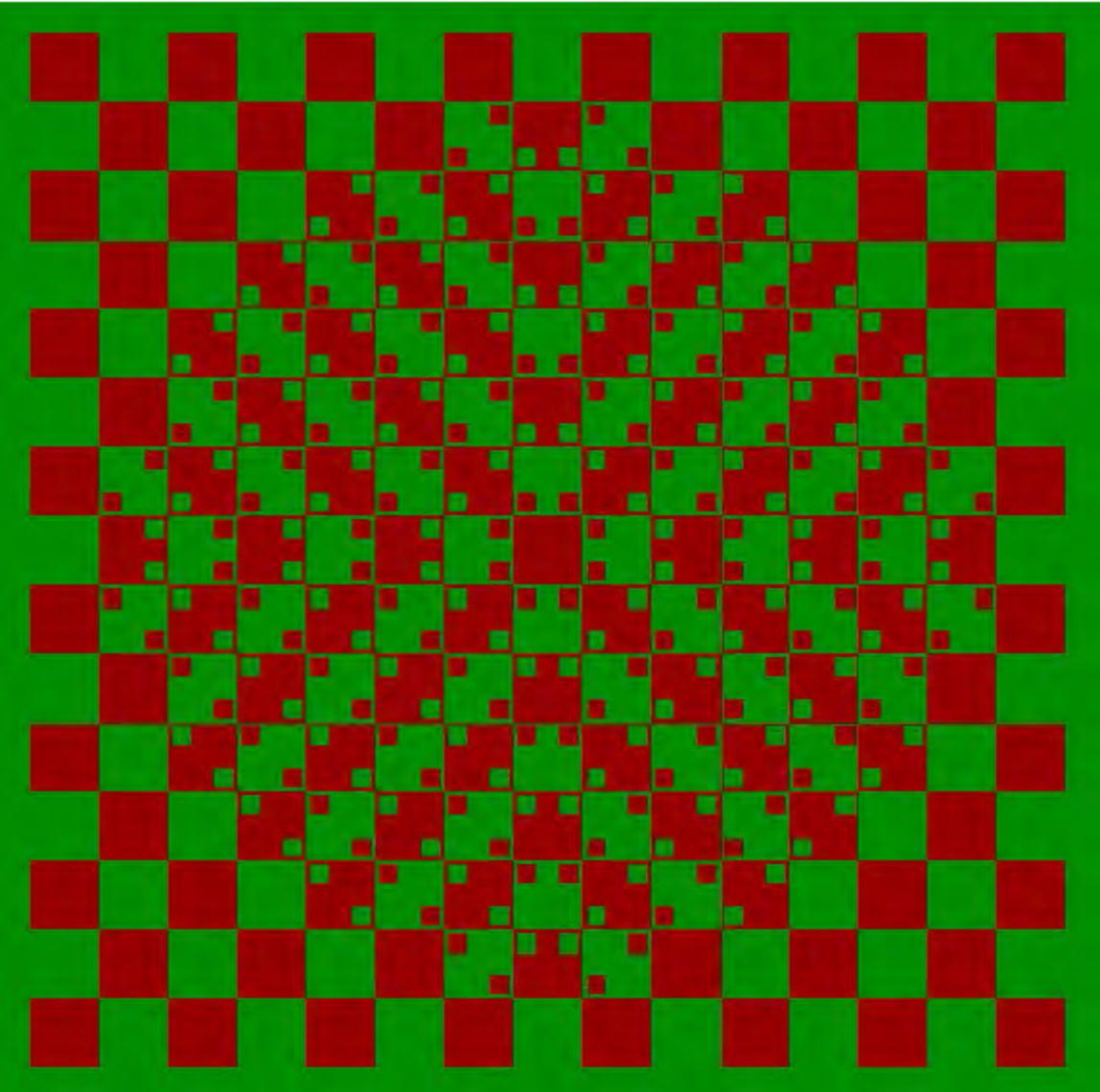


Image Processor

BackGround

50 | 120 | 50

150 | 150 | 0

Enabled

ForeGround

180 | 180 | 180

220 | 220 | 220

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

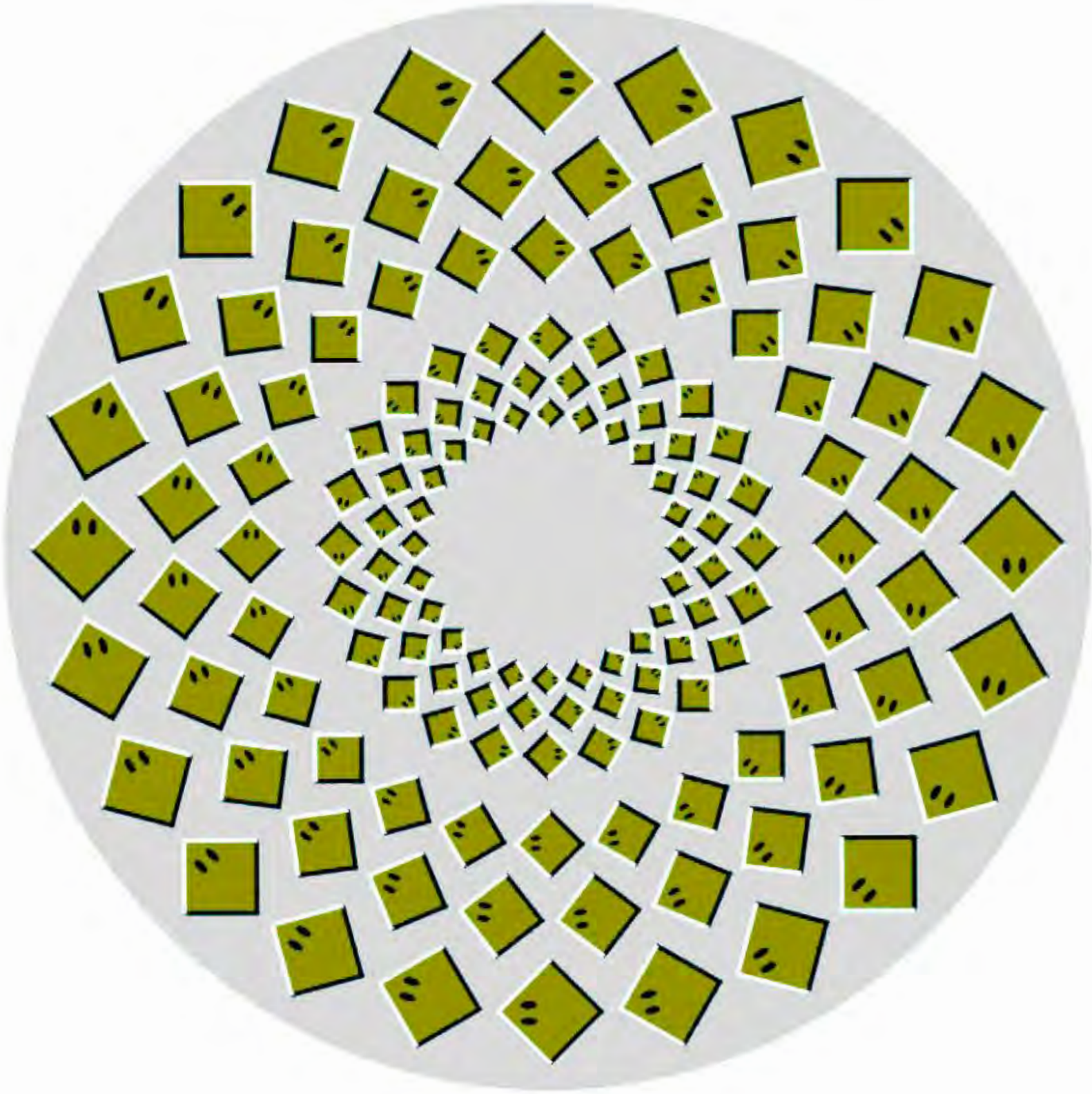


Image Processor

BackGround

0 | 201 | 254

150 | 150 | 150

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

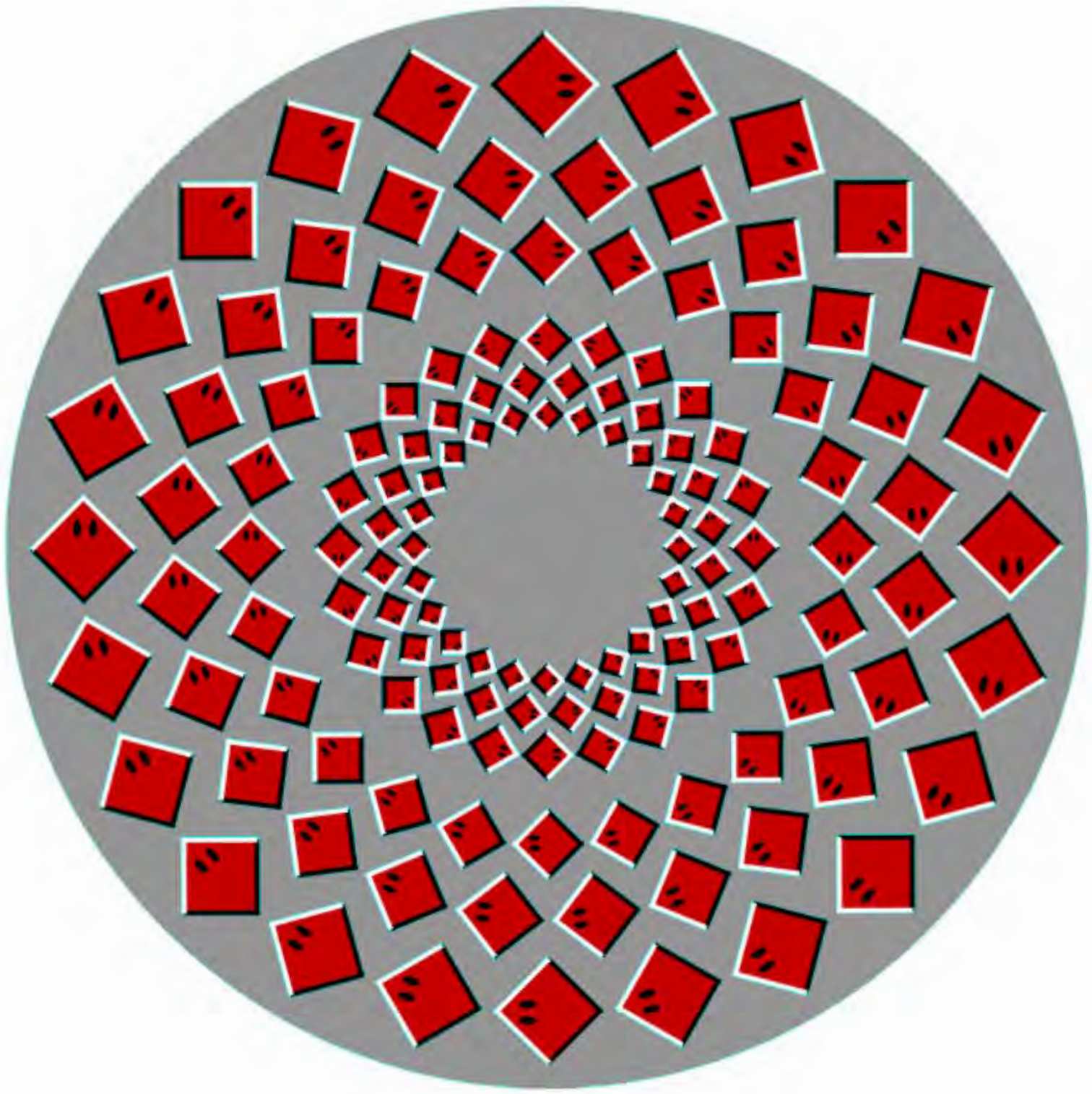
Rotate CW | Rotate CCW

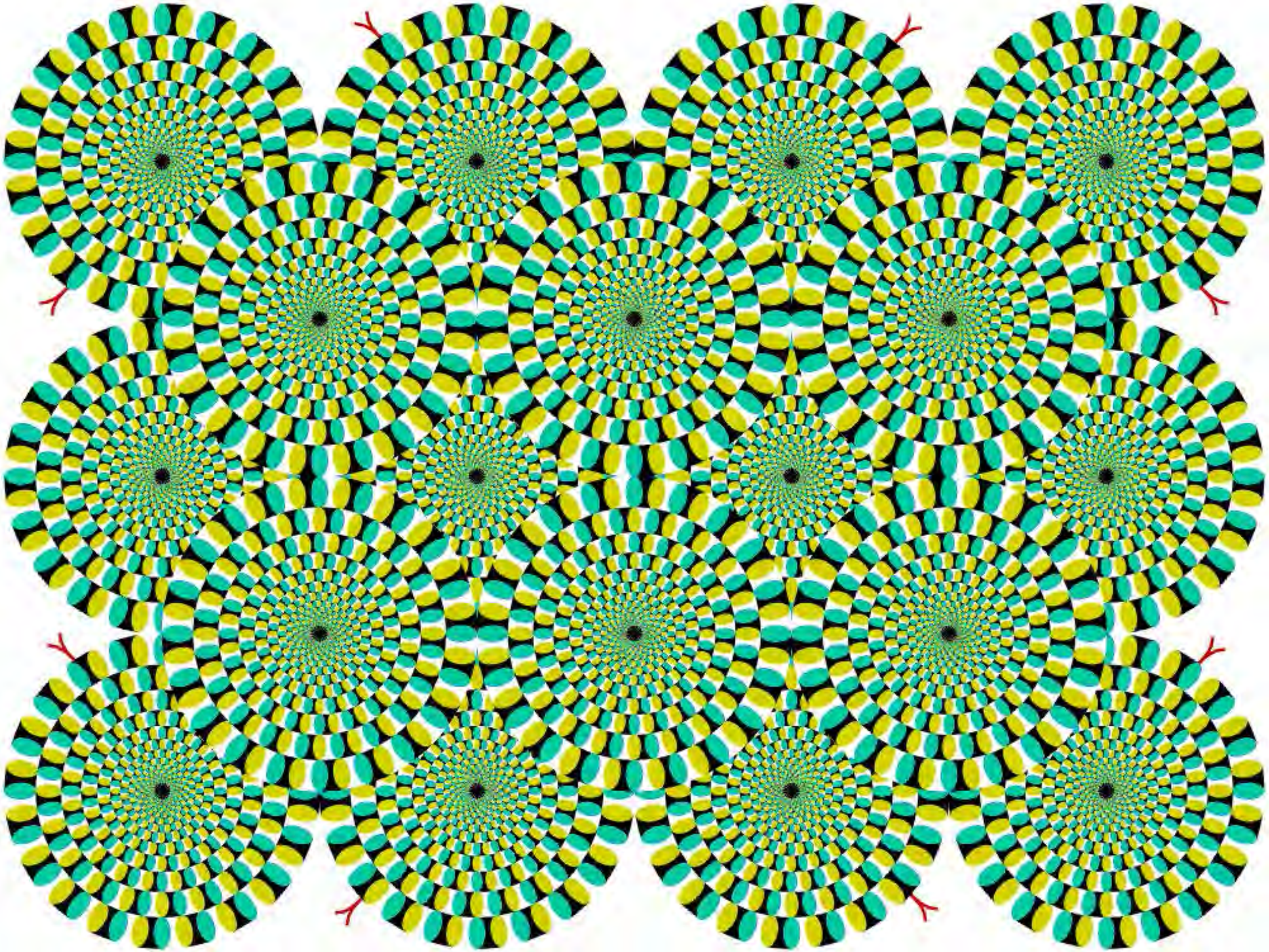
0°

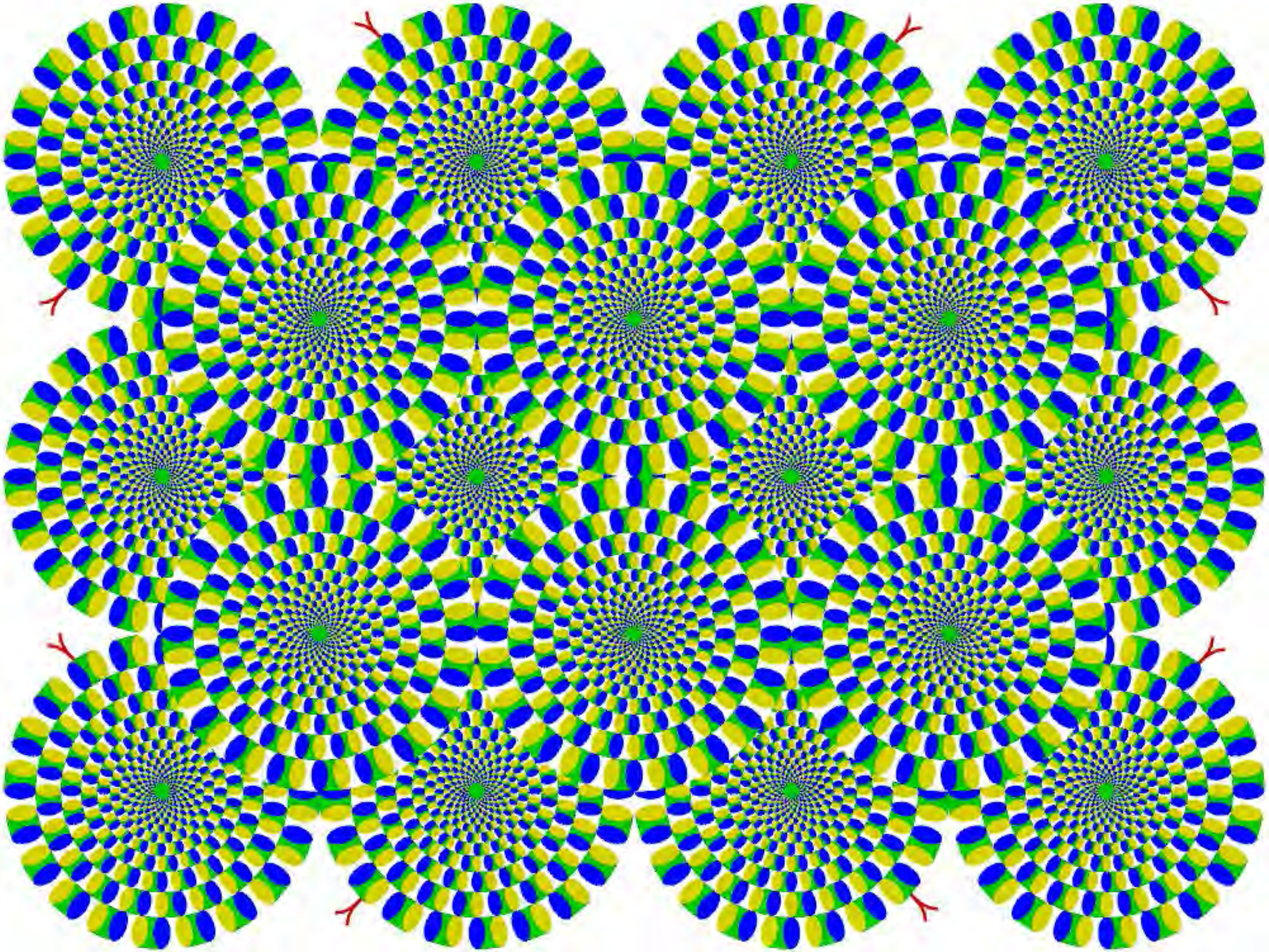
Save Image

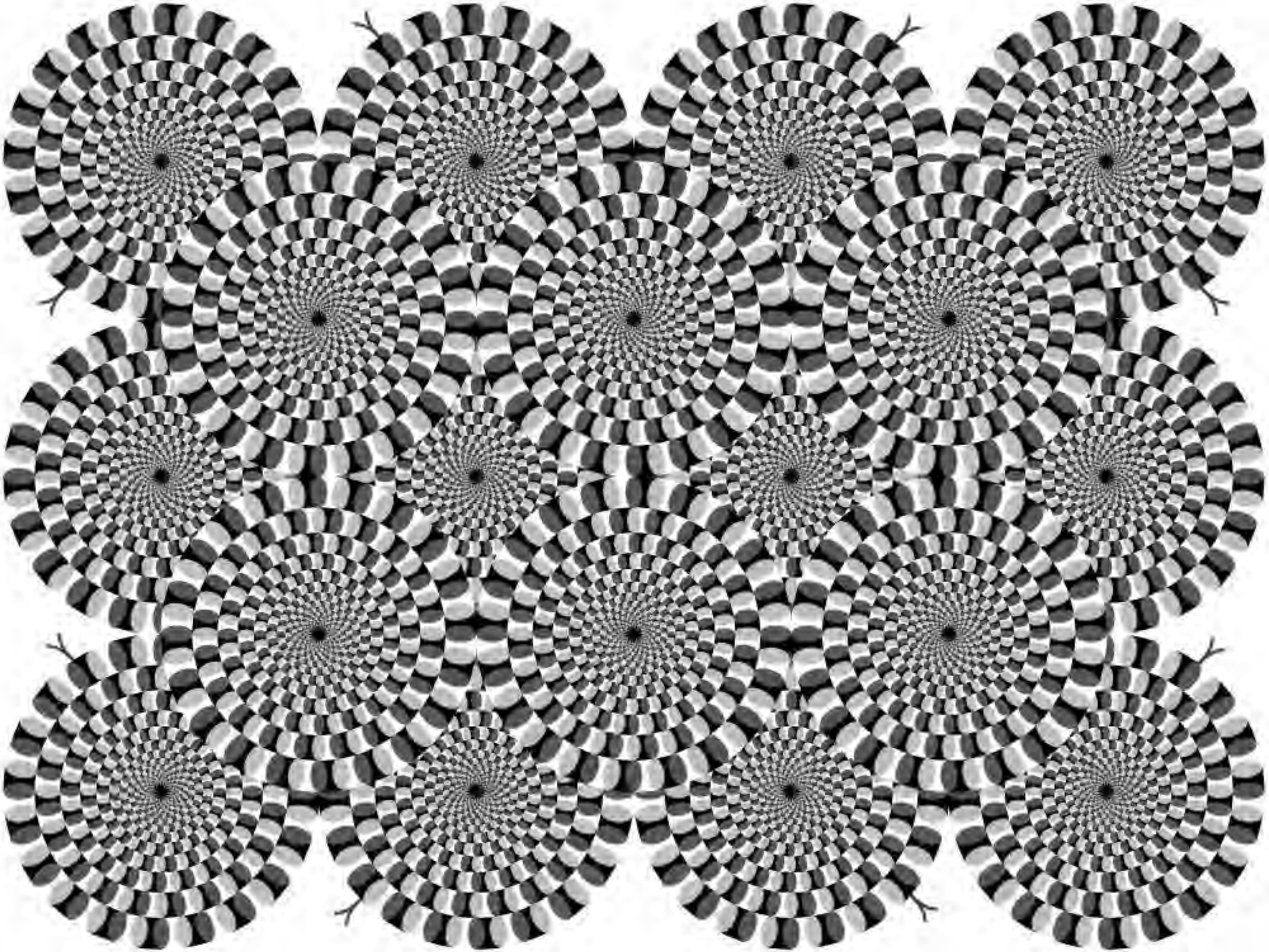
Load Image

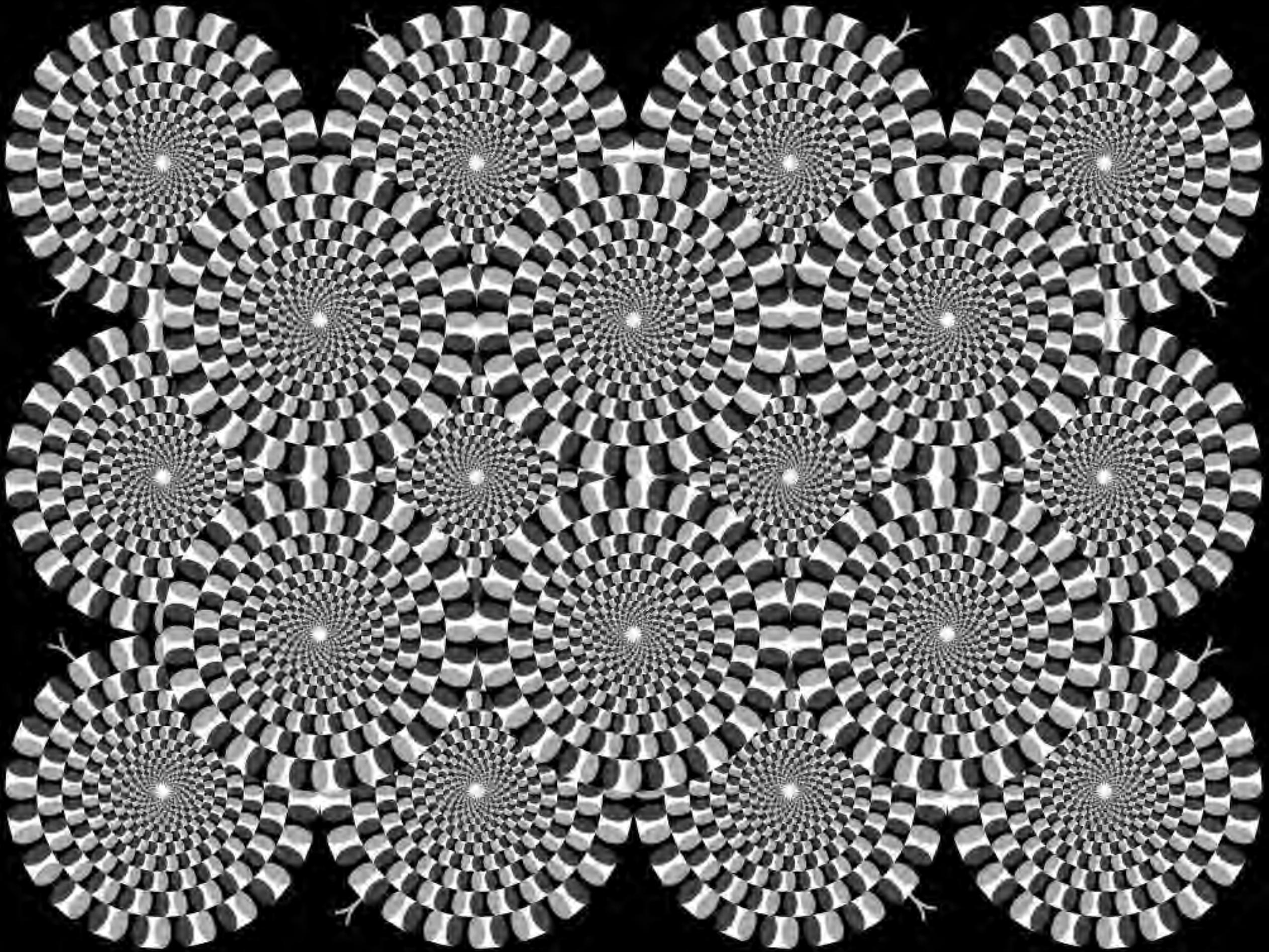
Reload Image











```
unit Confirm;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls;

type
  TUseConfirm = class(TForm)
    Memo1: TMemo;
    Button1: TButton;
    Label1: TLabel;
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  UseConfirm: TUseConfirm;

implementation
uses CPT;
{$R *.dfm}

procedure TUseConfirm.Button1Click(Sender: TObject);
begin
  CPT.Form1.BtnConfirm.Visible:=True;
  UseConfirm.Close;
end;

end.
```

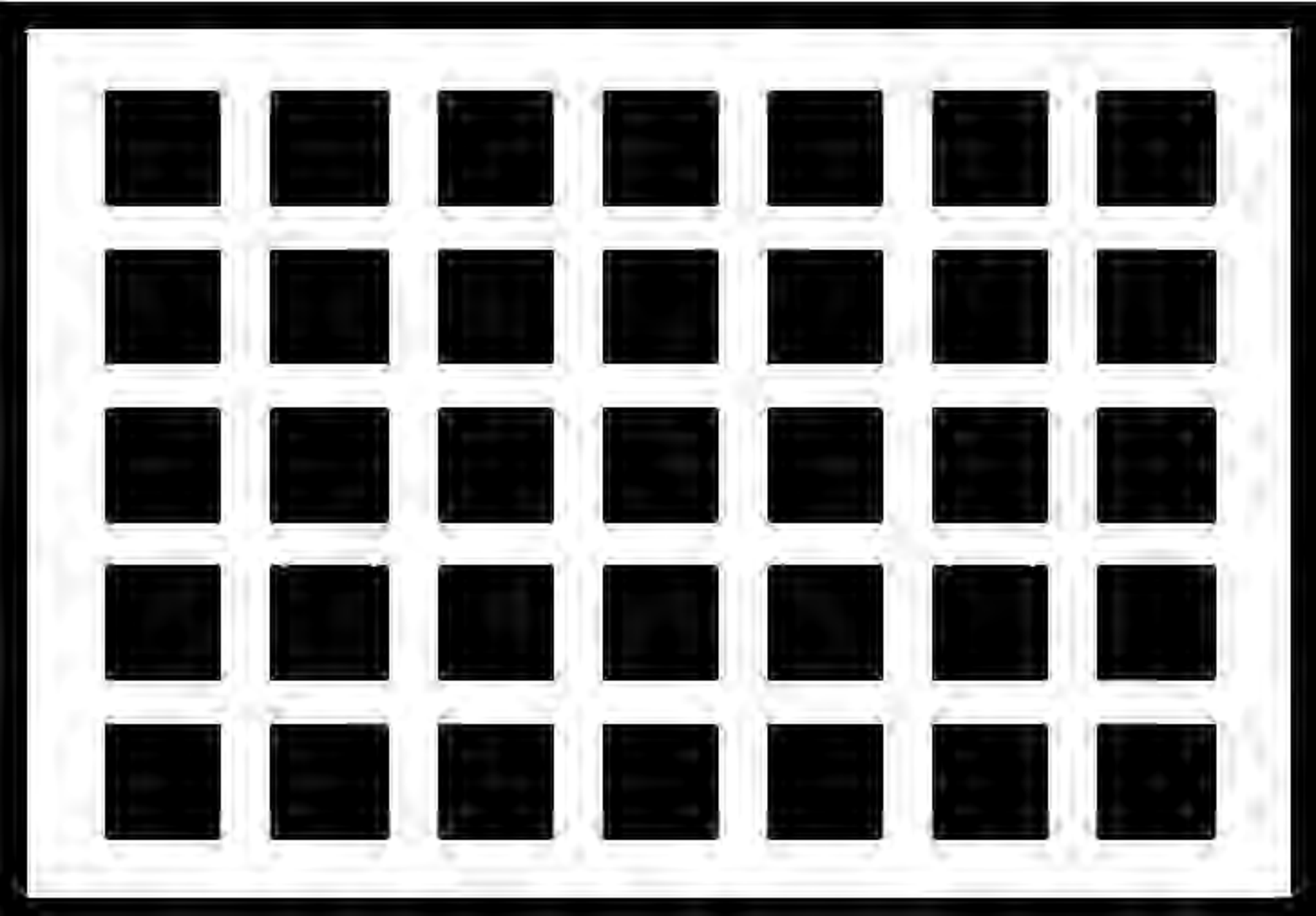


Image Processor

BackGround

50 120 50

0 254 0

Enabled

ForeGround

180 180 180

0 254 254

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

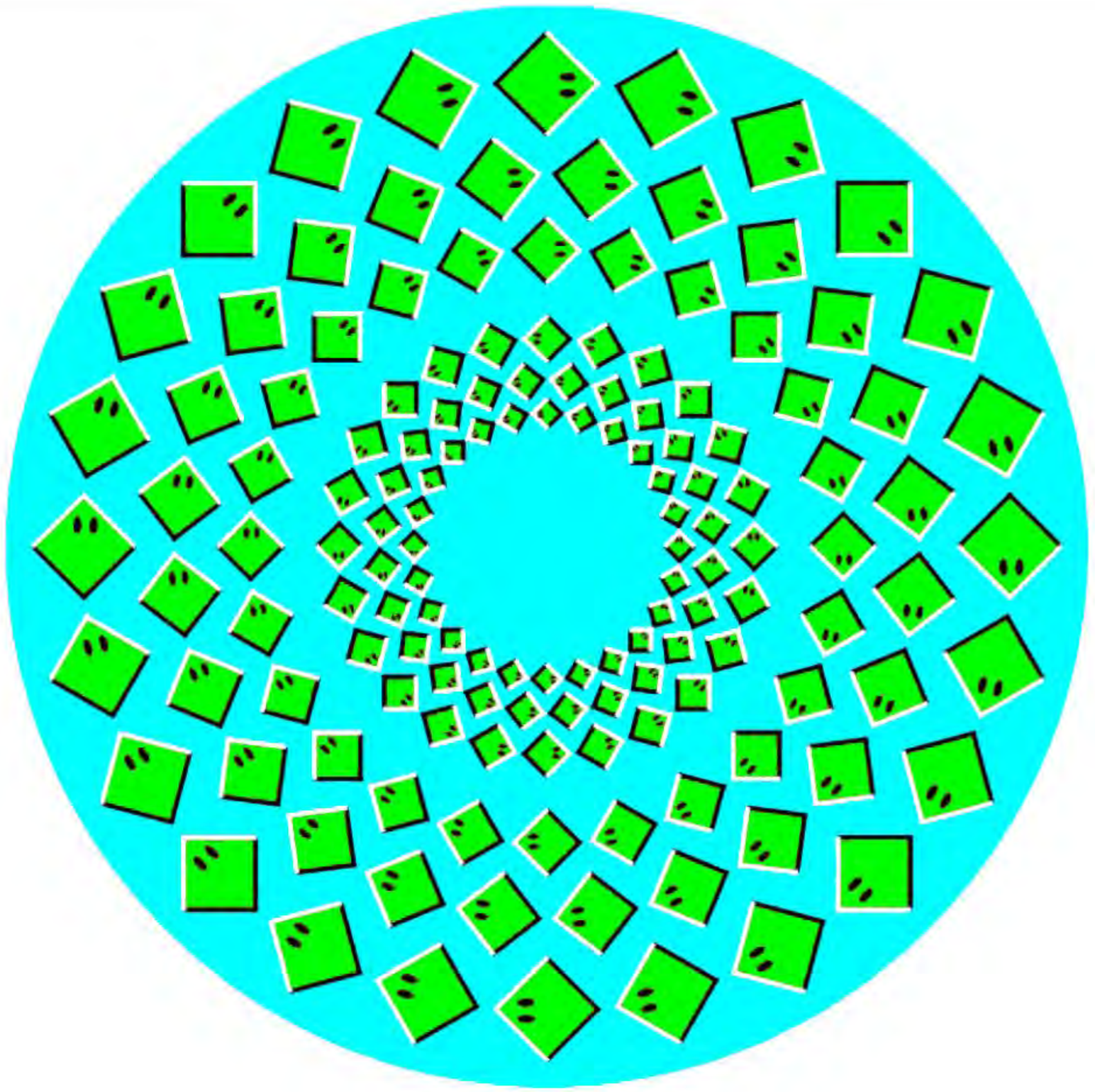


Image Processor

BackGround

50 | 120 | 50

0 | 200 | 200

Enabled

ForeGround

180 | 180 | 180

220 | 220 | 220

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

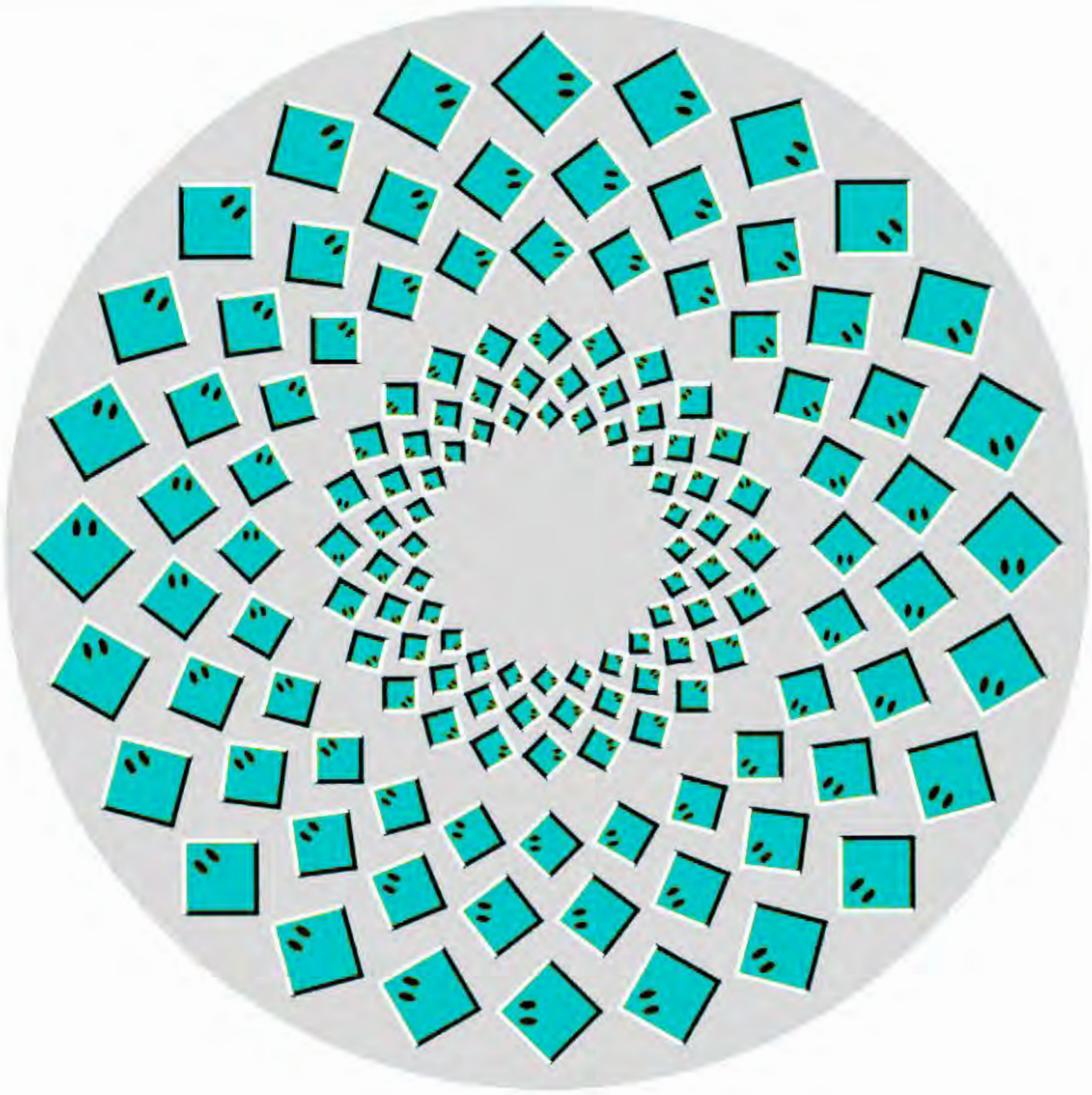


Image Processor

BackGround

50 120 50

0 255 0

Enabled

ForeGround

180 180 180

0 255 255

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

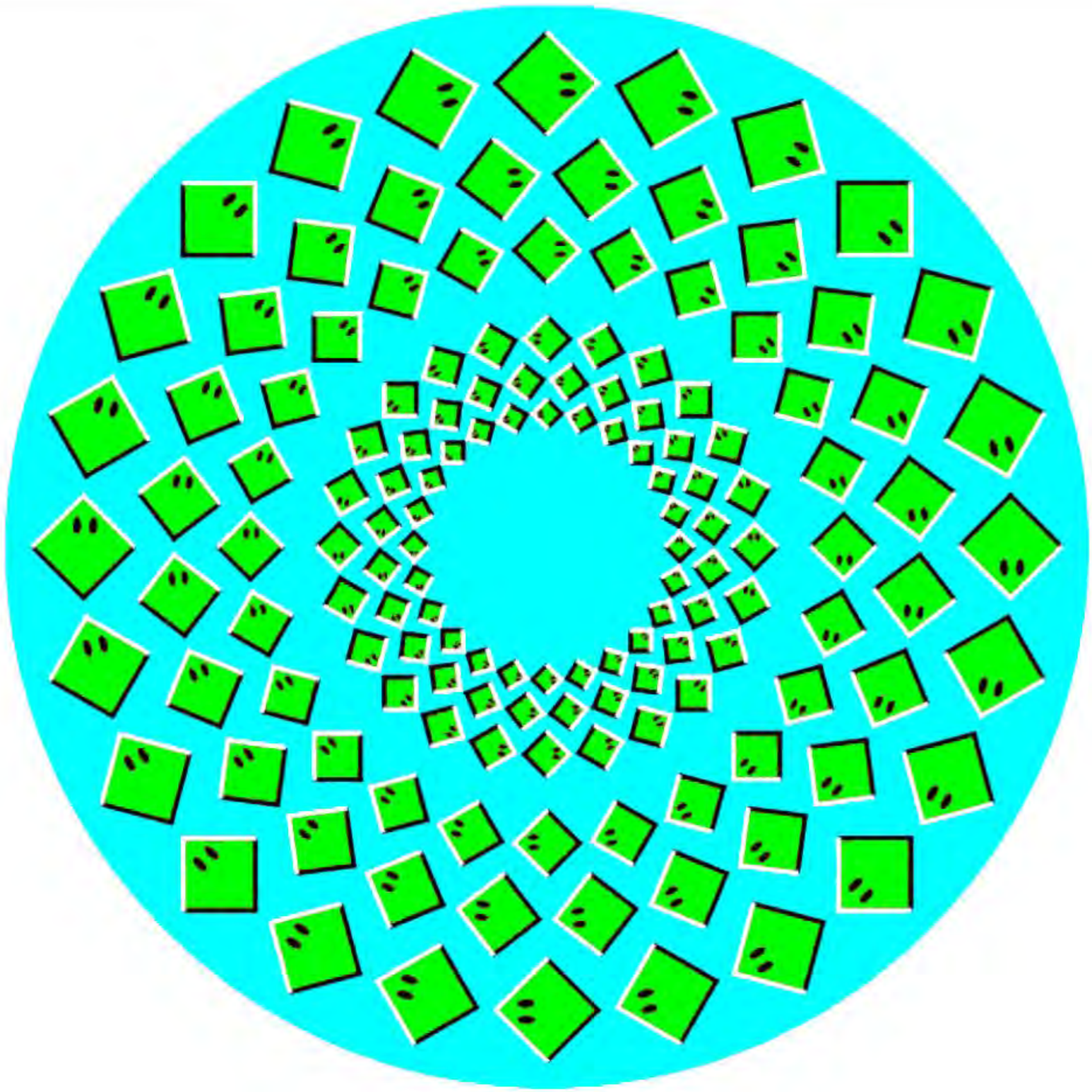


Image Processor

BackGround

50 120 50

0 220 0

Enabled

ForeGround

180 180 180

220 220 220

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

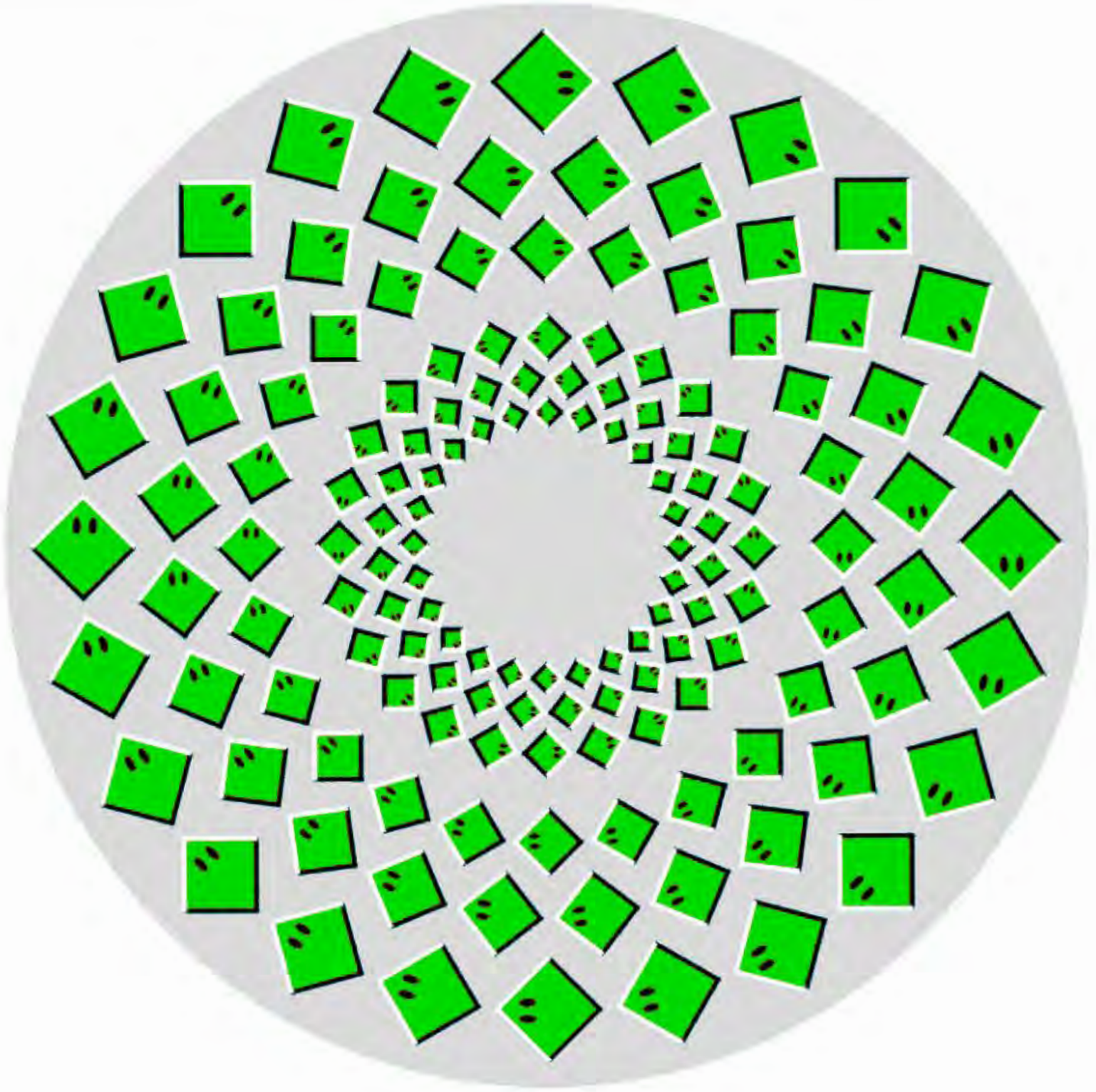


Image Processor

BackGround

0 | 201 | 254

250 | 250 | 150

Enabled

ForeGround

255 | 255 | 255

0 | 150 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

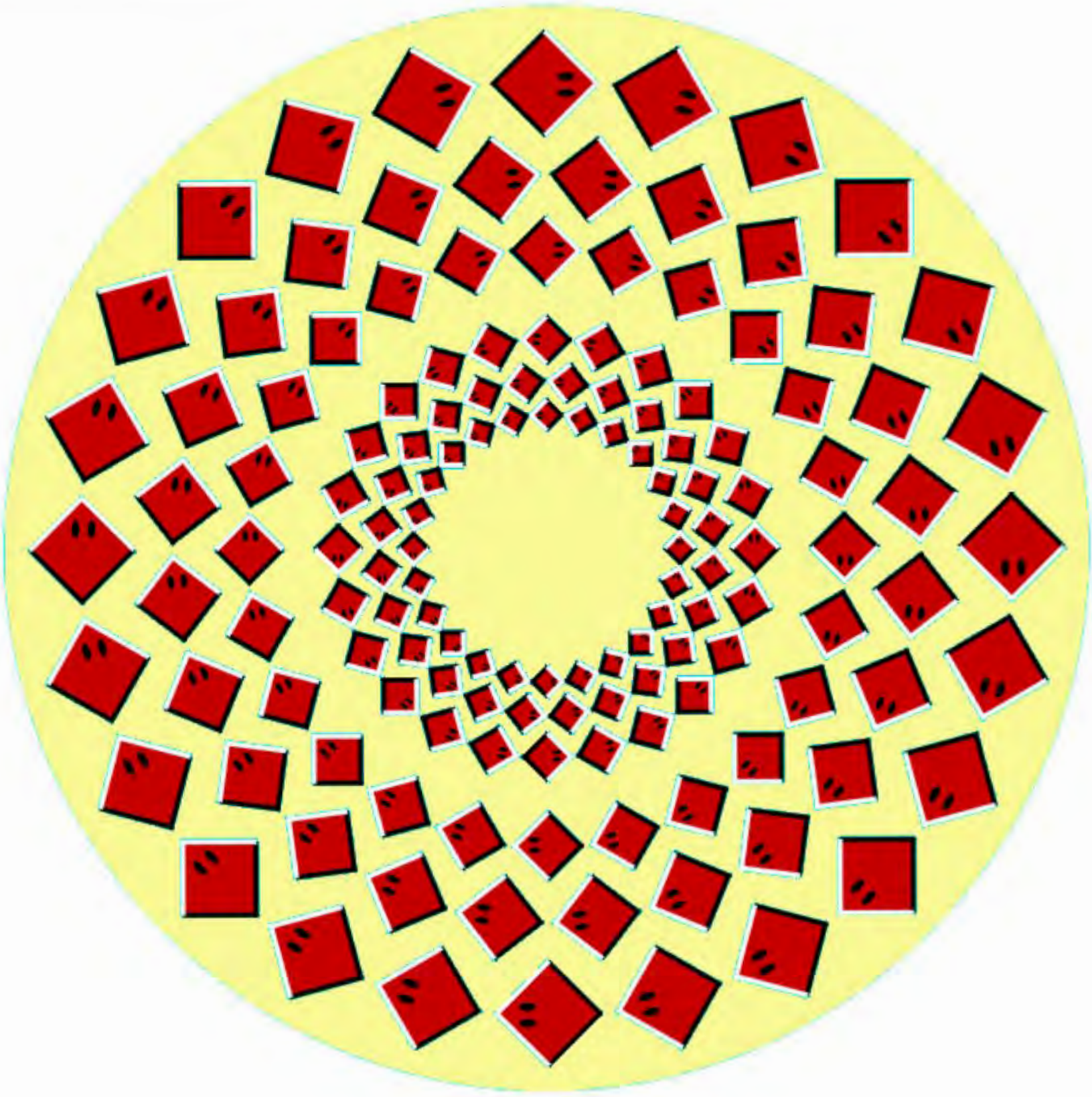


Image Processor

BackGround

50 120 50

0 170 0

Enabled

ForeGround

180 180 180

220 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

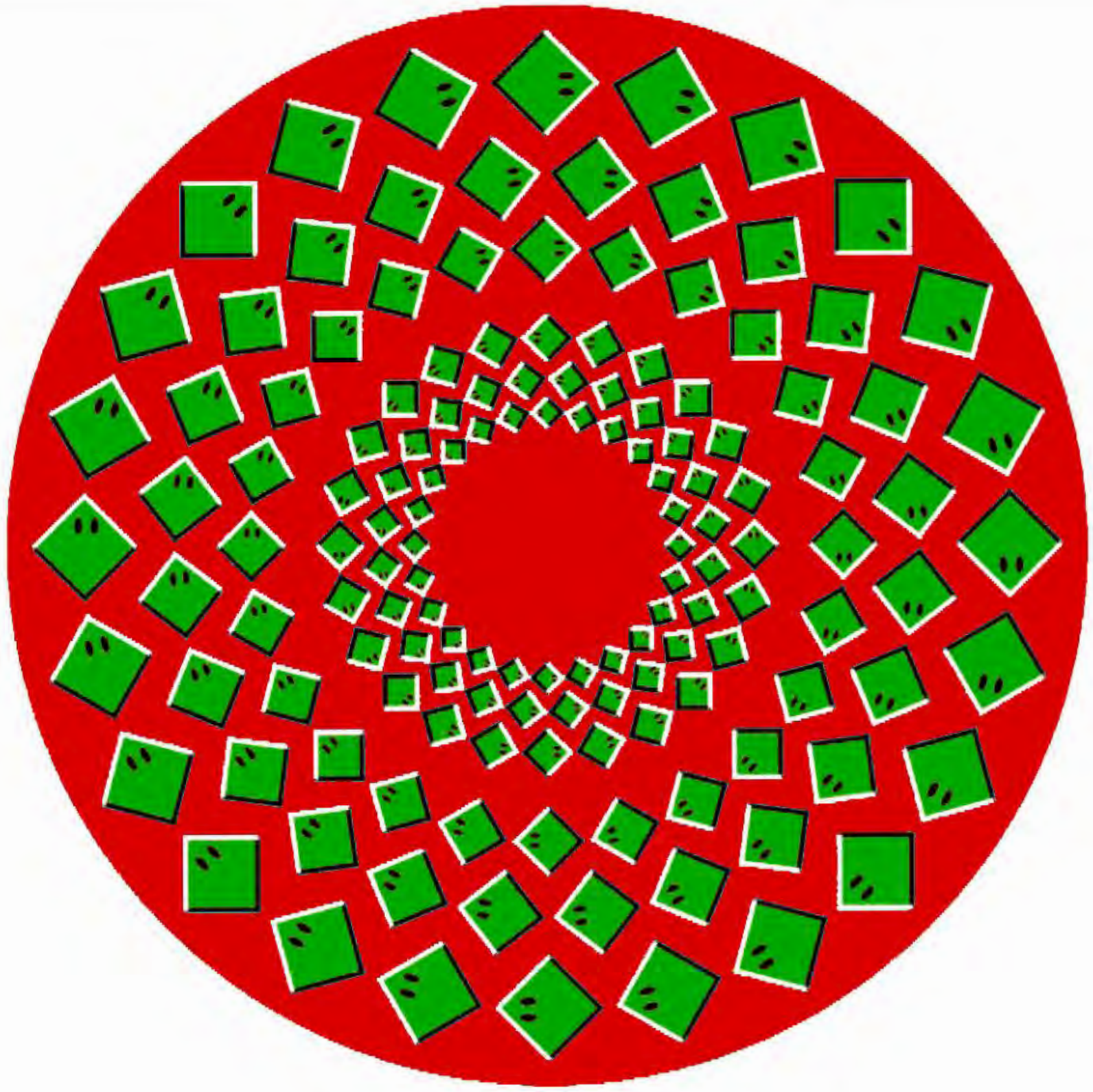


Image Processor

BackGround

50 120 50

200 200 0

Enabled

ForeGround

180 180 180

220 220 220

Enabled

Expander1

Apply

Reload and Apply

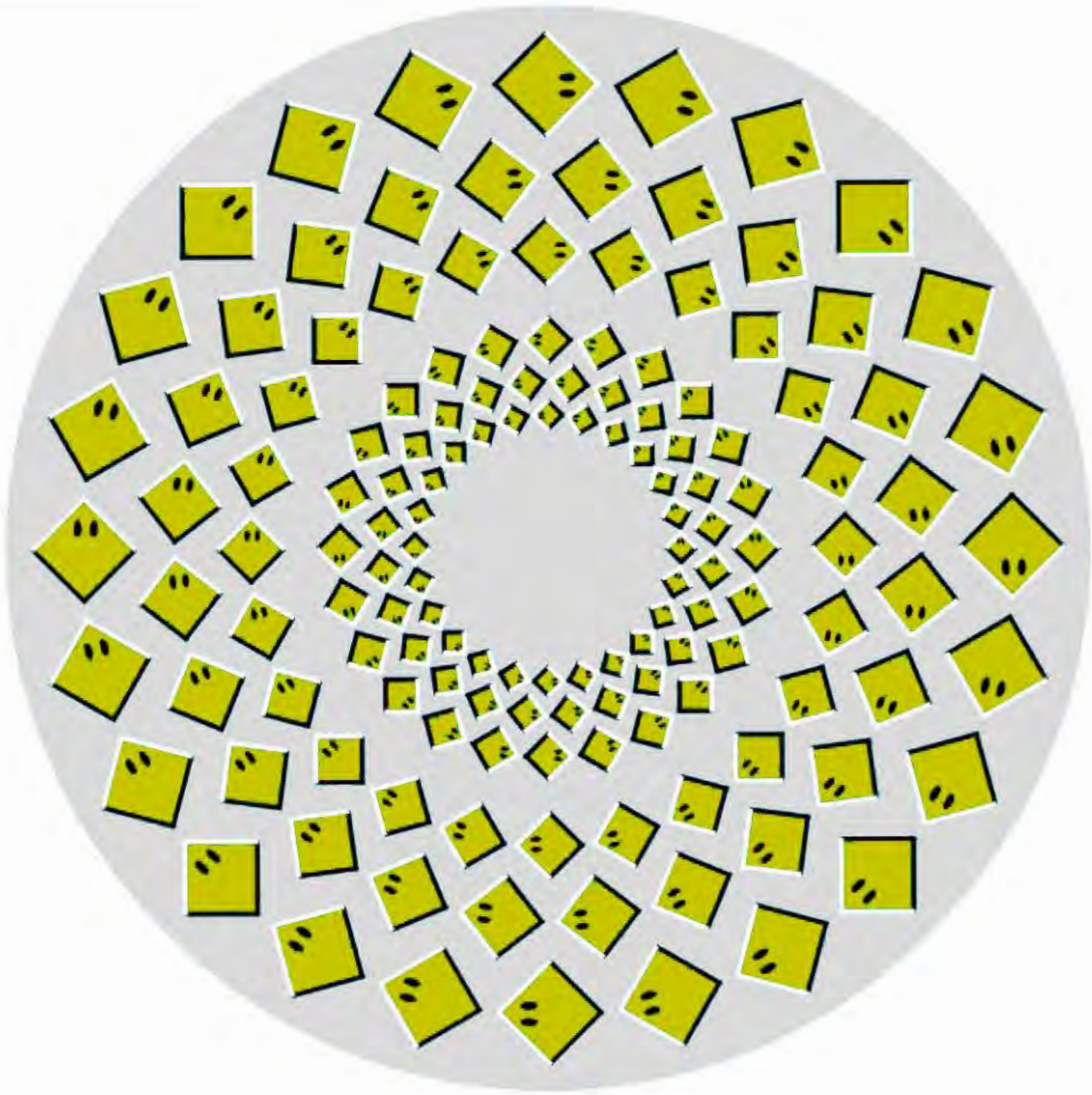
Rotate CW Rotate CCW

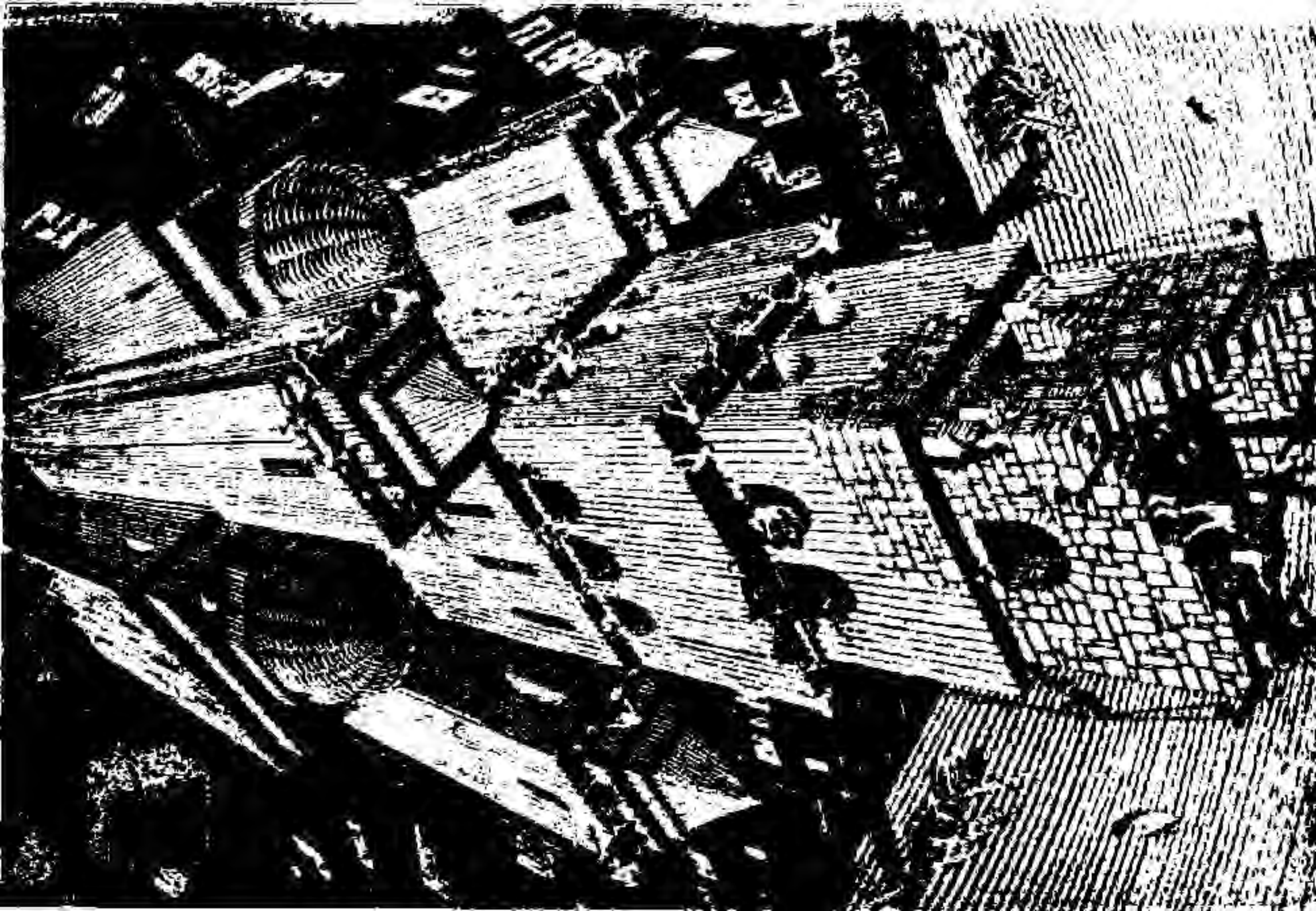
0°

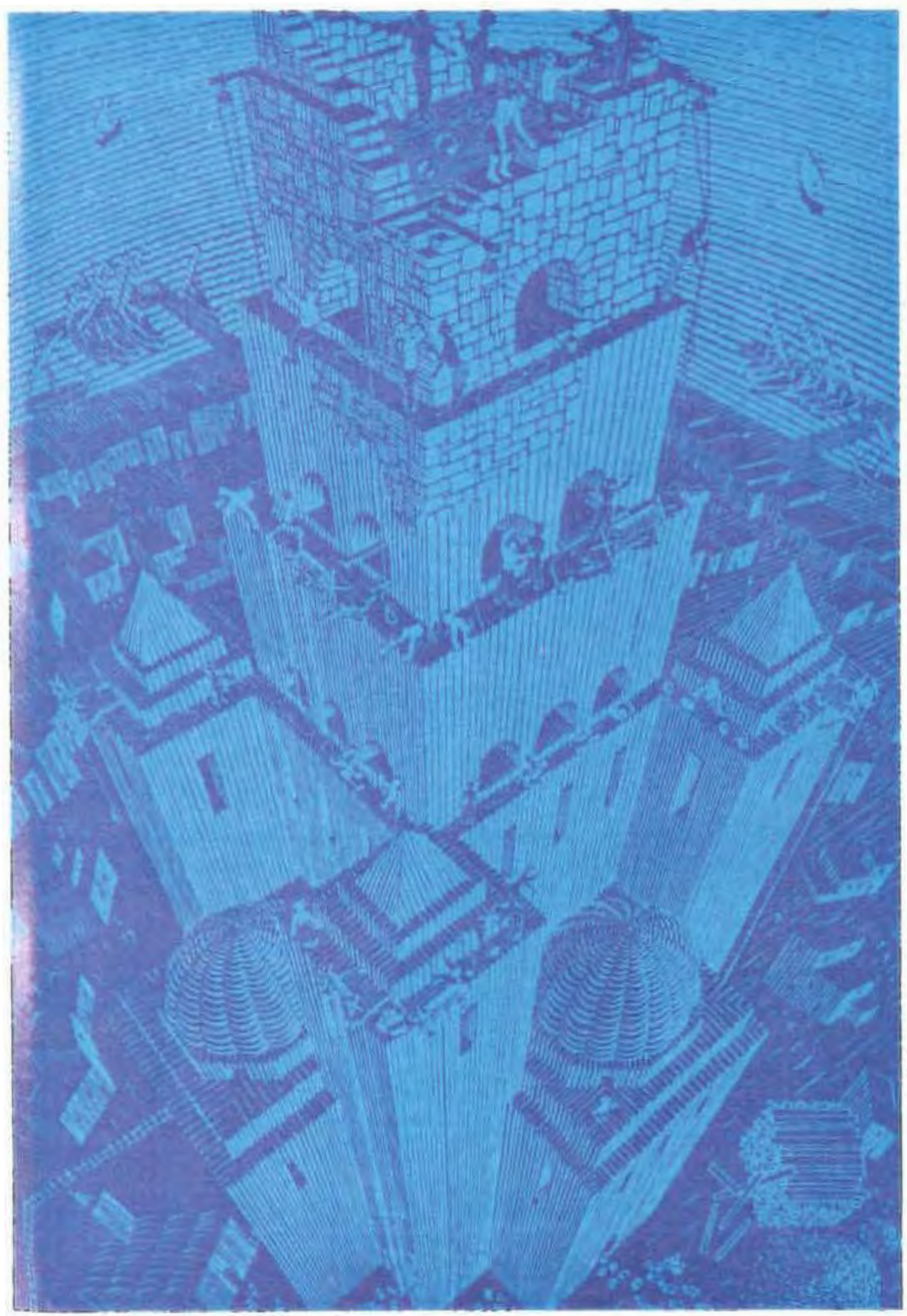
Save Image

Load Image

Reload Image







TOWER OF BABEL, by M. C. Escher, demonstrates the effect of luminance contrast on depth perception. Colors with different luminance levels, in this case blue and black (*left*), create the perception of a vividly three-dimensional image. But when the black is replaced (*right*) with a shade of green that has luminosity close to that of blue, the three-dimensional effect is lost and the image is hard to see. To convince yourself that both images are identical, look at them through a piece of blue glass or plastic.

```
unit login;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls,StrUtils, ADODB, DB;

type
    TLoginForm = class(TForm)
    passwordEdit: TEdit;
    pwdLabel: TLabel;
    LogInButton: TButton;
    Label1: TLabel;
    Button1: TButton;
    Memo1: TMemo;
    ADOCommand1: TADOCommand;
    ADOQuery1: TADOQuery;
    Lblresult: TLabel;
    ADONameTable: TADOTable;
    ComboBox1: TComboBox;
    procedure LogInButtonClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure ComboBox1Select(Sender: TObject);
    Procedure CreateTableParticipants;
  private
    { Private declarations }
  public
    class function Execute : boolean;
  end;

var
  LoginForm: TLoginForm;
  VNumber : String;
  Okay : Boolean;
  ConnectionList : TStrings;
  Var CString : WideString;
implementation

{$R *.dfm}

Procedure TLoginForm.CreateTableParticipants;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`participants` (';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`ID` varchar(9) NOT NULL,';
  SString:=SString+'`Gender` varchar(1) DEFAULT NULL,';
  SString:=SString+'`Age` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`QResponse` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Date` datetime NOT NULL,';
```

```
//  SString:=SString+'`Block` int(10) unsigned DEFAULT NULL,';
//  SString:=SString+'`Colour` varchar(1) DEFAULT NULL,';
//  SString:=SString+'`SubTimer1` varchar(4) DEFAULT NULL,';
//  SString:=SString+'`SubTimer2` varchar(4) DEFAULT NULL,';
//  SString:=SString+'`Counter` int(10) unsigned DEFAULT NULL,';
//  SString:=SString+'`Diff` int(10) unsigned DEFAULT NULL,';
//  SString:=SString+'`Item` varchar(15) DEFAULT NULL,';
//  SString:=SString+'`SubItem` varchar(15) DEFAULT NULL,';
//  SString:=SString+'`Flag` TINYINT(1) unsigned DEFAULT NULL,';
//  SString:=SString+'`BlockOrder` int(10) unsigned DEFAULT NULL,';
//  SString:=SString+'`First Language` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  ADOCommand1.CommandText:=SString;
  ADOCommand1.Execute;
End;

procedure TLoginForm.LogInButtonClick(Sender: TObject);
Var Command : WideString;
    EventDate : ShortString;
    Clicked : Boolean;
begin
//Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE1;Initial Catalog=uctexp1
  Okay:=False;
  Vnumber:=PasswordEdit.Text;
  if Length(VNumber)= 9 then
  Begin
    Clicked:=True;
    ADOQuery1.SQL.Text:=('Select * from participants');
    ADOQuery1.Active:=True;
    ADOQuery1.Locate('ID',VNumber,[]);
    if ADOQuery1.Fields.FieldByName('ID').Value=VNumber then
      Begin
        LblResult.Caption:='Number Exists';
        Okay:=False;
        ShowMessage('Number Exists - please select another ID');
      End
        Else
          Begin
            EventDate:=DateTimeToStr(Now);
            ADONameTable.Open;
            ADONameTable.Active:=True;
            With ADONameTable Do
              Begin
                Append;
                Fields.FieldByName('ID').Value:=VNumber;
                Fields.FieldByName('Date').AsDateTime:=StrToDateTime(EventDate);
                UpDateRecord;
                Post;
              End;
            LblResult.Caption:='Success!';
            Okay:=true;
            ShowMessage('Success! Please make a note of this number');
```

```
        End;
    if Okay then
    ModalResult := mrOK;
  End
    Else
      ShowMessage('Incorrect length - must be 9 characters or digits');
end;

procedure TLoginForm.Button1Click(Sender: TObject);
Var RR : Longint;
    RString : String;
begin
//  Randomize;
  Begin
    RR:=100000000+Random(99999999);
    RString:=IntToStr(RR);
    RString:=MidStr(RString,1,9);
    passwordEdit.Text:=RString;
  End;
end;

procedure TLoginForm.ComboBox1Select(Sender: TObject);
begin
  if ComboBox1.Text='Local Dell2'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT2 DELL;Mode=ReadWrite;Initial Catalog=iat2';
  if ComboBox1.Text='Mansfield'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT Mansfield;Mode=ReadWrite;Initial Catalog=iat';
  if ComboBox1.Text='Work PC2'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT2 WORK HP;Mode=ReadWrite;Initial Catalog=iat2';
  if ComboBox1.Text='UCT Server'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT CHOMSKY;Mode=ReadWrite;Initial Catalog=iat';
  if ComboBox1.Text='Windows 7 PC'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT W7HP;Mode=ReadWrite;Initial Catalog=iat';
  if ComboBox1.Text='W7 Siemens'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT DOUGLAS-PC;Mode=ReadWrite;Initial Catalog=iat';
  if ComboBox1.Text='MANSFIELD-PC'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT Mansfield-PC;Mode=ReadWrite;Initial Catalog=iat';
  ADOQuery1.ConnectionString:=CString;
  ADONameTable.ConnectionString:=CString;
  ADOCOmmand1.ConnectionString:=CString;
end;

class function TLoginForm.Execute: boolean;
begin
  with TLoginForm.Create(nil) do
  try
    Result := ShowModal = mrOk;
```

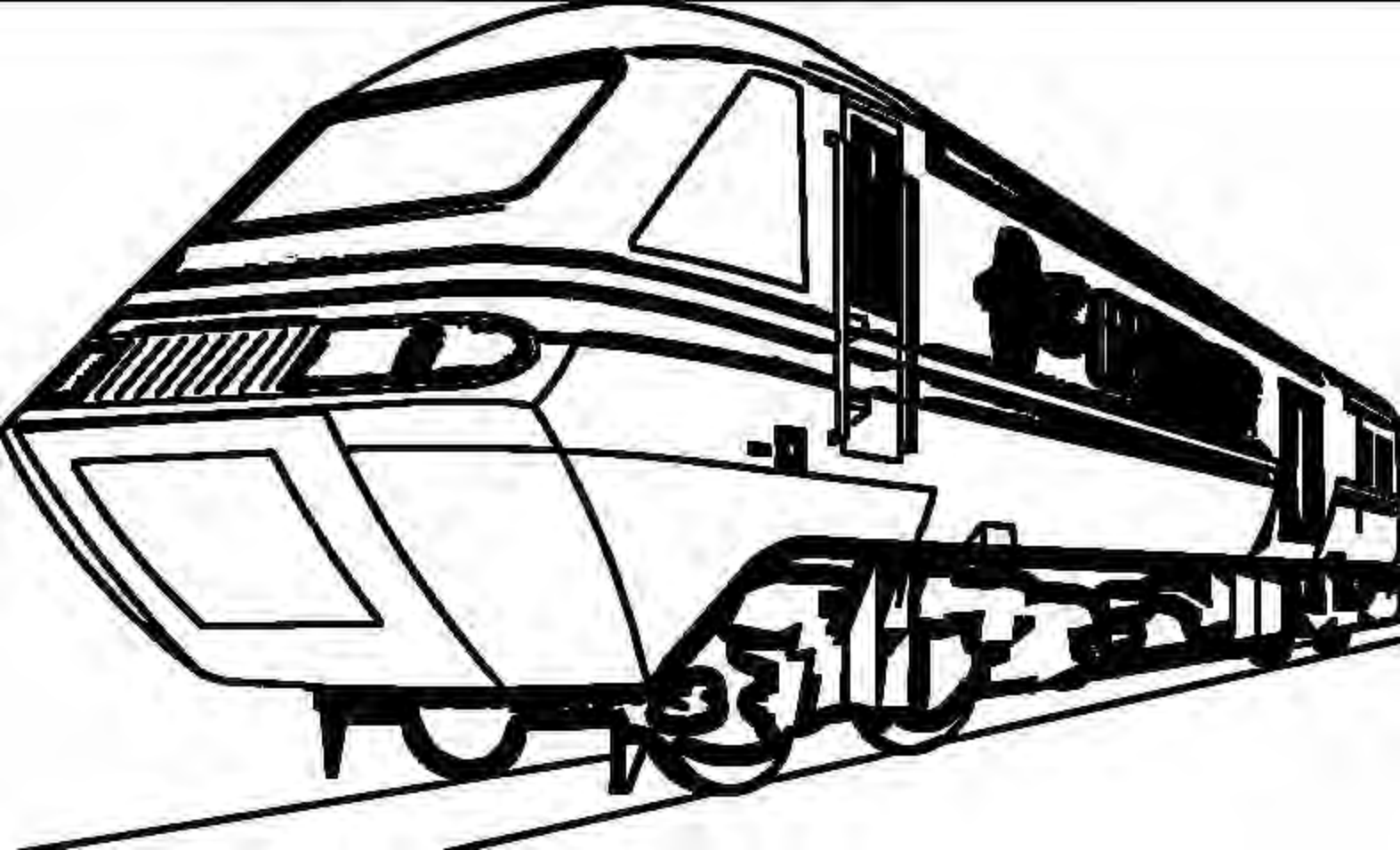
```
  finally
    Free;
  end;
end;

procedure TLoginForm.FormCreate(Sender: TObject);
begin
  passwordEdit.Text := '';
  ConnectionList:=TStringList.Create;
  ConnectionList.LoadFromFile('c:\IAT\Files\MSQLConnections.txt');
  Begin
    Try
      CString:=ConnectionList[1];
      ADOQuery1.ConnectionString:=CString;
      ADONameTable.ConnectionString:=CString;
      ADOCOmmand1.ConnectionString:=CString;
      CreateTableParticipants;
      ADOQuery1.SQL.Text:=('Select * from participants');
      ADOQuery1.Active:=True;
  Except
    ShowMessage('Unable to connect to database - please select');
    ComboBox1.Visible:=True;
  End;
  End;
end;

end.
```

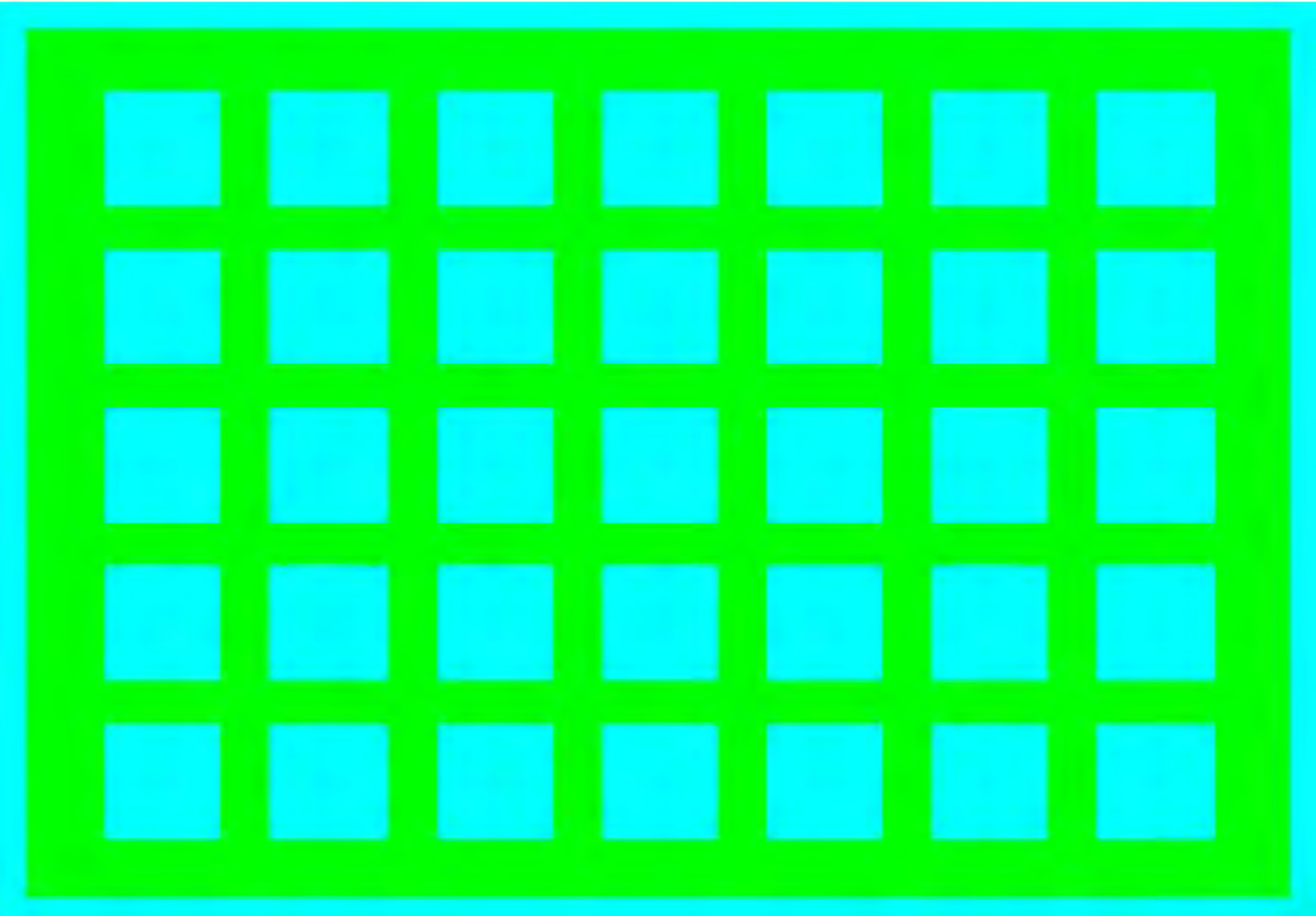
```
unit login;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls,StrUtils, ADODB, DB;

type
    TLoginForm = class(TForm)
    passwordEdit: TEdit;
    pwdLabel: TLabel;
    LogInButton: TButton;
    Label1: TLabel;
    Button1: TButton;
    Memo1: TMemo;
    ADOCommand1: TADOCommand;
    ADOQuery1: TADOQuery;
    Lblresult: TLabel;
    ADONameTable: TADOTable;
    ComboBox1: TComboBox;
    procedure LogInButtonClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure ComboBox1Select(Sender: TObject);
  private
    { Private declarations }
  public
    class function Execute : boolean;
  end;

var
  LoginForm: TLoginForm;
  VNumber : String;
  Okay : Boolean;
  ConnectionList : TStrings;
  Var CString : WideString;
implementation

{$R *.dfm}

Procedure CreateTableParticipants;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`participants` (';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`ID` varchar(9) NOT NULL,';
  SString:=SString+'`Gender` varchar(1) NOT NULL,';
  SString:=SString+'`Age` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Visual Rating` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Colour Blind` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Date` datetime NOT NULL,';
```

```
  SString:=SString+'`V1` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V2` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V3` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V4` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V5` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V6` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V7` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V8` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V9` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`First Language` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  LoginForm.ADOCommand1.CommandText:=SString;
  LoginForm.ADOCommand1.Execute;
End;

procedure TLoginForm.LogInButtonClick(Sender: TObject);
Var Command : WideString;
    EventDate : ShortString;
    Clicked : Boolean;
begin
//Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE1;Initial Catalog=uctexp1
  Okay:=False;
  Vnumber:=PasswordEdit.Text;
  if Length(VNumber)= 9 then
  Begin
    Clicked:=True;
    ADOQuery1.SQL.Text:=('Select * from participants');
    ADOQuery1.Active:=True;
    ADOQuery1.Locate('ID',VNumber,[]);
    if ADOQuery1.Fields.FieldByName('ID').Value=VNumber then
      Begin
        LblResult.Caption:='Number Exists';
        Okay:=False;
        ShowMessage('Number Exists - please select another ID');
      End
        Else
          Begin
            EventDate:=DateTimeToStr(Now);
            ADONameTable.Open;
            ADONameTable.Active:=True;
            With ADONameTable Do
              Begin
                Append;
                Fields.FieldByName('ID').Value:=VNumber;
                Fields.FieldByName('Date').AsDateTime:=StrToDateTime(EventDate);
                UpDateRecord;
                Post;
              End;
            LblResult.Caption:='Success!';
            Okay:=true;
            ShowMessage('Success! Please make a note of this number');
          End;
```

```
    if Okay then
    ModalResult := mrOK;
  End
    Else
      ShowMessage('Incorrect length - must be 9 characters or digits');
end;

procedure TLoginForm.Button1Click(Sender: TObject);
Var RR : Longint;
    RString : String;
begin
  Randomize;
  Begin
    RR:=100000000+Random(99999999);
    RString:=IntToStr(RR);
    RString:=MidStr(RString,1,9);
    passwordEdit.Text:=RString;
  End;
end;

procedure TLoginForm.ComboBox1Select(Sender: TObject);
begin
  if ComboBox1.Text='Local Vista'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE2;Initial Catalog=uctexp1';
  if ComboBox1.Text='Mansfield'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE1;Initial Catalog=uctexp1';
  if ComboBox1.Text='Work PC'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE3;Initial Catalog=uctexp1';
  if ComboBox1.Text='UCT Server'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE5;Initial Catalog=uctexp1';
  if ComboBox1.Text='Windows 7 PC'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=DOUGAS-HP;Initial Catalog=uctexp1';
  ADOQuery1.ConnectionString:=CString;
  ADOQuery1.ConnectionString:=CString;
  ADONameTable.ConnectionString:=CString;
  ADOCOmmand1.ConnectionString:=CString;
end;

class function TLoginForm.Execute: boolean;
begin
  with TLoginForm.Create(nil) do
  try
    Result := ShowModal = mrOk;
  finally
    Free;
  end;
end;

procedure TLoginForm.FormCreate(Sender: TObject);
```

```
//Var CString : WideString;
begin
  passwordEdit.Text := '';
  ConnectionList:=TStringList.Create;
  ConnectionList.LoadFromFile('c:\cpt\StFiles\MSQLConnections.txt');
  Begin
    Try
      CString:=ConnectionList[0];
      ADOQuery1.ConnectionString:=CString;
      ADOQuery1.SQL.Text:=('Select * from participants');
      ADOQuery1.Active:=True;
  Except
    ShowMessage('Unable to connect to database - please select');
    ComboBox1.Visible:=True;
  End;
  End;
end;

end.
```





```
unit CPT;
 // amended 06 January 2011 RAD 2010
 // dll for working out Zscores
 // dll for random function
 // dll for the item sorting function
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ExtCtrls, Buttons, TeeProcs, TeEngine, Chart, Series, jpeg,
  ComCtrls,Types,DateUtils,StrUtils,Graph,Confirm,AdvSmoothProgressBar, AdvSmoothButton,
  AdvSmoothGauge, AdvSmoothLabel, AdvSmoothJogWheel, DB, ADODB, ImgList,Math,
  AdvSmoothStatusIndicator, AdvSmoothPanel, Grids, DBGrids;
  Procedure StopClock(Var EndTime: Single); External
  'StopWatch.dll';
  Procedure StartClock; External
  'StopWatch.dll';

type
   TForm1 = class(TForm)
    AgeField: TEdit;
    Banner2: TLabel;
    BitBtn1: TBitBtn;
    BtnChoosePracticeOrTest: TButton;
    BtnChoose: TButton;
    LoadInstructions: TButton;
    BtnGetDetailsPractice: TButton;
    Field: TLabel;
    Gender: TLabel;
    Label1: TLabel;
    LblAge: TLabel;
    Line2: TLabel;
    NameField: TEdit;
    BtnBeginPractice: TButton;
    BtnStartTest: TButton;
    RBFemale: TRadioButton;
    RBMale: TRadioButton;
    BtnEndTestStartGraph: TButton;
    Timer1: TTimer;
    Timer2: TTimer;
    SaveDialog1: TSaveDialog;
    Time2: TLabel;
    Dprime1: TLabel;
    Dprime2: TLabel;
    Timer3: TTimer;
    BtnShowBarGraph: TButton;
    Timer4: TTimer;
    ProgressBar1: TAdvSmoothProgressBar;
    Target: TImage;
    Timer5: TTimer;
    Banner1: TAdvSmoothLabel;
    Fixation: TTimer;
    Sizer: TTimer;
```

FReactions: TFlowPanel;
V1: TRadioButton;
V2: TRadioButton;
V3: TRadioButton;
V4: TRadioButton;
V5: TRadioButton;
V6: TRadioButton;
LblReactions: TLabel;
V7: TRadioButton;
ADOConnection1: TADOConnection;
ADONameTable1: TADOTable;
BtnColour: TButton;
ImageColour: TImage;
ColourMem: TMemo;
BtnConsent: TButton;
BtnDecline: TButton;
BtnStartColour: TButton;
EdColour: TEdit;
LblInstruct: TLabel;
BtnVTest: TButton;
LblVresult: TLabel;
btnCalibrateP: TButton;
ADOCommand1: TADOCommand;
ADOSummaryTable: TADOTable;
ADOPTable1: TADOTable;
Stimulus: TLabel;
First: TTimer;
Second: TTimer;
RunP: TTimer;
Cycle: TTimer;
ScrollBar1: TScrollBar;
MatrixPanel: TAdvSmoothPanel;
T20S1: TAdvSmoothStatusIndicator;
T20S2: TAdvSmoothStatusIndicator;
T20S3: TAdvSmoothStatusIndicator;
T20S4: TAdvSmoothStatusIndicator;
T20S5: TAdvSmoothStatusIndicator;
T18S1: TAdvSmoothStatusIndicator;
T18S2: TAdvSmoothStatusIndicator;
T18S3: TAdvSmoothStatusIndicator;
T18S4: TAdvSmoothStatusIndicator;
T18S5: TAdvSmoothStatusIndicator;
T16S1: TAdvSmoothStatusIndicator;
T16S2: TAdvSmoothStatusIndicator;
T16S3: TAdvSmoothStatusIndicator;
T16S4: TAdvSmoothStatusIndicator;
T16S5: TAdvSmoothStatusIndicator;
T14S1: TAdvSmoothStatusIndicator;
T14S2: TAdvSmoothStatusIndicator;
T14S3: TAdvSmoothStatusIndicator;
T14S4: TAdvSmoothStatusIndicator;
T14S5: TAdvSmoothStatusIndicator;
T12S1: TAdvSmoothStatusIndicator;
T12S2: TAdvSmoothStatusIndicator;

T12S3: TAdvSmoothStatusIndicator;
T12S4: TAdvSmoothStatusIndicator;
T12S5: TAdvSmoothStatusIndicator;
LblFlickerHz: TLabel;
S1: TLabel;
S2: TLabel;
S3: TLabel;
S4: TLabel;
S5: TLabel;
ADOPCalTable: TADOTable;
ADOPQuery: TADOQuery;
ADOPQueryR1: TFloatField;
ADOPQueryG1: TFloatField;
ADOPQueryB1: TFloatField;
ADOPQueryR2: TFloatField;
ADOPQueryG2: TFloatField;
ADOPQueryB2: TFloatField;
ADOSummaryParvo: TADOTable;
LblLang: TLabel;
FLanguage: TFlowPanel;
LangIA: TRadioButton;
LangAfrikaans: TRadioButton;
LangEng: TRadioButton;
LangEurope: TRadioButton;
LangME: TRadioButton;
LangHisp: TRadioButton;
LangEast: TRadioButton;
ADOMTable1: TADOTable;
ADOSummaryMain: TADOTable;
RPanel: TAdvSmoothPanel;
CaptionTargets: TAdvSmoothStatusIndicator;
CaptionError: TAdvSmoothStatusIndicator;
CaptionScore: TAdvSmoothStatusIndicator;
RTime: TAdvSmoothStatusIndicator;
ETime: TAdvSmoothStatusIndicator;
ValidNumber: TLabel;
ErrorNumber: TLabel;
KeyCounter: TLabel;
Latency: TLabel;
ELatency: TLabel;
PanelCValues: TAdvSmoothPanel;
Label3: TLabel;
Label5: TLabel;
Label7: TLabel;
Label9: TLabel;
Label11: TLabel;
LblR: TLabel;
LblG: TLabel;
LblB: TLabel;
LblB1: TLabel;
LblG1: TLabel;
LblR1: TLabel;
ValidNumberP: TLabel;
ErrorNumberP: TLabel;

```
KeyCounterP: TLabel;
CaptionNonTargets: TAdvSmoothStatusIndicator;
NValidP: TLabel;
NValidM: TLabel;
NulTargets: TAdvSmoothStatusIndicator;
NullTM: TLabel;
NullTP: TLabel;
AdvSmoothStatusIndicator1: TAdvSmoothStatusIndicator;
NullErrorsP: TLabel;
NullErrorsM: TLabel;
Memo1: TRichEdit;
ImageList1: TImageList;
BtnTestInstructions: TButton;
BtnPInstructions: TButton;
BtnContinue: TButton;
ADOSettingsQuery: TADOQuery;
ChkS1: TCheckBox;
ChkS2: TCheckBox;
ChkS3: TCheckBox;
ChkS4: TCheckBox;
ChkS5: TCheckBox;
BtnConfirm: TButton;
CFirst: TTimer;
CSecond: TTimer;
ChkT1: TCheckBox;
ChkT2: TCheckBox;
ChkT3: TCheckBox;
ChkT4: TCheckBox;
ChkT5: TCheckBox;
Jog: TAdvSmoothJogWheel;
BtnDisable: TButton;
Correct: TAdvSmoothButton;
Incorrect: TAdvSmoothButton;
FForm1: TAdvSmoothLabel;

procedure Timer3Timer(Sender: TObject);
procedure Timer2Timer(Sender: TObject);
procedure Timer1Timer(Sender: TObject);
procedure BtnGetDetailsPracticeClick(Sender: TObject);
procedure LoadInstructionsClick(Sender: TObject);
procedure StartAllTesting(Sender: TObject);
procedure BtnShowBarGraphClick(Sender: TObject);
procedure ChoosePracticeOrTest(Sender: TObject);
procedure BtnBeginPracticeClick(Sender: TObject);
procedure BtnStartTestClick(Sender: TObject);
procedure BtnEndTestStartGraphClick(Sender: TObject);
procedure FormKeyPress(Sender: TObject; var Key: Char);
procedure Timer4Timer(Sender: TObject);
procedure Timer5Timer(Sender: TObject);
procedure FixationTimer(Sender: TObject);
procedure SizerTimer(Sender: TObject);
procedure BtnColourClick(Sender: TObject);
procedure BtnConsentClick(Sender: TObject);
procedure BtnStartColourClick(Sender: TObject);
```

```
  procedure BtnVTestClick(Sender: TObject);
  procedure EdColourChange(Sender: TObject);
  procedure BtnDeclineClick(Sender: TObject);
  procedure btnCalibratePClick(Sender: TObject);
  procedure RunPTimer(Sender: TObject);
  procedure FirstTimer(Sender: TObject);
  procedure SecondTimer(Sender: TObject);
  Procedure ColourCalibrate;
  Procedure RunPTimes;
  procedure CycleTimer(Sender: TObject);
  procedure ScrollBar1Scroll(Sender: TObject; ScrollCode: TScrollCode;
    var ScrollPos: Integer);
  Procedure ScoreCP;
  Procedure UpdateMatrix;
  Function Position(Button : TAdvSmoothStatusIndicator; Width,Left,First: Integer): Integer;
  Procedure CalcPAverages;
  procedure Button1Click(Sender: TObject);
  Procedure ProcessParvoPics;
  Procedure UpdateCursor;
  Procedure InitialiseMatrix;
  Procedure DisplayCM;
  Procedure DisplayM(VAR Letter,Last: String);
  Procedure ScoreM;
  Procedure ScoreCM;
  Procedure ScoreMStages;
  Procedure WritePreviousStage;
  Procedure ContrastInstructions;
  procedure BtnPInstructionsClick(Sender: TObject);
  procedure BtnTestInstructionsClick(Sender: TObject);
  procedure BtnContinueClick(Sender: TObject);
  Procedure ClockStop;
  Procedure ConfirmColour;
  procedure BtnConfirmClick(Sender: TObject);
  procedure ChkT1Click(Sender: TObject);
  procedure ChkT2Click(Sender: TObject);
  procedure ChkT3Click(Sender: TObject);
  procedure ChkT4Click(Sender: TObject);
  procedure ChkT5Click(Sender: TObject);
  procedure CFirstTimer(Sender: TObject);
  procedure ChkS1Click(Sender: TObject);
  procedure ChkS2Click(Sender: TObject);
  procedure ChkS3Click(Sender: TObject);
  procedure ChkS4Click(Sender: TObject);
  procedure ChkS5Click(Sender: TObject);
  procedure CSecondTimer(Sender: TObject);
  procedure JogValueChanged(Sender: TObject; Value: Double;
    CurrentMode: TAdvSmoothJogWheelModeType);
  procedure BtnDisableClick(Sender: TObject);
 private
  { Private declarations }
public

  { Public declarations }
end;
```

```
Type
  Results1 = Record
  TimeT : String[10];
  FirstWordT: String[10];
  SecondWordT: String[10];
  M1Latency: Integer;
  M1Elatency : Integer;
  FirstWordF: String[10];
  SecondWordF: String[10];
  TimeE : String[10];
  Value: Integer;
  ReportSeen : String[15];
  KeyPressed : Integer;
  R,G,B : Integer;
  It: Integer;
  Width : Integer;
  Height : Integer;
  Area : Integer;
  EIt: Integer;
  PicPro : Integer;
  EPicPro : Integer;
  EWidth : Integer;
  EHeight : Integer;
  EArea : Integer;
  end;

  MResults = Record
  TimeT : String[10];
  FirstWordT: String[10];
  SecondWordT: String[10];
  MLatency: Integer;
  MElatency : Integer;
  FirstWordF: String[10];
  SecondWordF: String[10];
  TimeE : String[10];
  Value : Integer;
  ReportSeen : String[15];
  TargetType : Integer;
  KeyPressed : Integer;
  It : Integer;
  EIt : Integer;
  end;

  TargetList = Record
  TargetStr : String[10];
  TargetValue : Integer;
  End;

  NonTargetList = Record
  NonTargetString : String[10];
  NonTargetValue : Integer;
  End;
```

```
Combined = Record
  Word: String[10];
  Value: Integer;
  TargetType : Integer;
End;

PracticeList = Record
  Word: String[10];
  Value: Integer;
  Timing : Integer;
End;

VRec = Record
  Correct : Integer;
  Score : Integer;
End;

PData = Record
  Timing: Integer;
  ColourSolution : Integer;
  C1R,C1G,C1B,C2R,C2G,C2B : Byte;
  KeyPress : Boolean;
End;

Sl1 = Record
  SR1,SG1,SB1 : Integer;
End;

Sl2 = Record
  SR2,SG2,SB2 : Integer;
End;

MStage = Record
  MTotM: Integer;
  METotM : Integer;
  MTargetsM : Integer;
  MNonTargetsM : Integer;
  MScoreM : Integer;
  MErrorsM : Integer;
  NullErrorsM : Integer;
  NullM : Integer;
  ELatencyM : Integer;
  TLatencyM : Integer;
  MTotP: Integer;
  METotP : Integer;
  MTargetsP : Integer;
  MNonTargetsP : Integer;
  MScoreP : Integer;
  MErrorsP : Integer;
  NullErrorsP : Integer;
  NullP : Integer;
  ELatencyP : Integer;
  TLatencyP : Integer;
End;
```

```
    TargetArray = Array Of TargetList;
    TNonTargets = Array OF NonTargetList;
    TBoth = Array Of Combined;  // yes yes yes
    TPracTargets = Array Of String[10];
    TpracNonTargets = Array Of String[10];
    TPract = Array OF PracticeList; // yes yes yes
//    SOL1 = Array[1..5] OF SL1;
//    SOL2 = Array[1..5] OF SL2;
var
  Form1: TForm1;
  InfileP, InfileM,OutfileP,OutfileM,OutfileC1, OutfileC2,OutFileP2: Text;
  Namecode,Last: String;
  Phase  : String;
  Finished, RedFirst, PEnabled,MEnabled,Masking,Panel,TextChanged: Boolean;
  Letter,LetterF,MaskCh,Colour1,Colour2,Ban2: String;
  PracticeNonTargets,Counter,valid,errors,M1Tot,M2Tot,M3Tot,M4Tot,M1ETot,M2ETot,M3ETot,M4ETot,

Age,M1TargetsM,M2TargetsM,M3TargetsM,M4TargetsM,M1NonTargetsM,M2NonTargetsM,M3NonTargetsM,M4N
onTargetsM,

M1TargetsP,M2TargetsP,M3TargetsP,M4TargetsP,M1NonTargetsP,M2NonTargetsP,M3NonTargetsP,M4NonTargetsP
,

M1ScoreP,M2ScoreP,M3ScoreP,M4ScoreP,M1ErrorsP,M2ErrorsP,M3ErrorsP,M4ErrorsP,M1ScoreM,M2ScoreM,M3S
coreM,

M4ScoreM,M1ErrorsM,M2ErrorsM,M3ErrorsM,M4ErrorsM,M1NullErrors,M2NullErrors,M3NullErrors,M4NullError
s,

M1ELatencyM,M2ELatencyM,M3ELatencyM,M4ELatencyM,M1TLatencyM,M2TLatencyM,M3TLatencyM,M4TLate
ncyM,

M1ELatencyP,M2ELatencyP,M3ELatencyP,M4ELatencyP,M1TLatencyP,M2TLatencyP,M3TLatencyP,M4TLatencyP,
  GlobalCounter,T1,T2,T3,SizerInt,PErrs,PScore,PTargets,PRTrials,P1Tot,P1ETot,P1Mean,P1EMean,P1Errors:
Integer;
  Gend: Char;
  TS: TTimeStamp;
  StartTime, StopTime : TDateTime;
  Time1,Time2,Diff,Total:Longint;
  SOL1 : Array[1..5] OF SL1;
  SOL2 : Array[1..5] OF SL2;
  MScores : Array[1..4] Of MStage;
  ScoresP : Array Of Results1;
  Scores1: Array[0..200] Of MResults;
  Scores2: Array[0..200] Of MResults;
  Scores3: Array[0..200] Of MResults;
  Scores4: Array[0..200] Of MResults;
  PTiming : Array[1..5] OF Integer;
  PCalData : Array[1..1000] Of PData;
  PracTargets : TargetArray;
  LTargets : Integer;
  PracNonTargets : TNonTargets;
  LNon : Integer;
```

```
  Both : TBoth;
  Pract : TPract; // yes yes yes
  PracTargetArr : TPracTargets;
  LPracticeTargets : Integer;
  PracNonT : TpracNonTargets;
  LPracNonT : Integer;
  Stage : Integer;
  MainT,Iteration: Integer;
  PracticeD,Main1DM,Main1DP,Main2DM,Main2DP,Main3DM,Main3DP,Main4DM,Main4DP: Double;
  GraphShown,Cal,RColour,Overide,Pressed : Boolean;
  EndTime, Performance: Single;
  ArrayLength: Integer;
  TDPrime: Double;
  FileCR: String;
  MStream : TMemoryStream;
  HowLong : Int64;
  SFactor,i,PicNo : Integer;
  CList : TStrings;
  VScores: Array[0..8] Of VRec;
  SR,SG,SB,DR,DG,DB,FSR,FSG,FSB,FDR,FDG,FDB: Integer;
  MSR,MSG,MSB,MDR,MDG,MDB,MFSR,MFSG,MFSB,MFDR,MFDG,MFDB,Equi : Integer;
  ABox1,ABox2,GoingUp,Started: Boolean;
  Solution,CValue1,CValue2,Red,Green,Blue,Red1,Green1,Blue1,Iterations: Integer;
  Turnaround,TurnAroundV,MaxRuns,PRun, PDataIndx,CIterations : integer;
  ColourDown,ColourUp,Logged,IsSizer,UserCal,Prescribed : Boolean;
  Solution1H,Solution1Av,Solution2H,Solution2Av,Solution3H,Solution3Av,Solution4H,
  Solution4Av,Solution5H,Solution5Av,Solution1T,Solution2T,Solution3T,Solution4T,
  Solution5T,FinalSolution,CheckSolution: Integer;
  TargetWidth,TargetHeight,TargetLeft,TargetTop : Integer;

S1C1R,S1C1G,S1C1B,S1C2R,S1C2G,S1C2B,S2C1R,S2C1G,S2C1B,S2C2R,S2C2G,S2C2B,S3C1R,S3C1G,S3C1B,S
3C2R,S3C2G,S3C2B,
  S4C1R,S4C1G,S4C1B,S4C2R,S4C2G,S4C2B,S5C1R,S5C1G,S5C1B,S5C2R,S5C2G,S5C2B,S1Threshold,

S2Threshold,S3Threshold,S4Threshold,S5Threshold,UVM,AVM,LVM,Cal1R,Cal1G,Cal1B,Cal2R,Cal2G,Cal2B,PC1
R,PC1G,PC1B,
  PC2R,PC2G,PC2B : Integer;
  FormR,FormG,FormB,FieldR,FieldG,FieldB,StimulusR,StimulusG,StimulusB : Byte;
  FLarge,FSmall,FSmallerCM,FBiggerCM : String;

  Const
  Fileloc = 'c:\cpt\files\';
  Ext = '.txt';
  GFloc = 'c:\Cpt\Files\ShoeBox\LumPics\';
  GAloc = 'c:\Cpt\Files\ShoeBox\All Pictures\';
  GMLoc = 'c:\Cpt\Files\ShoeBox\M\';
  GPFloc = 'c:\Cpt\Files\ShoeBox\All Pictures\';
  GPTLoc = 'c:\Cpt\Files\ShoeBox\P\';
  BMPALL = 'c:\CPT\StFiles\BMPListALL.txt';

implementation
Uses FileFunctions,Login,Sorting,Thread,DatabaseFunctions,ScoreAndUpDateM;
Procedure TrueFalse(Var Flag: Boolean); External
'TF.dll';
```

```
{$R *.dfm}
Procedure StartupTime;
Begin
  StartTime:=Now;
End;

Procedure EndupTime;
Begin
  StopTime:=Now;
End;

Procedure TForm1.ClockStop;
Var EndTime: Single;
Begin
  StopClock(EndTime);
  Diff:=Round(EndTime);
End;

// Procedure to score Practice trials
Procedure TForm1.ScoreCM;//(Latency,Elatency: Tlabel;Correct,Incorrect: TAdvSmoothButton; Key : Char;
//Sizer: TTimer; Target : TImage);
Begin
   ScoreandUpDateM.ScorePUpdate;
// This is where the old code before Unit ScoreAndUpdateP was implemented, used to be
End;

Procedure TForm1.ScoreM;
Begin
  ScoreAndUpDateM.ScoreMUpdate;
// This is where the old code before Unit ScoreAndUpdateM was implemented, used to be
End;


Procedure StopM(Timer1 : TTimer; Buttonnext,TestPhase2 : TButton;
Label1,Rtime,Latency,ETime,ELatency,Validnumber,ErrorNumber,KeyCounter : Tlabel;
Bevel1: TBevel; CaptionTargets,CaptionError,CaptionScore,Field: TLabel);
Begin
  EndupTime;
  Timer1.Enabled:=False; // just put this in here for correctness sake! ****
  Field.Visible:=False;
End;

Procedure StopM2(Timer1 : TTimer; Label1,Rtime,Etime,Latency,Elatency,Validnumber,
ErrorNumber,KeyCOunter : Tlabel; Buttonnext,TheEnd : Tbutton; Bevel1: TBevel;
CaptionTargets,CaptionError,CaptionScore,Field: TLabel);
 Begin
  EndupTime;
  Timer1.Enabled:=False; // just put this in here for correctness sake! ****
  Field.Visible:=False;
  Rtime.Caption:='Mean T.L.';
  ETime.Caption:='Mean E.L.';
  Label1.Caption:='End of Phase 2';
  TheEnd.Visible:=True;
```

```
  Buttonnext.Visible:=False;
End;

procedure TForm1.BtnShowBarGraphClick(Sender: TObject);
begin
//   NameCode:='132722567';
   Form2.Visible:=true;
   //TForm2.PlotDPrime(NameCode);?
 end;


procedure TForm1.ChkS1Click(Sender: TObject);
Var R: Integer;
begin
if ChkS1.Checked then
  Begin
    CheckSolution:=1;
//    ChkS1.Checked:=True;
    ChkS2.Checked:=False;
    ChkS3.Checked:=False;
    ChkS4.Checked:=False;
    ChkS5.Checked:=False;
    Jog.Value:=Sol2[CheckSolution].SR2;
  End;
end;

procedure TForm1.ChkS2Click(Sender: TObject);
begin
if ChkS2.Checked then
    Begin
      CheckSolution:=2;
//      ChkS2.Checked:=True;
      ChkS1.Checked:=False;
      ChkS3.Checked:=False;
      ChkS4.Checked:=False;
      ChkS5.Checked:=False;
      Jog.Value:=Sol2[CheckSolution].SR2;
    End;
end;

procedure TForm1.ChkS3Click(Sender: TObject);
begin
if ChkS3.Checked then
    Begin
    CheckSolution:=3;
//    ChkS3.Checked:=True;
    ChkS2.Checked:=False;
    ChkS1.Checked:=False;
    ChkS4.Checked:=False;
    ChkS5.Checked:=False;
    Jog.Value:=Sol2[CheckSolution].SG2;
    End;
end;
```

```
procedure TForm1.ChkS4Click(Sender: TObject);
begin
  if ChkS4.Checked then
  Begin
    CheckSolution:=4;
//    ChkS4.Checked:=True;
    ChkS1.Checked:=False;
    ChkS3.Checked:=False;
    ChkS2.Checked:=False;
    ChkS5.Checked:=False;
    Jog.Value:=Sol1[CheckSolution].SR1;
  End;
end;

procedure TForm1.ChkS5Click(Sender: TObject);
begin
  if ChkS5.Checked then
    Begin
      CheckSolution:=5;
//      ChkS5.Checked:=True;
      ChkS2.Checked:=False;
      ChkS3.Checked:=False;
      ChkS4.Checked:=False;
      ChkS1.Checked:=False;
      Jog.Value:=Sol2[CheckSolution].SR2;
    End;
end;

procedure TForm1.ChkT1Click(Sender: TObject);
begin
  if ChKT1.Checked then
    Begin
      CFirst.Interval:=PTiming[1];
      CSecond.Interval:=PTiming[1];
      ChKT2.Checked:=False;
      ChkT3.Checked:=False;
      ChkT4.Checked:=False;
      ChkT5.Checked:=False;
    End;
end;

procedure TForm1.ChkT2Click(Sender: TObject);
begin
  if ChKT2.Checked then
    Begin
      CFirst.Interval:=PTiming[2];
      CSecond.Interval:=PTiming[2];
      ChKT1.Checked:=False;
      ChkT3.Checked:=False;
      ChkT4.Checked:=False;
      ChkT5.Checked:=False;
    End;
end;
```

```
procedure TForm1.ChkT3Click(Sender: TObject);
begin
  if ChKT3.Checked then
    Begin
      CFirst.Interval:=PTiming[3];
      CSecond.Interval:=PTiming[3];
      ChKT2.Checked:=False;
      ChkT1.Checked:=False;
      ChkT4.Checked:=False;
      ChkT5.Checked:=False;
    End;
end;

procedure TForm1.ChkT4Click(Sender: TObject);
begin
  if ChKT4.Checked then
    Begin
      CFirst.Interval:=PTiming[4];
      CSecond.Interval:=PTiming[4];
      ChKT2.Checked:=False;
      ChkT3.Checked:=False;
      ChkT1.Checked:=False;
      ChkT5.Checked:=False;
    End;
end;

procedure TForm1.ChkT5Click(Sender: TObject);
begin
  if ChKT5.Checked then
    Begin
      CFirst.Interval:=PTiming[5];
      CSecond.Interval:=PTiming[5];
      ChKT1.Checked:=False;
      ChkT2.Checked:=False;
      ChkT3.Checked:=False;
      ChkT4.Checked:=False;
    End;
end;

procedure TForm1.ChoosePracticeOrTest(Sender: TObject);
begin
  Memo1.Visible:=false;
  BtnGetDetailsPractice.Visible:=True;
  BtnChoosePracticeOrTest.Visible:=False;
  Line2.Visible:=false;
  Banner1.Caption.Text:='Click Information button';
end;

procedure TForm1.EdColourChange(Sender: TObject);
Var Resp : String;
    Value : Integer;
begin
  Resp:=EdColour.Text;
  if Resp<> " then
```

```
  Begin
   TextChanged:=True;
   //  Else TextChanged:=False;
   if Not TryStrToInt(Resp,Value) then
    Begin
     ShowMessage('Invalid Response');
     EdColour.Text:='';
    End;
  End;
end;

Function CheckFloor: Boolean;
Begin
  if (DR<LVM) AND (DG<LVM) AND (DB<LVM) then
  Result:=True
    Else
      Result:=False;
End;

Function CheckRoof: Boolean;
Begin
  if (DR>UVM) AND (DG>UVM) AND (DB>UVM) then
  Result:=True
    Else
      Result:=False;
End;

Function CheckDetected : Boolean;
Begin
  if Pract[GlobalCounter-1].Value=1 then
  Begin
  if ScoresP[GlobalCounter-1].ReportSeen='True Positive'
  Then
    Result:=True
    Else
      Result:=False;
  End;
End;

procedure Calibrate;
Var Floor,Roof,Detected,Threshold : Boolean;
begin
  if GlobalCounter =1 then
    Begin
 {     FSR:=122;
      FSG:=122;
      FSB:=122;
 {    FDR:=108;
     FDG:=108;
     FDB:=108;  }
{      FDR:=119;
      FDG:=119;
      FDB:=119;  }
      GoingUp :=True;
```

```
    Detected:=False;
   End;
  if GlobalCounter>1 Then
  Begin
   Roof:=CheckRoof;
   Floor:=CheckFloor;
   if Roof then GoingUp:=False;
   if Floor then GoingUp:=True;
//  GoingUp:=Not(CheckRoof);
   Detected:=False;
   Detected:=CheckDetected;
   if Detected then
    Inc(Turnaround)
     Else if (TurnAround>0) then
         Dec(Turnaround);
   Threshold:=False;
   if TurnAround=TurnAroundV then
    Begin
     Threshold:=True;
     TurnAround:=0;
     TurnAroundV:=TurnAroundV+1;
    End;
   if (GoingUp) AND (Threshold=False) then
   Begin
    Inc(DR); //FDR
    Inc(DG); //FDG
    Inc(DB); //FDB
   End;
   if (GoingUp) AND (Threshold) then
   Begin
    GoingUp:=False;
    Dec(DR);
    Dec(DG);
    Dec(DB);
   End
    Else
     if (GoingUp=False) AND (Threshold=False) then
      Begin
       Dec(DR);
       Dec(DG);
       Dec(DB);
      End
       Else
        if (GoingUp=False) AND (Threshold) then
         Begin
          GoingUp:=True;
          Inc(DR);
          Inc(DG);
          Inc(DB);
         End;
  End;
   Form1.LblR.Caption:=IntToStr(DR);
   Form1.LblG.Caption:=IntToStr(DG);
   Form1.LblB.Caption:=IntToStr(DB);
```

```
end;

//THIS ONE DISPLAYS THE CALIBRATES M EQUILUMINANT TARGETS
Procedure TForm1.DisplayCM;
Var
 Filename,FromFolder,ToFolder,FList : String;
 Var AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,AToBB,RR: Integer;
  ABox1,ABox2: Boolean;
  Adjustment : Real;
BEGIN
  StopClock(EndTime);
  GlobalCounter:=GlobalCounter+1;
  Progressbar1.Next;
  ABox2:=True;
  ABox1:=True;
  IF GlobalCounter<=(ArrayLength) THEN
   BEGIN
     If Pract[GlobalCounter].Value=1 Then
       Begin
         Valid:=Valid+1;
         Validnumber.Caption:=IntToStr(valid);
       End;
     If Pract[GlobalCounter].Value=0 Then
       Begin
         PracticeNonTargets:=PRacticeNonTargets+1;
       End;
    if (GlobalCounter>1) And (ScoresP[GlobalCounter-1].KeyPressed<>1) then
     Begin
       EndupTime;
       if Pract[GlobalCounter-1].Value=1 then
         Begin
           ScoresP[GlobalCounter-1].FirstWordT:='Target';
           ScoresP[GlobalCounter-1].SecondWordT:=Letter;
           ScoresP[GlobalCounter-1].TimeE:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
           ScoresP[GlobalCounter-1].R:=DR-Equi; // FDR-Equi;
           ScoresP[GlobalCounter-1].G:=DG-Equi; // FDG-Equi;
           ScoresP[GlobalCounter-1].B:=DB-Equi; // FDB-Equi;
           ScoresP[GlobalCounter-1].KeyPressed:=0;
           ScoresP[GlobalCounter-1].ReportSeen:='False Negative';
           ScoresP[GlobalCounter-1].Value:=0;
         End;
       if Pract[GlobalCounter-1].Value=0 then
         Begin
           ScoresP[GlobalCounter-1].FirstWordT:='Non-target';
           ScoresP[GlobalCounter-1].SecondWordT:=Letter;
           ScoresP[GlobalCounter-1].FirstWordF:='blank';
           ScoresP[GlobalCounter-1].SecondWordF:=LetterF;
           ScoresP[GlobalCounter-1].TimeE:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
           ScoresP[GlobalCounter-1].R:=DR-Equi; //FDR-Equi;
           ScoresP[GlobalCounter-1].G:=DG-Equi; //FDG-Equi;
           ScoresP[GlobalCounter-1].B:=DB-Equi; //FDB-Equi;
           ScoresP[GlobalCounter-1].KeyPressed:=0;
           ScoresP[GlobalCounter-1].ReportSeen:='True Negative';
           ScoresP[GlobalCounter-1].Value:=1;
```

```
    End;

 With Form1.ADOPTable1 Do
  Begin
    Append;
    Fields.FieldByName('ID').Value:=Namecode;
    Fields.FieldByName('TimeT').Value:=ScoresP[GlobalCounter-1].TimeE;
    Fields.FieldByName('Stimulus Type').Value:=ScoresP[GlobalCounter-1].FirstWordT;
    Fields.FieldByName('Stimulus Item').Value:=ScoresP[GlobalCounter-1].SecondWordT;
    Fields.FieldByName('Non Target').Value:=ScoresP[GlobalCounter-1].FirstWordF;
    Fields.FieldByName('Non Target Item').Value:=ScoresP[GlobalCounter-1].SecondWordF;
    Fields.FieldByName('TimeE').Value:=ScoresP[GlobalCounter-1].TimeE;
    Fields.FieldByName('Value').Value:=ScoresP[GlobalCounter-1].Value;
    Fields.FieldByName('Report Seen').Value:=ScoresP[GlobalCounter-1].ReportSeen;
    Fields.FieldByName('Key Pressed').Value:=ScoresP[GlobalCounter-1].KeyPressed;
    Fields.FieldByName('R').Value:=ScoresP[GlobalCounter-1].R;
    Fields.FieldByName('G').Value:=ScoresP[GlobalCounter-1].G;
    Fields.FieldByName('B').Value:=ScoresP[GlobalCounter-1].B;
    UpdateRecord;
    Post;
   End;
  End;
Letter:=Pract[GlobalCounter].Word;
Filename:=Concat(GFloc,Pract[GlobalCounter].Word,'.bmp');
MStream:=TMemoryStream.Create;
if GlobalCounter=1 then
  Calibrate;
if (GlobalCounter>1) AND (Pract[GlobalCounter-1].Value=1)
 then
  Calibrate;
If Pract[GlobalCounter].Value=1 Then
  Begin
    AFR:=Sr;
    AFG:=Sg;
    AFB:=Sb;
    AToR:=Dr;
    AToG:=Dg;
    AToB:=Db;
    AFBR:=FSR;
    AFBG:=FSG;
    AFBB:=FSB;
    AToBR:=FDR;
    AToBG:=FDG;
    AToBB:=FDB;
    Validnumber.Caption:=IntToStr(valid);
    Thread.ImageThreadM.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
    AToBB,ABox1,ABox2,Filename);
  End;
  If Pract[GlobalCounter].Value=0 Then
    Begin
      AFR:=Sr;
      AFG:=Sg;
      AFB:=Sb;
      AToR:=Equi;//Dr;
```

```
        AToG:=Equi;//Dg;
        AToB:=Equi;//Db;
        AFBR:=FSR;
        AFBG:=FSG;
        AFBB:=FSB;
        AToBR:=Equi;
        AToBG:=Equi;
        AToBB:=Equi;
        Letter:=Pract[GlobalCounter].Word;
       Repeat
        RR:=Random(ArrayLength);
       Until (Pract[RR].Word<>'blank') AND (RR>0);
        LetterF:=Pract[RR].Word;
        Filename:=Concat(GFloc,Pract[RR].Word,'.bmp');
        Thread.ImageThreadM.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
          AToBB,ABox1,ABox2,Filename);
       End;
  end;
IF GLobalCounter>ArrayLength Then
  Begin
   EndupTime;
   Form1.ADOPTable1.Active:=False;
   Form1.ADOPTable1.Close;
   Timer1.Enabled:=false;
   RTime.Caption:='Mean T.L. ';
   ETime.Caption:='Mean E.L. ';
   IF (Counter>0) Then P1Mean:=(P1Tot DIV Counter);
   If (Errors>0) Then P1EMean:=(P1ETot DIV Errors);
   Latency.Caption:=IntToStr(P1Mean);
   ELatency.Caption:=IntToStr(P1EMean);
   FileFunctions.EndPracticeCM(PracticeD,Adjustment);  //?
   Adjustment:=INT(Round(Adjustment));
   if Adjustment<2
    Then Adjustment:=2;
    Adjustment:=Adjustment+AVM;
    AFR:=Sr;
    AFG:=Sg;
    AFB:=Sb;
    AToR:=Equi+Trunc(Adjustment);
    AToG:=Equi+Trunc(Adjustment);
    AToB:=Equi+Trunc(Adjustment);
    AFBR:=FSR;
    AFBG:=FSG;
    AFBB:=FSB;
    AToBR:=Equi; //  -Trunc(Adjustment); // + -> -
    AToBG:=Equi; //-Trunc(Adjustment);
    AToBB:=Equi; //-Trunc(Adjustment);
    FromFolder:=GAloc;
    ToFolder:=GMLoc;
    FList:=BMPALL;
    Form1.Cursor:=crHourGlass;
    RPanel.Visible:=Panel;
    Field.Visible:=True;
    Started:=false;
```

```
    Form1.Banner1.Caption.Text:='Processing - please wait';
    Thread.ImageThreadAllM.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
    AToBB,ABox1,ABox2,FromFolder,ToFolder,FList);
  end;
   Application.ProcessMessages;
END;

Procedure TForm1.WritePreviousStage;
Begin
  WritePreviousStage;
  Application.ProcessMessages;
End;

Procedure TForm1.ScoreMStages;
Begin
  ScoreAndUpDateM.ScoreMStages;
  Application.ProcessMessages;
End;

// THIS ONE DISPLAYS THE TARGETS
// displays the stimulus and then starts timer2 which switches
// the mask off for a specified period
Procedure TForm1.DisplayM(VAR Letter,Last: String);
 Var
  AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,AToBB,RR: Integer;
  ABox1,ABox2: Boolean;
  Filename : String;
  Stage : Integer;
BEGIN
  StopClock(EndTime);
  ABox1:=True;
  ABox2:=True;
  Stage:=StrToInt(Phase);
  GlobalCounter:=GlobalCounter+1;
  Progressbar1.Next;
  IF GlobalCounter<=(ArrayLength) THEN
   BEGIN
    if Stage=1 then
    Begin
      If Both[GlobalCounter].Value=0 Then
      Begin
        if Both[GlobalCounter].TargetType=0 then
          Begin
            MScores[Stage].MTargetsM:=MScores[Stage].MTargetsM+1;
            Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsM);
          End;
        if Both[GlobalCounter].TargetType=1 then
          Begin
            MScores[Stage].MTargetsP:=MScores[Stage].MTargetsP+1;
            ValidnumberP.Caption:=IntToStr(MScores[Stage].MTargetsP);
          End;
      End;
      If Both[GlobalCounter].Value=1 Then
      Begin
```

```
    if Both[GlobalCounter].TargetType=0 then
      Begin
       if Both[GlobalCounter].Word<>'Null' then
         Begin
           MScores[Stage].MNonTargetsM:=MScores[Stage].MNonTargetsM+1; //         MScores[Stage].
           NValidM.Caption:=IntToStr(MScores[Stage].MNonTargetsM);
         End;
          if Both[GlobalCounter].Word='Null' then
          Begin
           MScores[Stage].NullM:=MScores[Stage].Nullm+1;
           NullTM.Caption:=IntToStr(MScores[Stage].NullM);
          End;
      End;
    if Both[GlobalCounter].TargetType=1 then
      Begin
       if Both[GlobalCounter].Word<>'Null' then
         Begin
           MScores[Stage].MNonTargetsP:=MScores[Stage].MNonTargetsP+1;
           NvalidP.Caption:=IntToStr(MScores[Stage].MNonTargetsP);
         End;
       if Both[GlobalCounter].Word='Null' then
          Begin
           MScores[Stage].NullP:=MScores[Stage].NullP+1;
           NullTP.Caption:=IntToStr(MScores[Stage].NullP);
          End;
      End;
  End;
End;
if Stage=2 then
Begin
  If Both[GlobalCounter].Value=0 Then
  Begin
    if Both[GlobalCounter].TargetType=0 then
      Begin
       MScores[Stage].MTargetsM:=MScores[Stage].MTargetsM+1;
       Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsM);
      End;
    if Both[GlobalCounter].TargetType=1 then
      Begin
       MScores[Stage].MTargetsP:=MScores[Stage].MTargetsP+1;
       ValidnumberP.Caption:=IntToStr(MScores[Stage].MTargetsP);
      End;
  End;
  If Both[GlobalCounter].Value=1 Then
  Begin
    if Both[GlobalCounter].TargetType=0 then
      Begin
       if Both[GlobalCounter].Word<>'Null' then
         Begin
           MScores[Stage].MNonTargetsM:=MScores[Stage].MNonTargetsM+1; //         MScores[Stage].
           NValidM.Caption:=IntToStr(MScores[Stage].MNonTargetsM);
         End;
          if Both[GlobalCounter].Word='Null' then
          Begin
```

```
          MScores[Stage].NullM:=MScores[Stage].Nullm+1;
          NullTM.Caption:=IntToStr(MScores[Stage].NullM);
         End;
      End;
    if Both[GlobalCounter].TargetType=1 then
      Begin
       if Both[GlobalCounter].Word<>'Null' then
         Begin
          MScores[Stage].MNonTargetsP:=MScores[Stage].MNonTargetsP+1;
          NvalidP.Caption:=IntToStr(MScores[Stage].MNonTargetsP);
         End;
       if Both[GlobalCounter].Word='Null' then
         Begin
          MScores[Stage].NullP:=MScores[Stage].NullP+1;
          NullTP.Caption:=IntToStr(MScores[Stage].NullP);
         End;
      End;
  End;
End;
if Stage=3 then
Begin
  If Both[GlobalCounter].Value=0 Then
  Begin
    if Both[GlobalCounter].TargetType=0 then
      Begin
       MScores[Stage].MTargetsM:=MScores[Stage].MTargetsM+1;
       Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsM);
      End;
    if Both[GlobalCounter].TargetType=1 then
      Begin
       MScores[Stage].MTargetsP:=MScores[Stage].MTargetsP+1;
       ValidnumberP.Caption:=IntToStr(MScores[Stage].MTargetsP);
      End;
  End;
  If Both[GlobalCounter].Value=1 Then
  Begin
    if Both[GlobalCounter].TargetType=0 then
      Begin
       if Both[GlobalCounter].Word<>'Null' then
         Begin
          MScores[Stage].MNonTargetsM:=MScores[Stage].MNonTargetsM+1; //          MScores[Stage].
          NValidM.Caption:=IntToStr(MScores[Stage].MNonTargetsM);
         End;
        if Both[GlobalCounter].Word='Null' then
        Begin
          MScores[Stage].NullM:=MScores[Stage].Nullm+1;
          NullTM.Caption:=IntToStr(MScores[Stage].NullM);
         End;
      End;
    if Both[GlobalCounter].TargetType=1 then
      Begin
       if Both[GlobalCounter].Word<>'Null' then
         Begin
          MScores[Stage].MNonTargetsP:=MScores[Stage].MNonTargetsP+1;
```

```
          NvalidP.Caption:=IntToStr(MScores[Stage].MNonTargetsP);
        End;
      if Both[GlobalCounter].Word='Null' then
        Begin
          MScores[Stage].NullP:=MScores[Stage].NullP+1;
          NullTP.Caption:=IntToStr(MScores[Stage].NullP);
        End;
    End;
  End;
End;
if Stage=4 then
Begin
  If Both[GlobalCounter].Value=0 Then
  Begin
    if Both[GlobalCounter].TargetType=0 then
      Begin
        MScores[Stage].MTargetsM:=MScores[Stage].MTargetsM+1;
        Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsM);
      End;
    if Both[GlobalCounter].TargetType=1 then
      Begin
        MScores[Stage].MTargetsP:=MScores[Stage].MTargetsP+1;
        ValidnumberP.Caption:=IntToStr(MScores[Stage].MTargetsP);
      End;
  End;
  If Both[GlobalCounter].Value=1 Then
  Begin
    if Both[GlobalCounter].TargetType=0 then
      Begin
       if Both[GlobalCounter].Word<>'Null' then
        Begin
          MScores[Stage].MNonTargetsM:=MScores[Stage].MNonTargetsM+1; //  MScores[Stage].
          NValidM.Caption:=IntToStr(MScores[Stage].MNonTargetsM);
        End;
        if Both[GlobalCounter].Word='Null' then
        Begin
          MScores[Stage].NullM:=MScores[Stage].Nullm+1;
          NullTM.Caption:=IntToStr(MScores[Stage].NullM);
        End;
      End;
    if Both[GlobalCounter].TargetType=1 then
      Begin
        if Both[GlobalCounter].Word<>'Null' then
          Begin
            MScores[Stage].MNonTargetsP:=MScores[Stage].MNonTargetsP+1;
            NvalidP.Caption:=IntToStr(MScores[Stage].MNonTargetsP);
          End;
        if Both[GlobalCounter].Word='Null' then
          Begin
            MScores[Stage].NullP:=MScores[Stage].NullP+1;
            NullTP.Caption:=IntToStr(MScores[Stage].NullP);
          End;
      End;
  End;
```

```
End;
ScoreAndUpDateM.ScoreMStages;    // should be just write data away (GlobalCounter)
Letter:=Both[GlobalCounter].Word;
MStream:=TMemoryStream.Create;
if Letter<>'Null' then
  Begin
    If Both[GlobalCounter].TargetType=0 Then
    Filename:=Concat('c:\Cpt\Files\ShoeBox\M\',Both[GlobalCounter].Word,'.bmp');
    If Both[GlobalCounter].TargetType=1 Then
    Filename:=Concat('c:\Cpt\Files\ShoeBox\P\',Both[GlobalCounter].Word,'.bmp');
    MStream.LoadFromFile(Filename);
    Target.Hide;
    Fixation.Enabled:=True;
  End;
if Letter='Null' then
  Begin
    if Both[GlobalCounter].TargetType=0 then
    Begin
     AFR:=Sr;//5; ?
     AFG:=SG;//5;
     AFB:=SB;//5;
     AToR:=Equi;//119;
     AToG:=Equi;
     AToB:=Equi;
     AFBR:=MFSR; //?
     AFBG:=MFSG;
     AFBB:=MFSB;
     AToBR:=Equi;
     AToBG:=Equi;
     AToBB:=Equi;
   Repeat
     RR:=Random(ArrayLength);
   Until (Both[RR].Word<>'Null') AND (RR>0);
     Filename:=Concat(GALoc,Both[RR].Word,'.bmp');
     Thread.ImageThreadM.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
       AToBB,ABox1,ABox2,Filename);
   End;
    if FinalSOlution=0 Then FinalSolution:=4;
    if Both[GlobalCounter].TargetType=1 then
    Begin
      if FinalSolution=1 then
        Begin
          AFR:=Sr;//5;
          AFG:=Sg;//5;
          AFB:=Sb;//5;
          AToR:=FDr;
          AToG:=FDg;
          AToB:=FDb;
          AFBR:=MFSR;//250; ?
          AFBG:=MFSG;//250;
          AFBB:=MFSB;//250;
          AToBR:=FDR;
          AToBG:=FDG;
          AToBB:=FDB;
```

```
  End;
 if FinalSolution=2 then
  Begin
   AFR:=Sr;//5;
   AFG:=Sg;//5;
   AFB:=Sb;//5;
   AToR:=FDr;
   AToG:=FDg;
   AToB:=FDb;
   AFBR:=MFSR;//250; ?
   AFBG:=MFSG;//250;
   AFBB:=MFSB;//250;
   AToBR:=FDR;
   AToBG:=FDG;
   AToBB:=FDB;
  End;
 if FinalSolution=3 then
  Begin
   AFR:=Sr;//5;
   AFG:=Sg;//5;
   AFB:=Sb;//5;
   AToR:=FDr;
   AToG:=FDg;
   AToB:=FDb;
   AFBR:=MFSR;//250; ?
   AFBG:=MFSG;//250;
   AFBB:=MFSB;//250;
   AToBR:=FDR;
   AToBG:=FDG;
   AToBB:=FDB;
  End;
 if FinalSolution=4 then
  Begin
   AFR:=Sr;//5;
   AFG:=Sg;//5;
   AFB:=Sb;//5;
   AToR:=FDr;
   AToG:=FDg;
   AToB:=FDb;
   AFBR:=MFSR;//250; ?
   AFBG:=MFSG;//250;
   AFBB:=MFSB;//250;
   AToBR:=FDR;
   AToBG:=FDG;
   AToBB:=FDB;
  End;
 if FinalSolution=5 then
  Begin
   AFR:=Sr;//5;
   AFG:=Sg;//5;
   AFB:=Sb;//5;
   AToR:=FDr;
   AToG:=FDg;
   AToB:=FDb;
```

```
          AFBR:=MFSR;//250; ?
          AFBG:=MFSG;//250;
          AFBB:=MFSB;//250;
          AToBR:=FDR;
          AToBG:=FDG;
          AToBB:=FDB;
         End;
      Repeat
       RR:=Random(ArrayLength);
       Until (Both[RR].Word<>'Null') AND (RR>0);
       Filename:=Concat(GALoc,Both[RR].Word,'.bmp');
       Thread.ImageThreadM.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
         AToBB,ABox1,ABox2,Filename);
      End;
    End; //end if Letter = Null
  End;
If GlobalCounter=ArrayLength+1 Then
 BEGIN
  ScoreMStages;
  Finished:=true;
  Timer1.Enabled:=false;
  Rtime.Caption:='Mean T.L.';
  ETime.Caption:='Mean E.L.';
  if Stage=1 then
   Begin
    if (MScores[Stage].MTotM>0) And (MScores[Stage].METotM>0)  then
    Begin
     M1TLatencyM:=MScores[Stage].TLatencyM DIV MScores[Stage].MTotM;
     Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsM);
     M1ELatencyM:=MScores[Stage].ELatencyM DIV MScores[Stage].METotM;
     ELatency.Caption:=IntToStr(M1ELatencyM);
     Latency.Caption:=IntToStr(M1TLatencyM);
     Etime.Visible:=True;
     RPanel.Visible:=Panel;
    End;
    if (MScores[Stage].MTotP>0) And (MScores[Stage].METotP>0)  then
    Begin
     M1TLatencyP:=MScores[Stage].TLatencyP DIV MScores[Stage].MTotP;
     Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsP);
     M1ELatencyP:=MScores[Stage].ELatencyP DIV MScores[Stage].METotP;
     ELatency.Caption:=IntToStr(M1ELatencyP);
     Latency.Caption:=IntToStr(M1TLatencyP);
     Etime.Visible:=True;
     RPanel.Visible:=Panel;
    End;
    EndMain(Main1DM,Main1DP,Stage);
   End;
  if Stage=2 then
   Begin
    if (MScores[Stage].MTotM>0) And (MScores[Stage].METotM>0)  then
    Begin
     M2TLatencyM:=MScores[Stage].TLatencyM DIV MScores[Stage].MTotM;
     Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsM);
     M2ELatencyM:=MScores[Stage].ELatencyM DIV MScores[Stage].METotM;
```

```
      ELatency.Caption:=IntToStr(M2ELatencyM);
      Latency.Caption:=IntToStr(M2TLatencyM);
      Etime.Visible:=True;
      RPanel.Visible:=Panel;
    End;
    if (MScores[Stage].MTotP>0) And (MScores[Stage].METotP>0)  then
    Begin
      M2TLatencyP:=MScores[Stage].TLatencyP DIV MScores[Stage].MTotP;
      Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsP);
      M2ELatencyP:=MScores[Stage].ELatencyP DIV MScores[Stage].METotP;
      ELatency.Caption:=IntToStr(M2ELatencyP);
      Latency.Caption:=IntToStr(M2TLatencyP);
      Etime.Visible:=True;
      RPanel.Visible:=Panel;
    End;
    EndMain(Main2DM,Main2DP,Stage);
  End;
if Stage=3 then
  Begin
    if (MScores[Stage].MTotM>0) And (MScores[Stage].METotM>0)  then
    Begin
      M3TLatencyM:=MScores[Stage].TLatencyM DIV MScores[Stage].MTotM;
      Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsM);
      M3ELatencyM:=MScores[Stage].ELatencyM DIV MScores[Stage].METotM;
      ELatency.Caption:=IntToStr(M3ELatencyM);
      Latency.Caption:=IntToStr(M3TLatencyM);
      Etime.Visible:=True;
      RPanel.Visible:=Panel;
    End;
    if (MScores[Stage].MTotP>0) And (MScores[Stage].METotP>0)  then
    Begin
      M3TLatencyP:=MScores[Stage].TLatencyP DIV MScores[Stage].MTotP;
      Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsP);
      M3ELatencyP:=MScores[Stage].ELatencyP DIV MScores[Stage].METotP;
      ELatency.Caption:=IntToStr(M3ELatencyP);
      Latency.Caption:=IntToStr(M3TLatencyP);
      Etime.Visible:=True;
      RPanel.Visible:=Panel;
    End;
    EndMain(Main3DM,Main3DP,Stage);
  End;
if Stage=4 then
  Begin
    if (MScores[Stage].MTotM>0) And (MScores[Stage].METotM>0)  then
    Begin
      M4TLatencyM:=MScores[Stage].TLatencyM DIV MScores[Stage].MTotM;
      Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsM);
      M4ELatencyM:=MScores[Stage].ELatencyM DIV MScores[Stage].METotM;
      ELatency.Caption:=IntToStr(M4ELatencyM);
      Latency.Caption:=IntToStr(M4TLatencyM);
      Etime.Visible:=True;
      RPanel.Visible:=Panel;
    End;
    if (MScores[Stage].MTotP>0) And (MScores[Stage].METotP>0)  then
```

```
        Begin
          M4TLatencyP:=MScores[Stage].TLatencyP DIV MScores[Stage].MTotP;
          Validnumber.Caption:=IntToStr(MScores[Stage].MTargetsP);
          M4ELatencyP:=MScores[Stage].ELatencyP DIV MScores[Stage].METotP;
          ELatency.Caption:=IntToStr(M4ELatencyP);
          Latency.Caption:=IntToStr(M4TLatencyP);
          Etime.Visible:=True;
          RPanel.Visible:=Panel;
        End;
        EndMain(Main4DM,Main4DP,Stage);
      End;
    FileFunctions.OpenItemMain(NameCode,Stage);
    BtnChoose.Visible:=false;
    BtnStartTest.Caption:='Next';
    BtnStartTest.Visible:=True; // make TestPhase2 button visible here
    Field.Visible:=False;
  End;
  Application.ProcessMessages;
END;

Procedure InitPracticeT;
Var Ind: Integer;
Begin
 for Ind := 0 to ArrayLength do
   Begin
     ScoresP[Ind].TimeT:='';
     ScoresP[Ind].FirstWordT:='';
     ScoresP[Ind].SecondWordT:='';
     ScoresP[Ind].M1Latency:=0;
     ScoresP[Ind].M1Elatency:=0;
     ScoresP[Ind].FirstWordF:='';
     ScoresP[Ind].SecondWordF:='';
     ScoresP[Ind].TimeE:='';
     ScoresP[Ind].ReportSeen:='';
     ScoresP[Ind].KeyPressed:=0;
     ScoresP[Ind].R:=0;
     ScoresP[Ind].G:=0;
     ScoresP[Ind].B:=0;
     ScoresP[Ind].Value:=0;
   End;
End;

procedure TForm1.StartAllTesting(Sender: TObject);
begin
 Progressbar1.Visible:=True;
 ProgressBar1.Minimum:=0;
 ProgressBar1.Position:=0;
 ProgressBar1.Step:=1;
 FForm1.Fill.Color:=RGB(FormR,FormG,FormB);
 FForm1.Fill.ColorTo:=RGB(FormR,FormG,FormB);
 Field.Color:=RGB(FieldR,FieldG,FieldB);
 Stimulus.Color:=RGB(StimulusR,StimulusG,StimulusB);
 PanelCValues.Top:=385;
 PanelCValues.Left:=672;
```

```
If (Phase = 'CM') Then
  Begin
   SR:=MSR;//5
   SG:=MSG;//5
   SB:=MSB;//5 MSB
   DR:=MDR;//122; MDR
   DG:=MDG;//122; MDG
   DB:=MDB;//122; MDB

   FSR:=MFSR;//250; MFSR
   FSG:=MFSG;//250; MFSG
   FSB:=MFSB;//250; MFSB
   FDR:=MFDR;//119; MFDR
   FDG:=MFDG;//119; MFDG
   FDB:=MFDB;//119; MFDB  // this the equiluminance point 'Box 2' foreground calculation
   SetLength(ScoresP,ArrayLength+1);
   InitPracticeT;
   Sorting.SortPractice(LPracticeTargets,LPracNonT,PracTargetArr,PracNonT,Pract);
   FileFunctions.ChecklistP;
   ProgressBar1.Maximum:=ArrayLength;
   Timer1.Interval:=T1;       // 1000
   Timer2.Interval:=T2;       // 100 actual stimulus exposure time
   Timer3.Interval:=T3;       // 100 mask exposure time
   Sizer.Interval:=SizerInt;  // The interval of the sizing timer
   P1Tot:=0;
   Banner1.Caption.Text:='Contrast Calibration';
   RPanel.Visible:= Panel;
   PanelCValues.Visible:=Panel;
   GlobalCounter:=0;
   Label1.Caption:='Testing...';
   StartupTime;
   BtnChoose.Visible:=false;
   Form1.KeyPreview:=true;
   Started:=true;
   Timer1.Enabled:=true;
  End;
If (Phase <>'CM') Then
Begin
  RTime.Caption:='T Latency';
  ETime.Caption:='E Latency';
  LblFlickerHz.Visible:=False;
  MatrixPanel.Visible:=False;
  Stimulus.Visible:=False;
  Field.Visible:=False;
  Timer1.Interval:=T1; //1500
  Timer2.Interval:=T2; //500
  Timer3.Interval:=T3; //500
  Sizer.Interval:=SizerInt;  // The interval of the sizing timer
  RPanel.Visible:=Panel;// Not Panel then
  GlobalCounter:=0;
  if Phase = 'CP' Then
    Begin
      ProgressBar1.Position:=0;
      Label1.Caption:='Testing Stage '+Phase+'...';
```

```
      Banner1.Caption.Text:='Testing Stage '+Phase;
     End;
   if (Phase<>'CM') AND (Phase<> 'CP') then
     Begin
      Label1.Caption:='Testing Stage 4';
      FileFunctions.OpenMainFiles(LTargets,LNon,Phase);
      Arraylength:=LTargets+LNon;
      Setlength(Both,ArrayLength+1);
     //  ProgressBar1.Position:=0;
      ProgressBar1.Maximum:=ArrayLength;
     //  Sorting.SortAll(LTargets,LNon,PracTargets,PracNonTargets,Both);
     //  Sorting.SortTargetType(Both,Arraylength);
     End;
   Arraylength:=LTargets+LNon;    //MainT:=LTargets+LNon;?
   Setlength(Both,ArrayLength+1);
   Sorting.SortAll(LTargets,LNon,PracTargets,PracNonTargets,Both);
   Sorting.SortTargetType(Both,Arraylength);
//    ProgressBar1.Position:=0;
   FileFunctions.Checklist1;
   Label1.Caption:='Testing...';
   StartupTime;
   Timer1.Enabled:=true;
   BtnChoose.Visible:=false;
   Form1.KeyPreview:=true;
   Started:=true;
  End;
end;

//displays the various instructon lines and switches on the practice and test
// buttons
procedure TForm1.LoadInstructionsClick(Sender: TObject);
Var S: TFileStream;
    TempS : String;
begin
  ADOConnection1.ConnectionString:=Login.CString;//    'Provider=MSDASQL.1;Password=douglasjohn;Persist
Security Info=True;User ID=Douglas;Data Source=UCTMSQLE1;Initial Catalog=uctexp1';
  ADOConnection1.Connected:=True;
  DataBaseFunctions.CreatePSummaryTable;
  DataBaseFunctions.CreatePTable(login.VNumber);
  DataBaseFunctions.CreateMSummaryTable;
  DataBaseFunctions.CreateMTable(login.VNumber);
  Logged:=True;
     ADOSettingsQuery.SQL.Text:=('Select * from settings');
     ADOSettingsQuery.Active:=true;
     ADOSettingsQuery.First;
   With ADOSettingsQuery Do
     Begin
       T1:=Fields.FieldByName('Timer1').Value;
       T2:=Fields.FieldByName('Timer2').Value;
       T3:=Fields.FieldByName('Timer3').Value;
       MaskCh:=Fields.FieldByName('Mask').Value;
         if MaskCh='enabled-' then Masking:=False;
       Colour1:=Fields.FieldByName('BGC1').Value;
       Colour2:=Fields.FieldByName('BGC2').Value;
```

```
SizerInt:=Fields.FieldByName('SizerInterval').Value;
Overide:=False;
TempS:=Fields.FieldByName('Overide').Value;
if TempS='overide' then Overide:=True;
TempS:=Fields.FieldByName('Random').Value;
If TempS='random' Then RColour:=True
  Else RColour:=False;
TempS:=Fields.FieldByName('PracticeLight').Value;
If TempS='enabled' then PEnabled:=true
  Else PEnabled:=False;
TempS:=Fields.FieldByName('MainLight').Value;
If TempS='enabled' then MEnabled:=true
  Else MEnabled:=False;
TempS:=Fields.FieldByName('Calibrate').Value;
  If TempS='calibrate' Then Cal:=True
    Else Cal:=False;
Ban2:=Fields.FieldByName('Title').Value;
TempS:=Fields.FieldByName('Graphics').Value;
TempS:=Fields.FieldByName('Panel').Value;
If TempS='Panel' Then
  Panel:=True
    Else
      Panel:=False;
CIterations:=Fields.FieldByName('CIterations').Value;
TurnAroundV:=Fields.FieldByName('TurnaroundM').Value;
MaxRuns:=Fields.FieldByName('MaxRuns').Value;
PTiming[1]:=Fields.FieldByName('P1Time').Value;
PTiming[2]:=Fields.FieldByName('P2Time').Value;
PTiming[3]:=Fields.FieldByName('P3Time').Value;
PTiming[4]:=Fields.FieldByName('P4Time').Value;
PTiming[5]:=Fields.FieldByName('P5Time').Value;
MSR:=Fields.FieldByName('FromBGR').Value;
MSG:=Fields.FieldByName('FromBGG').Value;
MSB:=Fields.FieldByName('FromBGB').Value;
MDR:=Fields.FieldByName('ToBGR').Value;
MDG:=Fields.FieldByName('ToBGG').Value;
MDB:=Fields.FieldByName('ToBGB').Value;
MFSR:=Fields.FieldByName('FromFGR').Value;
MFSG:=Fields.FieldByName('FromFGG').Value;
MFSB:=Fields.FieldByName('FromFGB').Value;
MFDR:=Fields.FieldByName('ToFGR').Value;
MFDG:=Fields.FieldByName('ToFGG').Value;
MFDB:=Fields.FieldByName('ToFGB').Value;
Equi:=Fields.FieldByName('Equiliminance').Value;
if Fields.FieldByName('Sizer').Value=1 then
  IsSizer:=True
    Else
      IsSizer:=False;
if Fields.FieldByName('User Calibration').Value=1 then
  UserCal:=True
    Else
      UserCal:=False;  //UserCal,Prescribed
if Fields.FieldByName('Prescribed Solution').Value=1 then
  Prescribed:=True
```

```
    Else
      Prescribed:=False;
SFactor:=Fields.FieldByName('SizerFactor').Value;
TargetWidth:=Fields.FieldByName('TargetWidth').Value;
TargetHeight:=Fields.FieldByName('TargetHeight').Value;
TargetLeft:=Fields.FieldByName('TargetLeft').Value;
TargetTop:=Fields.FieldByName('TargetTop').Value;
S1C1R:=Fields.FieldByName('S1C1R').Value;
S1C1G:=Fields.FieldByName('S1C1G').Value;
S1C1B:=Fields.FieldByName('S1C1B').Value;
S1C2R:=Fields.FieldByName('S1C2R').Value;
S1C2G:=Fields.FieldByName('S1C2G').Value;
S1C2B:=Fields.FieldByName('S1C2B').Value;
T20S1.Appearance.Fill.Color:=RGB(S1C1R,S1C1G,S1C1B);
T20S1.Appearance.Fill.ColorMirror:=RGB(S1C2R,S1C2G,S1C2B);
T18S1.Appearance.Fill.Color:=RGB(S1C1R,S1C1G,S1C1B);
T18S1.Appearance.Fill.ColorMirror:=RGB(S1C2R,S1C2G,S1C2B);
T16S1.Appearance.Fill.Color:=RGB(S1C1R,S1C1G,S1C1B);
T16S1.Appearance.Fill.ColorMirror:=RGB(S1C2R,S1C2G,S1C2B);
T14S1.Appearance.Fill.Color:=RGB(S1C1R,S1C1G,S1C1B);
T14S1.Appearance.Fill.ColorMirror:=RGB(S1C2R,S1C2G,S1C2B);
T12S1.Appearance.Fill.Color:=RGB(S1C1R,S1C1G,S1C1B);
T12S1.Appearance.Fill.ColorMirror:=RGB(S1C2R,S1C2G,S1C2B);
S2C1R:=Fields.FieldByName('S2C1R').Value;
S2C1G:=Fields.FieldByName('S2C1G').Value;
S2C1B:=Fields.FieldByName('S2C1B').Value;
S2C2R:=Fields.FieldByName('S2C2R').Value;
S2C2G:=Fields.FieldByName('S2C2G').Value;
S2C2B:=Fields.FieldByName('S2C2B').Value;
T20S2.Appearance.Fill.Color:=RGB(S2C1R,S2C1G,S2C1B);
T20S2.Appearance.Fill.ColorMirror:=RGB(S2C2R,S2C2G,S2C2B);
T18S2.Appearance.Fill.Color:=RGB(S2C1R,S2C1G,S2C1B);
T18S2.Appearance.Fill.ColorMirror:=RGB(S2C2R,S2C2G,S2C2B);
T16S2.Appearance.Fill.Color:=RGB(S2C1R,S2C1G,S2C1B);
T16S2.Appearance.Fill.ColorMirror:=RGB(S2C2R,S2C2G,S2C2B);
T14S2.Appearance.Fill.Color:=RGB(S2C1R,S2C1G,S2C1B);
T14S2.Appearance.Fill.ColorMirror:=RGB(S2C2R,S2C2G,S2C2B);
T12S2.Appearance.Fill.Color:=RGB(S2C1R,S2C1G,S2C1B);
T12S2.Appearance.Fill.ColorMirror:=RGB(S2C2R,S2C2G,S2C2B);
S3C1R:=Fields.FieldByName('S3C1R').Value;
S3C1G:=Fields.FieldByName('S3C1G').Value;
S3C1B:=Fields.FieldByName('S3C1B').Value;
S3C2R:=Fields.FieldByName('S3C2R').Value;
S3C2G:=Fields.FieldByName('S3C2G').Value;
S3C2B:=Fields.FieldByName('S3C2B').Value;
T20S3.Appearance.Fill.Color:=RGB(S3C1R,S3C1G,S3C1B);
T20S3.Appearance.Fill.ColorMirror:=RGB(S3C2R,S3C2G,S3C2B);
T18S3.Appearance.Fill.Color:=RGB(S3C1R,S3C1G,S3C1B);
T18S3.Appearance.Fill.ColorMirror:=RGB(S3C2R,S3C2G,S3C2B);
T16S3.Appearance.Fill.Color:=RGB(S3C1R,S3C1G,S3C1B);
T16S3.Appearance.Fill.ColorMirror:=RGB(S3C2R,S3C2G,S3C2B);
T14S3.Appearance.Fill.Color:=RGB(S3C1R,S3C1G,S3C1B);
T14S3.Appearance.Fill.ColorMirror:=RGB(S3C2R,S3C2G,S3C2B);
T12S3.Appearance.Fill.Color:=RGB(S3C1R,S3C1G,S3C1B);
```

```
T12S3.Appearance.Fill.ColorMirror:=RGB(S3C2R,S3C2G,S3C2B);
S4C1R:=Fields.FieldByName('S4C1R').Value;
S4C1G:=Fields.FieldByName('S4C1G').Value;
S4C1B:=Fields.FieldByName('S4C1B').Value;
S4C2R:=Fields.FieldByName('S4C2R').Value;
S4C2G:=Fields.FieldByName('S4C2G').Value;
S4C2B:=Fields.FieldByName('S4C2B').Value;
T20S4.Appearance.Fill.Color:=RGB(S4C1R,S4C1G,S4C1B);
T20S4.Appearance.Fill.ColorMirror:=RGB(S4C2R,S4C2G,S4C2B);
T18S4.Appearance.Fill.Color:=RGB(S4C1R,S4C1G,S4C1B);
T18S4.Appearance.Fill.ColorMirror:=RGB(S4C2R,S4C2G,S4C2B);
T16S4.Appearance.Fill.Color:=RGB(S4C1R,S4C1G,S4C1B);
T16S4.Appearance.Fill.ColorMirror:=RGB(S4C2R,S4C2G,S4C2B);
T14S4.Appearance.Fill.Color:=RGB(S4C1R,S4C1G,S4C1B);
T14S4.Appearance.Fill.ColorMirror:=RGB(S4C2R,S4C2G,S4C2B);
T12S4.Appearance.Fill.Color:=RGB(S4C1R,S4C1G,S4C1B);
T12S4.Appearance.Fill.ColorMirror:=RGB(S4C2R,S4C2G,S4C2B);
S5C1R:=Fields.FieldByName('S5C1R').Value;
S5C1G:=Fields.FieldByName('S5C1G').Value;
S5C1B:=Fields.FieldByName('S5C1B').Value;
S5C2R:=Fields.FieldByName('S5C2R').Value;
S5C2G:=Fields.FieldByName('S5C2G').Value;
S5C2B:=Fields.FieldByName('S5C2B').Value;
T20S5.Appearance.Fill.Color:=RGB(S5C1R,S5C1G,S5C1B);
T20S5.Appearance.Fill.ColorMirror:=RGB(S5C2R,S5C2G,S5C2B);
T18S5.Appearance.Fill.Color:=RGB(S5C1R,S5C1G,S5C1B);
T18S5.Appearance.Fill.ColorMirror:=RGB(S5C2R,S5C2G,S5C2B);
T16S5.Appearance.Fill.Color:=RGB(S5C1R,S5C1G,S5C1B);
T16S5.Appearance.Fill.ColorMirror:=RGB(S5C2R,S5C2G,S5C2B);
T14S5.Appearance.Fill.Color:=RGB(S5C1R,S5C1G,S5C1B);
T14S5.Appearance.Fill.ColorMirror:=RGB(S5C2R,S5C2G,S5C2B);
T12S5.Appearance.Fill.Color:=RGB(S5C1R,S5C1G,S5C1B);
T12S5.Appearance.Fill.ColorMirror:=RGB(S5C2R,S5C2G,S5C2B);
S1Threshold:=Fields.FieldByName('S1T').Value;
S2Threshold:=Fields.FieldByName('S2T').Value;
S3Threshold:=Fields.FieldByName('S3T').Value;
S4Threshold:=Fields.FieldByName('S4T').Value;
S5Threshold:=Fields.FieldByName('S5T').Value;
FormR:=Fields.FieldByName('FormR').Value;
FormG:=Fields.FieldByName('FormG').Value;
FormB:=Fields.FieldByName('FormB').Value;
FieldR:=Fields.FieldByName('FieldR').Value;
FieldG:=Fields.FieldByName('FieldG').Value;
FieldB:=Fields.FieldByName('FieldB').Value;
StimulusR:=Fields.FieldByName('StimulusR').Value;
StimulusG:=Fields.FieldByName('StimulusG').Value;
StimulusB:=Fields.FieldByName('StimulusB').Value;
UVM:=Fields.FieldByName('UVM').Value;
LVM:=Fields.FieldByName('LVM').Value;
AVM:=Fields.FieldByName('AVM').Value;
PC1R:=Fields.FieldByName('PC1R').Value;
PC1G:=Fields.FieldByName('PC1G').Value;
PC1B:=Fields.FieldByName('PC1B').Value;
PC2R:=Fields.FieldByName('PC2R').Value;
```

```
      PC2G:=Fields.FieldByName('PC2G').Value;
      PC2B:=Fields.FieldByName('PC2B').Value;
    End;
  Banner2.Caption:=Ban2;
  Banner1.Caption.Text:='Introduction';
  Memo1.Visible:=true;
  S:=TFileStream.Create('c:\cpt\stfiles\Instructions.rtf',fmOpenRead);
  Memo1.Lines.LoadFromStream(S);
  S.Free;
  Memo1.Alignment:=taCenter;
  LoadInstructions.Visible:=false;
  Finished:= False;
  BtnChoosePracticeOrTest.Caption:='Next';
  BtnChoosePracticeOrTest.Visible:=True;
end;

procedure TForm1.BtnGetDetailsPracticeClick(Sender: TObject);
Var Flag,VisionNext : Boolean;
   VRating : Integer;
   Language : Integer;
begin
  Flag:=False;
  VisionNext:=false;
  Gend:='N';
  Age:=0;
  Phase:='CM';
  VRating:=0;
  Language:=0;
  LblLang.Visible:=True;
  FLanguage.Visible:=True;
  FReactions.Visible:=True;
  LblReactions.Visible:=True;
  Gender.Visible:=true;
  RBMale.Visible:=True;
  RBFemale.Visible:=True;
  LblAge.Visible:=True;
  AgeField.Visible:=True;
  Banner1.Visible:=false;
  line2.Visible:=true;
  line2.Font.Color:=clRed;
  Line2.Caption:='Please make a note of your ID and enter some details';
  BtnGetDetailsPractice.Caption:='Continue';
  Namefield.Visible:=true;
//  Namefield.SetFocus;
  Namefield.MaxLength:=9;
  if V1.Checked then
    VRating:=1;
  if V2.Checked then
    VRating:=2;
  if V3.Checked then
    VRating:=3;
  if V4.Checked then
    VRating:=4;
  if V5.Checked then
```

```
  VRating:=5;
if V6.Checked then
  VRating:=6;
if V7.Checked then
  VRating:=7;
if LangIA.Checked then
  Language:=1;
if LangAfrikaans.Checked then
  Language:=2;
if LangEng.checked then
  Language:=3;
if LangEurope.Checked then
  Language:=4;
if LangMe.Checked then
  Language:=5;
if LangHisp.Checked then
  Language:=6;
if LangEast.Checked then
  Language:=7;
NameField.Text:=Login.VNumber;
Namecode:=Login.VNumber;
IF RBFemale.Checked THEN Gend:='F';
IF RBMale.Checked THEN Gend:='M';
Try
   IF (AgeField.Text<>'') THEN
     Age:=StrToInt(AgeField.Text);
     Flag:=True;
    except
      ShowMessage('Wrong value for AGE  '" +AgeField.Text +'"   - use whole numbers only!');
      AgeField.SetFocus;
end;
IF (AgeField.Text='') and (Namefield.Text<>'') Then
Agefield.SetFocus;
IF(Namecode<>'') AND (Length(Login.VNumber)=9) AND (Gend<>'N') AND (Flag=True)
And (Age<>0) and (VRating<>0) And (Language<>0) then
  begin
  VisionNext:=True;
  ADOConnection1.Connected:=True;
  ADONameTable1.Open;
  ADONameTable1.Active:=True;
  ADONameTable1.Locate('ID',Login.VNumber,[]);
    With ADONameTable1 Do
      Begin
        Edit;
        Fields.FieldByName('Gender').Value:=Gend;
        Fields.FieldByName('Age').Value:=Age;
        Fields.FieldByName('Visual Rating').Value:=VRating;
        Fields.FieldByName('First Language').Value:=Language;
        UpDateRecord;
        Post;
      End;
    Banner1.Caption.Text:='Click Vision Test Button';
    LblLang.Visible:=False;
    FLanguage.Visible:=False;
```

```
      FReactions.Visible:=False;
      LblReactions.Visible:=False;
      Line2.Visible:=false;
      Namefield.Visible:=false;
      Gender.Visible:=False;
      RBMale.Visible:=False;
      RBFemale.Visible:=False;
      LblAge.Visible:=False;
      AgeField.Visible:=False;
      FileFunctions.OpenPracticeFile(Finished,Namecode);
      BtnGetDetailsPractice.Visible:=False;
      Form1.Visible:=True;
      BtnColour.Caption:='Vision Test';
      BtnColour.Visible:=True;
      Counter:=0;
      valid:=0;
      errors:=0;
    end;
    BitBtn1.Visible:=false;
    if Not VisionNext then
      Banner1.Caption.Text:='Please Enter Information';
    Banner1.Visible:=True;
end;

procedure TForm1.BtnPInstructionsClick(Sender: TObject);
Var S: TFileStream;
Begin
  BtnPInstructions.Visible:=False;
  Memo1.Clear;
  Memo1.Visible:=true;
  Banner1.Caption.Text:='Colour Calibration';
  S:=TFileStream.Create('c:\cpt\stfiles\PInstructions.rtf',fmOpenRead);
  Memo1.Lines.LoadFromStream(S);
  S.Free;
  btnCalibrateP.Visible:=True;
End;

procedure TForm1.BtnColourClick(Sender: TObject);
begin
  Banner1.Caption.Text:='Colour Vision Test';
  Banner1.Visible:=true;
  ColourMem.Lines.LoadFromFile('c:\CPT\StFiles\ColourBlindIntro.txt');
  ColourMem.Visible:=True;
  BtnDecline.Visible:=True;
  BtnConsent.Visible:=True;
  BtnColour.Visible:=False;
  TextChanged:=False;
end;

procedure TForm1.BtnConfirmClick(Sender: TObject);
begin
  Jog.MinimumValue:=5;
  Stimulus.Visible:=True;
  ChkT3.Checked:=True;
```

```
  CFirst.Interval:=PTiming[3];
  CSecond.Interval:=PTiming[3];
  CheckSolution:=FinalSolution;
  Red:=Sol1[CheckSolution].SR1;
  Green:=Sol1[CheckSolution].SG1;
  Blue:=Sol1[CheckSolution].SB1;
  Red1:=Sol2[CheckSolution].SR2;
  Green1:=Sol2[CheckSolution].SG2;
  Blue1:=Sol2[CheckSolution].SB2;
  CFirst.Enabled:=true;
  BtnConfirm.Visible:=False;
  BtnDisable.Visible:=True;
end;

procedure TForm1.BtnConsentClick(Sender: TObject);
Var S : TFileStream;
begin
  ColourMem.Lines.LoadFromFile('c:\CPT\StFiles\ColourBlindInstructions.txt');
  BtnStartColour.Visible:=True;
  BtnConsent.Visible:=False;
  BtnDecline.Visible:=False;
end;

procedure TForm1.BtnContinueClick(Sender: TObject);
begin
  Banner1.Caption.Text:='Contrast Calibration';
  ContrastInstructions;
  BtnStartTest.ImageIndex:=1;
  BtnContinue.Visible:=False;
end;

Procedure TForm1.ContrastInstructions;
Var S: TFileStream;
Begin
  Memo1.Visible:=true;
  S:=TFileStream.Create('c:\cpt\stfiles\CInstructions.rtf',fmOpenRead);
  Memo1.Lines.LoadFromStream(S);
  S.Free;
  BtnBeginPractice.Visible:=True;
End;

procedure TForm1.CSecondTimer(Sender: TObject);
begin
  Red1:=Sol2[CheckSolution].SR2;
  Green1:=Sol2[CheckSolution].SG2;
  Blue1:=Sol2[CheckSolution].SB2;
  Stimulus.Font.Color:=RGB(Red1,Green1,Blue1);
  CSecond.enabled:=False;
  CFirst.Enabled:=True;
end;

procedure TForm1.BtnDeclineClick(Sender: TObject);
begin
  LblVResult.Visible:=False;
```

```
  LblInstruct.Visible:=False;
  ImageColour.Visible:=False;
  EdColour.Visible:=False;
  BtnVTest.Visible:=false;
  ColourMem.Visible:=false;
  BtnDecline.Visible:=false;
  BtnConsent.Visible:=False;
  BtnStartColour.Visible:=false;
  BtnStartTest.ImageIndex:=1;
  Banner1.Caption.Text:='Contrast Calibration';
  ContrastInstructions;
end;

procedure TForm1.BtnDisableClick(Sender: TObject);
begin
  // background destination colour
 DR:= Sol1[CheckSolution].SR1;   //FDR
 DG:= Sol1[CheckSolution].SG1;   //FDG
 DB:= Sol1[CheckSolution].SB1;   //FDB

 //ForeGround destination colour
 FDR:= Sol2[CheckSolution].SR2;  //DR
 FDG:= Sol2[CheckSolution].SG2;  //DG
 FDB:= Sol2[CheckSolution].SB2;  //DB
 BtnDisable.Visible:=False;
 MatrixPanel.Visible:=False;
 if CFirst.Enabled then
   CFirst.Enabled:=False;
 if CSecond.Enabled then
   CSecond.Enabled:=False;
 Jog.Visible:=False;
 ProcessParvoPics;
 ADOSummaryParvo.TableName:='ParvoCalibration';
 ADOSummaryParvo.Open;
 ADOSummaryParvo.Active:=True;
 With ADOSummaryParvo Do
  Begin
    ADOSummaryParvo.Locate('ID',Login.VNumber,[]);
    Edit;
    Fields.FieldByName('Cal1R').Value:=DR;
    Fields.FieldByName('Cal1G').Value:=DG;
    Fields.FieldByName('Cal1B').Value:=DB;
    Fields.FieldByName('Cal2R').Value:=FDR;
    Fields.FieldByName('Cal2G').Value:=FDG;
    Fields.FieldByName('Cal2B').Value:=FDB;
    UpDateRecord;
    Post;
  End;
 Application.ProcessMessages;
end;

procedure TForm1.FixationTimer(Sender: TObject);
begin
  HowLong:=MStream.Size;
```

```
  Target.Picture.BitMap.LoadFromStream(MStream);
  MStream.Free;
  Fixation.Enabled:=False;
  Timer5.Enabled:=true;
end;

procedure TForm1.FormKeyPress(Sender: TObject; var Key: Char);
begin
if Started then
Begin
  If Not Pressed then
  Begin
    Pressed:=True;
    Key:=Upcase(Key);
    IF (Phase = 'CM') Then ScoreCM;//(Latency,Elatency,Correct,Incorrect,Key,Sizer,Target);
    IF ((Phase = '1') OR (Phase = '2') OR (Phase = '3') OR (Phase = '4')) Then
    Begin
      If Sizer.Enabled Then
        Begin
          Sizer.Enabled:=False;
          Timer2.Enabled:=True;
        End;
      ScoreM;//(Latency,Elatency,Correct,Incorrect,Key,Sizer,Target);
    End;
    IF (Phase='CP') then ScoreCP;
  End;
End;
end;

Procedure TForm1.UpdateCursor;
Begin
  Form1.Cursor:=crDefault;
  Form1.KeyPreview:=False;
End;

// this does the luminance calibration
procedure TForm1.BtnBeginPracticeClick(Sender: TObject);
begin
  PanelCValues.Top:=385;
  PanelCValues.Left:=672;
  BtnStartColour.Visible:=false;
  BtnDecline.Visible:=false;
  BtnVTest.Visible:=false;
  ColourMem.Visible:=False;
  Memo1.Visible:=False;
  Phase:='CM';    //CM
  ProgressBar1.Position:=0;
  Validnumber.Caption:='';
  KeyCounter.Caption:='';
  errornumber.Caption:='';
  Latency.Caption:='';
  Elatency.Caption:='';
  BtnBeginPractice.Visible:=false;
  BtnStartTest.Visible:=false;
```

```
  BtnChoose.Caption:='Start...';
  BtnChoose.ImageIndex:=12;
  Banner1.Caption.Text:='Contrast Calibration';
  BtnChoose.Visible:=true;
  Counter:=0;
  Last:=' ';
  Valid:=0;
  Errors:=0;
  Turnaround:=0;
  PracticeNonTargets:=0;
  RPanel.Visible:=False;
  Form1.Color:=RGB(114,114,114);
  Field.Color:=RGB(114,114,114);
  Stimulus.Color:=RGB(114,114,114);
  Form1.ADOPTable1.TableName:='Lum'+Namecode;
  Form1.ADOPTable1.Open;
  Form1.ADOPTable1.Active:=true;
  Form1.LblR.Visible:=True;
  Form1.LblG.Visible:=True;
  Form1.LblB.Visible:=True;
end;

Procedure Zero;
Begin
  Counter:=0;
  Last:=' ';
  valid:=0;
  errors:=0;
End;

procedure TForm1.BtnStartColourClick(Sender: TObject);
Var VLoop : Integer;
begin
  Vscores[0].Correct:=16;
  Vscores[0].Score:=0;
  Vscores[1].Correct:=2;
  Vscores[1].Score:=0;
  Vscores[2].Correct:=5;
  Vscores[2].Score:=0;
  Vscores[3].Correct:=42;
  Vscores[3].Score:=0;
  Vscores[4].Correct:=7;
  Vscores[4].Score:=0;
  Vscores[5].Correct:=29;
  Vscores[5].Score:=0;
  Vscores[6].Correct:=6;
  Vscores[6].Score:=0;
  Vscores[7].Correct:=57;
  Vscores[7].Score:=0;
  Vscores[8].Correct:=10;
  Vscores[8].Score:=0;
  ColourMem.Visible:=False;
  CList:=TStringList.Create;
  CList.LoadFromFile('c:\CPT\StFiles\CTestList.txt');
```

```
  TextChanged:=False;
  i := CList.Count-1;
  PicNo:=0;
  ImageColour.Picture.LoadFromFile(CList[PicNo]);
  ImageColour.Visible:=true;
  EdColour.Visible:=True;
  BtnStartColour.Visible:=false;
  BtnVTest.Visible:=True;
  EdColour.SetFocus;
end;

// this is to do phase 1 after the practice phase. It uses the same names acquired
// by the practice procedure but the output file name is modified by the procedure
// OpenMainFile so that it is named appropriately
procedure TForm1.BtnStartTestClick(Sender: TObject);
Var CStage : Integer;
begin
   Memo1.Visible:=False;
   MatrixPanel.Visible:=False;
   PanelCValues.Visible:=False;
   if ((Phase<>'1') AND (Phase<>'2') AND (Phase<>'3') AND (Phase<>'4')) then
     Begin
     Phase:='1';
       for CStage:=1 To 4 Do
         Begin
           MScores[CStage].MTotM:=0;
           MScores[CStage].METotM:=0;
           MScores[CStage].MTargetsM:=0;
           MScores[CStage].MNonTargetsM:=0;
           MScores[CStage].MScoreM:=0;
           MScores[CStage].MErrorsM:=0;
           MScores[CStage].ELatencyM:=0;
           MScores[CStage].TLatencyM:=0;
           MScores[CStage].NullErrorsM:=0;
           MScores[CStage].MTotP:=0;
           MScores[CStage].METotP:=0;
           MScores[CStage].MTargetsP:=0;
           MScores[CStage].MNonTargetsP:=0;
           MScores[CStage].MScoreP:=0;
           MScores[CStage].MErrorsP:=0;
           MScores[CStage].ELatencyP:=0;
           MScores[CStage].TLatencyP:=0;
           MScores[CStage].NullErrorsP:=0;
         End;
       Stage:=StrToInt(Phase);
       ADOMTable1.TableName:='Main'+login.VNumber;
       ADOMTable1.Open;
       ADOMTable1.Active:=true;
     End
     Else
     Begin
       Stage:=StrToInt(Phase);
       Stage:=Stage+1;
       Phase:=IntToStr(Stage);
```

```
    End;
   if Stage<5 then
   Begin
    ProgressBar1.Position:=0;
    ProgressBar1.Maximum:=Arraylength;
    Validnumber.Caption:='';
    ValidnumberP.Caption:='';
    NValidM.Caption:='';
    NValidP.Caption:='';
    NullTM.Caption:='';
    NullTP.Caption:='';
    KeyCounter.Caption:='';
    KeyCounterP.Caption:='';
    errornumber.Caption:='';
    errornumberP.Caption:='';
    NullErrorsM.Caption:='';
    NullErrorsP.Caption:='';
    Latency.Caption:='';
    Elatency.Caption:='';
    BtnBeginPractice.Visible:=false;
    FileFunctions.OpenMainFile(Finished,Namecode);
    BtnChoose.Caption:='Block '+Phase;
    Banner1.Caption.Text:='Stage 4: Block '+Phase;
    BtnChoose.ImageIndex:=8;
    BtnChoose.Visible:=true;
    BtnStartTest.Visible:=false;
    Zero;
    RPanel.Visible:=Panel;// Not Panel then
    Finished:=false;
   End;
   if Stage=5 then
   Begin
    BtnStartTest.Visible:=false;
    BtnEndTestStartGraph.Visible:=True;
   End;
   Application.ProcessMessages;
end;

procedure TForm1.BtnTestInstructionsClick(Sender: TObject);
Var S: TFileStream;
Begin
  BtnTestInstructions.Visible:=False;
  Memo1.Clear;
  Memo1.Visible:=true;
  S:=TFileStream.Create('c:\cpt\stfiles\Stage4.rtf',fmOpenRead);
  Memo1.Lines.LoadFromStream(S);
  S.Free;
  BtnStartTest.Caption:='Phase 1';
  BtnStartTest.Visible:=True;
End;

Procedure TForm1.ProcessParvoPics;
Var
 FromFolder,ToFolder,FList : String;
```

```
 AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,AToBB,RR: Integer;
 ABox1,ABox2: Boolean;
Begin
    AFR:=SR;
    AFG:=SG;
    AFB:=SB;
    AToR:=DR;
    AToG:=DG;
    AToB:=DB;
    AFBR:=FSR;
    AFBG:=FSG;
    AFBB:=FSB;
    AToBR:=FDR;
    AToBG:=FDG;
    AToBB:=FDB;
    FromFolder:=GAloc;
    ToFolder:=GPTLoc;
    FList:=BMPALL;
    ABox1:=True;
    ABox2:=True;
    Form1.Cursor:=crHourGlass;
    Form1.Banner1.Caption.Text:='Processing - please wait';
    Thread.ImageThreadAllP.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
    AToBB,ABox1,ABox2,FromFolder,ToFolder,FList);
End;

Procedure TForm1.ConfirmColour;
Begin
  BtnConfirm.Visible:=True;
  ChkT1.Visible:=True;
  ChkT2.Visible:=True;
  ChkT3.Visible:=True;
  ChkT4.Visible:=True;
  ChkT5.Visible:=True;
  Jog.Visible:=True;
  UseConfirm.Visible:=True;
End;

Procedure TForm1.CalcPAverages;
Var Ix,Highest,SolutionIdx,LocalIx,SolutionDx : Integer;
    SL1: Array[1..5] Of Integer;
    SL2: Array[1..5] Of Integer;
    SL3: Array[1..5] Of Integer;
    SL4: Array[1..5] Of Integer;
    SL5: Array[1..5] Of Integer;
    C1R : Array Of Integer;
    C1G : Array Of Integer;
    C1B : Array Of Integer;
    C2R : Array Of Integer;
    C2G : Array Of Integer;
    C2B : Array Of Integer;
    Overall : Array[1..5] Of Integer;
Begin
  Solution1H:=0;
```

```
Solution1Av:=0;
Solution2H:=0;
Solution2Av:=0;
Solution3H:=0;
Solution3Av:=0;
Solution4H:=0;
Solution4Av:=0;
Solution5H:=0;
Solution5Av:=0;
FinalSolution:=0;

SL1[1]:=StrToInt(T20S1.Caption);
SL1[2]:=StrToInt(T18S1.Caption);
SL1[3]:=StrToInt(T16S1.Caption);
SL1[4]:=StrToInt(T14S1.Caption);
SL1[5]:=StrToInt(T12S1.Caption);
Solution1H:=MaxIntValue(SL1);
  if Solution1H>0 then
  Begin
    if Solution1H >=5 then
      Solution1Av:=(SumInt(SL1) Div 5);
    Solution1T:=SumInt(SL1);
    Overall[1]:=Solution1T;
  End;

SL2[1]:=StrToInt(T20S2.Caption);
SL2[2]:=StrToInt(T18S2.Caption);
SL2[3]:=StrToInt(T16S2.Caption);
SL2[4]:=StrToInt(T14S2.Caption);
SL2[5]:=StrToInt(T12S2.Caption);
Solution2H:=MaxIntValue(SL2);
if Solution2H>0 then
  Begin
    if Solution2H >=5 then
      Solution2Av:=(SumInt(SL2) Div 5);
    Solution2T:=SumInt(SL2);
    Overall[2]:=Solution2T;
  End;

SL3[1]:=StrToInt(T20S3.Caption);
SL3[2]:=StrToInt(T18S3.Caption);
SL3[3]:=StrToInt(T16S3.Caption);
SL3[4]:=StrToInt(T14S3.Caption);
SL3[5]:=StrToInt(T12S3.Caption);
Solution3H:=MaxIntValue(SL3);
if Solution3H>0 then
  Begin
    if Solution3H>=5 then
      Solution3Av:=(SumInt(SL3) Div 5);
    Solution3T:=SumInt(SL3);
    Overall[3]:=Solution3T;
  End;

SL4[1]:=StrToInt(T20S4.Caption);
```

```
SL4[2]:=StrToInt(T18S4.Caption);
SL4[3]:=StrToInt(T16S4.Caption);
SL4[4]:=StrToInt(T14S4.Caption);
SL4[5]:=StrToInt(T12S4.Caption);
Solution4H:=MaxIntValue(SL4);
if SOlution4H>0 then
  Begin
    if Solution4H>=5 then
      Solution4Av:=(SumInt(SL4) Div 5);
    Solution4T:=SumInt(SL4);
  Overall[4]:=Solution4T;
  End;

SL5[1]:=StrToInt(T20S5.Caption);
SL5[2]:=StrToInt(T18S5.Caption);
SL5[3]:=StrToInt(T16S5.Caption);
SL5[4]:=StrToInt(T14S5.Caption);
SL5[5]:=StrToInt(T12S5.Caption);
Solution5H:=MaxIntValue(SL5);
if SOlution5H>0  then
  Begin
    if Solution5H>=5 then
      Solution5Av:=(SumInt(SL5) Div 5);
    Solution5T:=SumInt(SL5);
    Overall[5]:=Solution5T;
  End;

if (MaxIntValue(Overall)>0)  then
Begin
  if Overall[1]=MaxIntValue(Overall) then FinalSolution:=1;
  if Overall[2]=MaxIntValue(Overall) then FinalSolution:=2;
  if Overall[3]=MaxIntValue(Overall) then FinalSolution:=3;
  if Overall[4]=MaxIntValue(Overall) then FinalSolution:=4;
  if Overall[5]=MaxIntValue(Overall) then FinalSolution:=5;
  if FinalSolution=1 then Highest:=Solution1H;
  if FinalSolution=2 then Highest:=Solution2H;
  if FinalSolution=3 then Highest:=Solution3H;
  if FinalSolution=4 then Highest:=Solution4H;
  if FinalSolution=5 then Highest:=Solution5H;
End
Else
  FinalSolution:=0;

ChkS1.Visible:=true;
ChkS2.Visible:=true;
ChkS3.Visible:=true;
ChkS4.Visible:=true;
ChkS5.Visible:=true;

if FinalSolution=1 then ChkS1.Checked:=true;
if FinalSolution=2 then ChkS2.Checked:=true;
if FinalSolution=3 then ChkS3.Checked:=true;
if FinalSolution=4 then ChkS4.Checked:=true;
if FinalSolution=5 then ChkS5.Checked:=true;
```

```
 //Calculates all colour solutions
for SolutionDx := 1 to 5 do
  Begin
    SolutionIdx:=0;
    for Ix := 1 to PDataIndx do
      Begin
        if PCalData[Ix].ColourSolution=SolutionDx then
          Begin
            SolutionIdx:=SolutionIdx+1;
          End;
      End;
      if SolutionIdx>0 then
      Begin
        SetLength(C1R,SolutionIdx);
        SetLength(C1G,SolutionIdx);
        SetLength(C1B,SolutionIdx);
        SetLength(C2R,SolutionIdx);
        SetLength(C2G,SolutionIdx);
        SetLength(C2B,SolutionIdx);
        LocalIx:=0;
        for Ix := 1 to PDataIndx do
          Begin
            if PCalData[Ix].ColourSolution=SolutionDx then
              Begin
                Inc(LocalIx);
                C1R[LocalIx-1]:=PCalData[Ix].C1R;
                C1G[LocalIx-1]:=PCalData[Ix].C1G;
                C1B[LocalIx-1]:=PCalData[Ix].C1B;
                C2R[LocalIx-1]:=PCalData[Ix].C2R;
                C2G[LocalIx-1]:=PCalData[Ix].C2G;
                C2B[LocalIx-1]:=PCalData[Ix].C2B;
              End;
          End;
      // background destination colour
          SOl1[SolutionDx].SR1:=(SumInt(C1R) Div SolutionIdx);   //FDR
          SOl1[SolutionDx].SG1:=(SumInt(C1G) Div SolutionIdx);   //FDG
          SOl1[SolutionDx].SB1:=(SumInt(C1B) Div SolutionIdx);   //FDB

          //ForeGround destination colour
          SOl2[SolutionDx].SR2:=(SumInt(C2R) Div SolutionIdx);    //DR
          SOl2[SolutionDx].SG2:=(SumInt(C2G) Div SolutionIdx);    //DG
          SOl2[SolutionDx].SB2:=(SumInt(C2B) Div SolutionIdx);    //DB
      End
      Else
        Begin
          if SolutionDx=1 then
            Begin
              SOl1[SolutionDx].SR1:=S1C1R;
              SOl1[SolutionDx].SG1:=S1C1G;
              SOl1[SolutionDx].SB1:=S1C1B;

              SOl2[SolutionDx].SR2:=S1C2R;
              SOl2[SolutionDx].SG2:=S1C2G;
```

```
          SOl2[SolutionDx].SB2:=S1C2B;
        End;
      if SolutionDx=2 then
        Begin
          SOl1[SolutionDx].SR1:=S2C1R;
          SOl1[SolutionDx].SG1:=S2C1G;
          SOl1[SolutionDx].SB1:=S2C1B;

          SOl2[SolutionDx].SR2:=S2C2R;
          SOl2[SolutionDx].SG2:=S2C2G;
          SOl2[SolutionDx].SB2:=S2C2B;
        End;
      if SolutionDx=3 then
        Begin
          SOl1[SolutionDx].SR1:=S3C1R;
          SOl1[SolutionDx].SG1:=S3C1G;
          SOl1[SolutionDx].SB1:=S3C1B;

          SOl2[SolutionDx].SR2:=S3C2R;
          SOl2[SolutionDx].SG2:=S3C2G;
          SOl2[SolutionDx].SB2:=S3C2B;
        End;
      if SolutionDx=4 then
        Begin
          SOl1[SolutionDx].SR1:=S4C1R;
          SOl1[SolutionDx].SG1:=S4C1G;
          SOl1[SolutionDx].SB1:=S4C1B;

          SOl2[SolutionDx].SR2:=S4C2R;
          SOl2[SolutionDx].SG2:=S4C2G;
          SOl2[SolutionDx].SB2:=S4C2B;
        End;
      if SolutionDx=5 then
        Begin
          SOl1[SolutionDx].SR1:=S5C1R;
          SOl1[SolutionDx].SG1:=S5C1G;
          SOl1[SolutionDx].SB1:=S5C1B;

          SOl2[SolutionDx].SR2:=S5C2R;
          SOl2[SolutionDx].SG2:=S5C2G;
          SOl2[SolutionDx].SB2:=S5C2B;
        End;
      End;
  End;

SolutionIdx:=0;
for Ix := 1 to PDataIndx do
Begin
  if PCalData[Ix].ColourSolution=FinalSolution then
  Begin
    SolutionIdx:=SolutionIdx+1;
  End;
End;
```

```
if SolutionIdx>0 then
Begin
  SetLength(C1R,SolutionIdx);
  SetLength(C1G,SolutionIdx);
  SetLength(C1B,SolutionIdx);
  SetLength(C2R,SolutionIdx);
  SetLength(C2G,SolutionIdx);
  SetLength(C2B,SolutionIdx);
End;

//Calculate averages by putting the FinalSolution values into the temporary array and calculating
LocalIx:=0;
for Ix := 1 to PDataIndx do
Begin
  if PCalData[Ix].ColourSolution=FinalSolution then
  Begin
    Inc(LocalIx);
    C1R[LocalIx-1]:=PCalData[Ix].C1R;
    C1G[LocalIx-1]:=PCalData[Ix].C1G;
    C1B[LocalIx-1]:=PCalData[Ix].C1B;
    C2R[LocalIx-1]:=PCalData[Ix].C2R;
    C2G[LocalIx-1]:=PCalData[Ix].C2G;
    C2B[LocalIx-1]:=PCalData[Ix].C2B;
  End;
End;

 // background from Colour  <=
 SR:=MSR;  //122;
 SG:=MSG;  //122;
 SB:=MSB;  //122;

 if FinalSolution =0 then
   Prescribed:=True;

 if FinalSolution<>0 then
 Begin
   // background destination colour
   Cal1R:=(SumInt(C1R) Div SolutionIdx);   //FDR
   Cal1G:=(SumInt(C1G) Div SolutionIdx);   //FDG
   Cal1B:=(SumInt(C1B) Div SolutionIdx);   //FDB

   //ForeGround destination colour
   Cal2R:=(SumInt(C2R) Div SolutionIdx);    //DR
   Cal2G:=(SumInt(C2G) Div SolutionIdx);    //DG
   Cal2B:=(SumInt(C2B) Div SolutionIdx);    //DB
 End;

 if  Prescribed then
 Begin
   DR:=PC1R;
   DG:=PC1G;
   DB:=PC1B;
   FDR:=PC2R;
   FDG:=PC2G;
```

```
    FDB:=PC2B;
  End
    Else
      Begin
       // background destination colour
       DR:=(SumInt(C1R) Div SolutionIdx);   //FDR
       DG:=(SumInt(C1G) Div SolutionIdx);   //FDG
       DB:=(SumInt(C1B) Div SolutionIdx);   //FDB

       //ForeGround destination colour
       FDR:=(SumInt(C2R) Div SolutionIdx);    //DR
       FDG:=(SumInt(C2G) Div SolutionIdx);    //DG
       FDB:=(SumInt(C2B) Div SolutionIdx);    //DB
      End;

//Foreground threshold Colout >=
FSR:=MFSR;  //122;
FSG:=MFSG;  //122;
FSB:=MFSB; //122;

If UserCal then
Begin
  BtnConfirm.Visible:=True;
  ChkT1.Visible:=True;
  ChkT2.Visible:=True;
  ChkT3.Visible:=True;
  ChkT4.Visible:=True;
  ChkT5.Visible:=True;
  BtnTestInstructions.Visible:=False;
  BtnChoosePracticeOrTest.Visible:=False;
  Jog.Visible:=True;
  UseConfirm.Visible:=True;
End
  Else
    ProcessParvoPics;
  //update the summary table
 ADOSummaryParvo.TableName:='ParvoCalibration';
 ADOSummaryParvo.Open;
 ADOSummaryParvo.Active:=True;
 With ADOSummaryParvo Do
 Begin
   Append;
   Fields.FieldByName('ID').Value:=Login.VNumber;
   Fields.FieldByName('Solution 1Hz').Value:=(1000 Div(Ptiming[1]));
   Fields.FieldByName('Solution 2Hz').Value:=(1000 Div(Ptiming[2]));
   Fields.FieldByName('Solution 3Hz').Value:=(1000 Div(Ptiming[3]));
   Fields.FieldByName('Solution 4Hz').Value:=(1000 Div(Ptiming[4]));
   Fields.FieldByName('Solution 5Hz').Value:=(1000 Div(Ptiming[5]));
   Fields.FieldByName('S1 Hz1 F').Value:=SL1[1];
   Fields.FieldByName('S1 Hz2 F').Value:=SL1[2];
   Fields.FieldByName('S1 Hz3 F').Value:=SL1[3];
   Fields.FieldByName('S1 Hz4 F').Value:=SL1[4];
   Fields.FieldByName('S1 Hz5 F').Value:=SL1[5];
```

```
Fields.FieldByName('S2 Hz1 F').Value:=SL2[1];
Fields.FieldByName('S2 Hz2 F').Value:=SL2[2];
Fields.FieldByName('S2 Hz3 F').Value:=SL2[3];
Fields.FieldByName('S2 Hz4 F').Value:=SL2[4];
Fields.FieldByName('S2 Hz5 F').Value:=SL2[5];

Fields.FieldByName('S3 Hz1 F').Value:=SL3[1];
Fields.FieldByName('S3 Hz2 F').Value:=SL3[2];
Fields.FieldByName('S3 Hz3 F').Value:=SL3[3];
Fields.FieldByName('S3 Hz4 F').Value:=SL3[4];
Fields.FieldByName('S3 Hz5 F').Value:=SL3[5];

Fields.FieldByName('S4 Hz1 F').Value:=SL4[1];
Fields.FieldByName('S4 Hz2 F').Value:=SL4[2];
Fields.FieldByName('S4 Hz3 F').Value:=SL4[3];
Fields.FieldByName('S4 Hz4 F').Value:=SL4[4];
Fields.FieldByName('S4 Hz5 F').Value:=SL4[5];

Fields.FieldByName('S5 Hz1 F').Value:=SL5[1];
Fields.FieldByName('S5 Hz2 F').Value:=SL5[2];
Fields.FieldByName('S5 Hz3 F').Value:=SL5[3];
Fields.FieldByName('S5 Hz4 F').Value:=SL5[4];
Fields.FieldByName('S5 Hz5 F').Value:=SL5[5];

Fields.FieldByName('S1 High').Value:=Solution1H;
Fields.FieldByName('S1 Avg').Value:=Solution1Av;
Fields.FieldByName('S2 High').Value:=Solution2H;
Fields.FieldByName('S2 Avg').Value:=Solution2Av;
Fields.FieldByName('S3 High').Value:=Solution3H;
Fields.FieldByName('S3 Avg').Value:=Solution3Av;
Fields.FieldByName('S4 High').Value:=Solution4H;
Fields.FieldByName('S4 Avg').Value:=Solution4Av;
Fields.FieldByName('S5 High').Value:=Solution5H;
Fields.FieldByName('S5 Avg').Value:=Solution5Av;
Fields.FieldByName('Final Solution').Value:=FinalSolution;
Fields.FieldByName('BG R').Value:=DR;
Fields.FieldByName('BG G').Value:=DG;
Fields.FieldByName('BG B').Value:=DB;
Fields.FieldByName('FG R').Value:=FDR;
Fields.FieldByName('FG G').Value:=FDG;
Fields.FieldByName('FG B').Value:=FDB;
if UserCal then
  Fields.FieldByName('User Calibration').Value:=1
    Else
      Fields.FieldByName('User Calibration').Value:=0;
if Prescribed then
  Fields.FieldByName('Prescribed Solution').Value:=1
    Else
      Fields.FieldByName('Prescribed Solution').Value:=0;
Fields.FieldByName('PC1R').Value:=PC1R;
Fields.FieldByName('PC1G').Value:=PC1G;
Fields.FieldByName('PC1B').Value:=PC1B;
Fields.FieldByName('PC2R').Value:=PC2R;
Fields.FieldByName('PC2G').Value:=PC2G;
```

```
    Fields.FieldByName('PC2B').Value:=PC2B;
    UpdateRecord;
    Post;
  End;
  Application.ProcessMessages;
End;

procedure TForm1.CFirstTimer(Sender: TObject);
begin
  Red:=Sol1[CheckSolution].SR1;
  Green:=Sol1[CheckSolution].SG1;
  Blue:=Sol1[CheckSolution].SB1;
  Stimulus.Font.Color:=RGB(Red,Green,Blue);
  CFirst.enabled:=False;
  CSecond.Enabled:=True;
end;

Procedure TForm1.RunPTimes;
Begin
  PRun:=PRun+1;
  If Prun<=MaxRuns then
  Begin
    Solution:=0;
    First.Interval:=PTiming[PRun];
    Second.Interval:=PTiming[PRun];
    LblFlickerHz.Caption:='Flicker Frequency (Hz): '+IntToStr(1000 Div(Ptiming[PRun]));
    RunP.Enabled:=True;
  End
    Else
    Begin
      LblR.Visible:=False;
      LblG.Visible:=False;
      LblB.Visible:=False;
      LblR1.Visible:=False;
      LblG1.Visible:=False;
      LblB1.Visible:=False;
      CalcPAverages;
      Form1.KeyPreview:=False;
      Started:=false;
    End;
End;

Function TForm1.Position(Button : TAdvSmoothStatusIndicator; Width,Left,First: Integer): Integer;
Begin
  if Button.Width>20 then
    Begin
      Result:=First-((Button.Width-20) Div 2);
    End;
End;

Procedure TForm1.UpdateMatrix;
Var First : Integer;
Begin
  if PRun=1 Then
```

```
  Begin
    S1.Caption:=IntToStr(1000 Div(Ptiming[PRun]));
    First:=49;//17;
    if Solution=1 then
      Begin
        T20S1.Width:=T20S1.Width+1;
        T20S1.Height:=T20S1.Height+1;
        T20S1.Left:=Position(T20S1,T20S1.Width,T20S1.Left,First);
//        T20S1.Left:=T20S1.Left+1;
        T20S1.Top:=T20S1.Top-1;
        T20S1.Caption:=IntToStr(T20S1.Width-20);
      End;
    if Solution=2 then
      Begin
        T20S2.Width:=T20S2.Width+1;
        T20S2.Height:=T20S2.Height+1;
  //      T20S2.Left:=T20S2.Left+1;
        T20S2.Left:=Position(T20S2,T20S2.Width,T20S2.Left,First);
        T20S2.Top:=T20S2.Top-1;
        T20S2.Caption:=IntToStr(T20S2.Width-20);
      End;
    if Solution=3 then
      Begin
        T20S3.Width:=T20S3.Width+1;
        T20S3.Height:=T20S3.Height+1;
   //     T20S3.Left:=T20S3.Left+1;
        T20S3.Left:=Position(T20S3,T20S3.Width,T20S3.Left,First);
        T20S3.Top:=T20S3.Top-1;
        T20S3.Caption:=IntToStr(T20S3.Width-20);
      End;
    if Solution=4 then
      Begin
        T20S4.Width:=T20S4.Width+1;
        T20S4.Height:=T20S4.Height+1;
     //   T20S4.Left:=T20S4.Left+1;
        T20S4.Left:=Position(T20S4,T20S4.Width,T20S4.Left,First);
        T20S4.Top:=T20S4.Top-1;
        T20S4.Caption:=IntToStr(T20S4.Width-20);
      End;
    if Solution=5 then
      Begin
        T20S5.Width:=T20S5.Width+1;
        T20S5.Height:=T20S5.Height+1;
     //   T20S5.Left:=T20S5.Left+1;
        T20S5.Left:=Position(T20S5,T20S5.Width,T20S5.Left,First);
        T20S5.Top:=T20S5.Top-1;
        T20S5.Caption:=IntToStr(T20S5.Width-20);
      End;
  End;
 if PRun=2 Then
  Begin
    First:=94;//62;
    S2.Caption:=IntToStr(1000 Div(Ptiming[PRun]));
    if Solution=1 then
```

```
      Begin
        T18S1.Width:=T18S1.Width+1;
        T18S1.Height:=T18S1.Height+1;
     //   T18S1.Left:=T18S1.Left+1;
        T18S1.Left:=Position(T18S1,T18S1.Width,T18S1.Left,First);
        T18S1.Top:=T18S1.Top-1;
        T18S1.Caption:=IntToStr(T18S1.Width-20);
      End;
    if Solution=2 then
      Begin
        T18S2.Width:=T18S2.Width+1;
        T18S2.Height:=T18S2.Height+1;
     //   T18S2.Left:=T18S2.Left+1;
        T18S2.Left:=Position(T18S2,T18S2.Width,T18S2.Left,First);
        T18S2.Top:=T18S2.Top-1;
        T18S2.Caption:=IntToStr(T18S2.Width-20);
      End;
    if Solution=3 then
      Begin
        T18S3.Width:=T18S3.Width+1;
        T18S3.Height:=T18S3.Height+1;
      //  T18S3.Left:=T18S3.Left+1;
        T18S3.Left:=Position(T18S3,T18S3.Width,T18S3.Left,First);
        T18S3.Top:=T18S3.Top-1;
        T18S3.Caption:=IntToStr(T18S3.Width-20);
      End;
    if Solution=4 then
      Begin
        T18S4.Width:=T18S4.Width+1;
        T18S4.Height:=T18S4.Height+1;
      //  T18S4.Left:=T18S4.Left+1;
        T18S4.Left:=Position(T18S4,T18S4.Width,T18S4.Left,First);
        T18S4.Top:=T18S4.Top-1;
        T18S4.Caption:=IntToStr(T18S4.Width-20);
      End;
    if Solution=5 then
      Begin
        T18S5.Width:=T18S5.Width+1;
        T18S5.Height:=T18S5.Height+1;
      //  T18S5.Left:=T18S5.Left+1;
        T18S5.Left:=Position(T18S5,T18S5.Width,T18S5.Left,First);
        T18S5.Top:=T18S5.Top-1;
        T18S5.Caption:=IntToStr(T18S5.Width-20);
      End;
  End;
if PRun=3 Then
  Begin
    First:=137;//105;
    S3.Caption:=IntToStr(1000 Div(Ptiming[PRun]));
    if Solution=1 then
      Begin
        T16S1.Width:=T16S1.Width+1;
        T16S1.Height:=T16S1.Height+1;
      //  T16S1.Left:=T16S1.Left+1;
```

```
      T16S1.Left:=Position(T16S1,T16S1.Width,T16S1.Left,First);
      T16S1.Top:=T16S1.Top-1;
      T16S1.Caption:=IntToStr(T16S1.Width-20);
     End;
   if Solution=2 then
     Begin
      T16S2.Width:=T16S2.Width+1;
      T16S2.Height:=T16S2.Height+1;
    //  T16S2.Left:=T16S2.Left+1;
      T16S2.Left:=Position(T16S2,T16S2.Width,T16S2.Left,First);
      T16S2.Top:=T16S2.Top-1;
      T16S2.Caption:=IntToStr(T16S2.Width-20);
     End;
   if Solution=3 then
     Begin
      T16S3.Width:=T16S3.Width+1;
      T16S3.Height:=T16S3.Height+1;
     // T16S3.Left:=T16S3.Left+1;
      T16S3.Left:=Position(T16S3,T16S3.Width,T16S3.Left,First);
      T16S3.Top:=T16S3.Top-1;
      T16S3.Caption:=IntToStr(T16S3.Width-20);
     End;
   if Solution=4 then
     Begin
      T16S4.Width:=T16S4.Width+1;
      T16S4.Height:=T16S4.Height+1;
     // T16S4.Left:=T16S4.Left+1;
      T16S4.Left:=Position(T16S4,T16S4.Width,T16S4.Left,First);
      T16S4.Top:=T16S4.Top-1;
      T16S4.Caption:=IntToStr(T16S4.Width-20);
     End;
   if Solution=5 then
     Begin
      T16S5.Width:=T16S5.Width+1;
      T16S5.Height:=T16S5.Height+1;
    //  T16S5.Left:=T16S5.Left+1;
      T16S5.Left:=Position(T16S5,T16S5.Width,T16S5.Left,First);
      T16S5.Top:=T16S5.Top-1;
      T16S5.Caption:=IntToStr(T16S5.Width-20);
     End;
  End;
if PRun=4 Then
  Begin
   First:=184;//152;
   S4.Caption:=IntToStr(1000 Div(Ptiming[PRun]));
   if Solution=1 then
     Begin
      T14S1.Width:=T14S1.Width+1;
      T14S1.Height:=T14S1.Height+1;
    //   T14S1.Left:=T14S1.Left+1;
      T14S1.Left:=Position(T14S1,T14S1.Width,T14S1.Left,First);
      T14S1.Top:=T14S1.Top-1;
      T14S1.Caption:=IntToStr(T14S1.Width-20);
     End;
```

```
    if Solution=2 then
      Begin
        T14S2.Width:=T14S2.Width+1;
        T14S2.Height:=T14S2.Height+1;
      //  T14S2.Left:=T14S2.Left+1;
        T14S2.Left:=Position(T14S2,T14S2.Width,T14S2.Left,First);
        T14S2.Top:=T14S2.Top-1;
        T14S2.Caption:=IntToStr(T14S2.Width-20);
      End;
    if Solution=3 then
      Begin
        T14S3.Width:=T14S3.Width+1;
        T14S3.Height:=T14S3.Height+1;
     //   T14S3.Left:=T14S3.Left+1;
        T14S3.Left:=Position(T14S3,T14S3.Width,T14S3.Left,First);
        T14S3.Top:=T14S3.Top-1;
        T14S3.Caption:=IntToStr(T14S3.Width-20);
      End;
    if Solution=4 then
      Begin
        T14S4.Width:=T14S4.Width+1;
        T14S4.Height:=T14S4.Height+1;
      //  T14S4.Left:=T14S4.Left+1;
        T14S4.Left:=Position(T14S4,T14S4.Width,T14S4.Left,First);
        T14S4.Top:=T14S4.Top-1;
        T14S4.Caption:=IntToStr(T14S4.Width-20);
      End;
    if Solution=5 then
      Begin
        T14S5.Width:=T14S5.Width+1;
        T14S5.Height:=T14S5.Height+1;
      //  T14S5.Left:=T14S5.Left+1;
        T14S5.Left:=Position(T14S5,T14S5.Width,T14S5.Left,First);
        T14S5.Top:=T14S5.Top-1;
        T14S5.Caption:=IntToStr(T14S5.Width-20);
      End;
  End;
if PRun=5 Then
  Begin
     First:=225;//193;
     S5.Caption:=IntToStr(1000 Div(Ptiming[PRun]));
     if Solution=1 then
     Begin
        T12S1.Width:=T12S1.Width+1;
        T12S1.Height:=T12S1.Height+1;
     //   T12S1.Left:=T12S1.Left+-1;
        T12S1.Left:=Position(T12S1,T12S1.Width,T12S1.Left,First);
        T12S1.Top:=T12S1.Top-1;
        T12S1.Caption:=IntToStr(T12S1.Width-20);
      End;
    if Solution=2 then
      Begin
        T12S2.Width:=T12S2.Width+1;
        T12S2.Height:=T12S2.Height+1;
```

```
      //  T12S2.Left:=T12S2.Left+1;
        T12S2.Left:=Position(T12S2,T12S2.Width,T12S2.Left,First);
        T12S2.Top:=T12S2.Top-1;
        T12S2.Caption:=IntToStr(T12S2.Width-20);
      End;
    if Solution=3 then
      Begin
        T12S3.Width:=T12S3.Width+1;
        T12S3.Height:=T12S3.Height+1;
      //  T12S3.Left:=T12S3.Left+1;
        T12S3.Left:=Position(T12S3,T12S3.Width,T12S3.Left,First);
        T12S3.Top:=T12S3.Top-1;
        T12S3.Caption:=IntToStr(T12S3.Width-20);
      End;
    if Solution=4 then
      Begin
        T12S4.Width:=T12S4.Width+1;
        T12S4.Height:=T12S4.Height+1;
     //   T12S4.Left:=T12S4.Left+1;
        T12S4.Left:=Position(T12S4,T12S4.Width,T12S4.Left,First);
        T12S4.Top:=T12S4.Top-1;
        T12S4.Caption:=IntToStr(T12S4.Width-20);
      End;
    if Solution=5 then
      Begin
        T12S5.Width:=T12S5.Width+1;
        T12S5.Height:=T12S5.Height+1;
     //   T12S5.Left:=T12S5.Left+1;
        T12S5.Left:=Position(T12S5,T12S5.Width,T12S5.Left,First);
        T12S5.Top:=T12S5.Top-1;
        T12S5.Caption:=IntToStr(T12S5.Width-20);
      End;
   End;
End;

Procedure TForm1.ScoreCP;
Begin
  LblR.Visible:=True;
  LblG.Visible:=True;
  LblB.Visible:=True;
  LblR1.Visible:=True;
  LblG1.Visible:=True;
  LblB1.Visible:=True;
  Inc(PDataIndx);
  PCalData[PDataInDx].Timing:=PTiming[PRun];
  PCalData[PDataInDx].ColourSolution:=Solution;
  PCalData[PDataInDx].C1R:=Red;
  PCalData[PDataInDx].C1G:=Green;
  PCalData[PDataInDx].C1B:=Blue;
  PCalData[PDataInDx].C2R:=Red1;
  PCalData[PDataInDx].C2G:=Green1;
  PCalData[PDataInDx].C2B:=Blue1;
  PCalData[PDataInDx].KeyPress:=True;
  LblR.Caption:=IntToStr(Red);
```

```
  LblG.Caption:=IntToStr(Green);
  LblB.Caption:=IntToStr(Blue);
  LblR1.Caption:=IntToStr(Red1);
  LblG1.Caption:=IntToStr(Green1);
  LblB1.Caption:=IntToStr(Blue1);
  LblR.Font.Color:=ClBlack;//RGB(Red,Green,Blue);
  LblG.Font.Color:=ClBlack;//RGB(Red,Green,Blue);
  LblB.Font.Color:=ClBlack;//RGB(Red,Green,Blue);
  LblR1.Font.Color:=ClBlack;//RGB(Red1,Green1,Blue1);
  LblG1.Font.Color:=ClBlack;//RGB(Red1,Green1,Blue1);
  LblB1.Font.Color:=ClBlack;//RGB(Red1,Green1,Blue1);
  With ADOPCalTable Do
  Begin
    Append;
    Fields.FieldByName('Timing').Value:=PTiming[PRun];
    Fields.FieldByName('Colour Solution').Value:=Solution;
    Fields.FieldByName('C1 Red').Value:=Red;
    Fields.FieldByName('C1 Green').Value:=Green;
    Fields.FieldByName('C1 Blue').Value:=Blue;
    Fields.FieldByName('C2 Red').Value:=Red1;
    Fields.FieldByName('C2 Green').Value:=Green1;
    Fields.FieldByName('C2 Blue').Value:=Blue1;
      if PCalData[PDataInDx].KeyPress=True then
        Fields.FieldByName('Key Pressed').Value:=1
        Else
         Fields.FieldByName('Key Pressed').Value:=0;
    UpdateRecord;
    Post;
  End;
  Application.ProcessMessages;
  UpdateMatrix;
End;

procedure TForm1.RunPTimer(Sender: TObject);
begin
 Solution:=Solution+1;
 Iterations:=0;
 if Solution=1 then
   Begin
      Red:=S1C1R;
      Green:=S1C1G;
      Blue:=S1C1B;
      Red1:=S1C2R;
      Green1:=S1C2G;
      Blue1:=S1C2B;
      ColourDown:=True;
      ColourUp:=False;
      Iterations:=0;
      Stimulus.Visible:=True;
      First.Enabled:=True;
   End;
    if Solution=2 then
   Begin
      Red:=S2C1R;
```

```
   Green:=S2C1G;
   Blue:=S2C1B;
   Red1:=S2C2R;
   Green1:=S2C2G;
   Blue1:=S2C2B;
   ColourDown:=True;
   ColourUp:=False;
   Iterations:=0;
   Stimulus.Visible:=True;
   First.Enabled:=True;
End;
 if Solution=3 then
Begin
   Red:=S3C1R;
   Green:=S3C1G;
   Blue:=S3C1B;
   Red1:=S3C2R;
   Green1:=S3C2G;
   Blue1:=S3C2B;
   ColourDown:=True;
   ColourUp:=False;
   Iterations:=0;
   Stimulus.Visible:=True;
   First.Enabled:=True;
End;
 if Solution=4 then
Begin
   Red:=S4C1R;
   Green:=S4C1G;
   Blue:=S4C1B;
   Red1:=S4C2R;
   Green1:=S4C2G;
   Blue1:=S4C2B;
   ColourDown:=True;
   ColourUp:=False;
   Iterations:=0;
   Stimulus.Visible:=True;
   First.Enabled:=True;
End;
if Solution=5 then
Begin
   Red:=S5C1R;
   Green:=S5C1G;
   Blue:=S5C1B;
   Red1:=S5C2R;
   Green1:=S5C2G;
   Blue1:=S5C2B;
   ColourDown:=True;
   ColourUp:=False;
   Iterations:=0;
   Stimulus.Visible:=True;
   First.Enabled:=True;
End;
 if Solution>5 Then
```

```
    Begin
      First.Enabled:=False;
      Second.Enabled:=False;
      Cycle.Enabled:=true;
    End;
    RunP.Enabled:=false;
end;

Procedure TForm1.InitialiseMatrix;
Begin
  T20S1.Width:=20;
  T18S1.Width:=20;
  T16S1.Width:=20;
  T14S1.Width:=20;
  T12S1.Width:=20;
  T20S2.Width:=20;
  T18S2.Width:=20;
  T16S2.Width:=20;
  T14S2.Width:=20;
  T12S2.Width:=20;
  T20S3.Width:=20;
  T18S3.Width:=20;
  T16S3.Width:=20;
  T14S3.Width:=20;
  T12S3.Width:=20;
  T20S4.Width:=20;
  T18S4.Width:=20;
  T16S4.Width:=20;
  T14S4.Width:=20;
  T12S4.Width:=20;
  T20S5.Width:=20;
  T18S5.Width:=20;
  T16S5.Width:=20;
  T14S5.Width:=20;
  T12S5.Width:=20;
  T20S1.Caption:='0';
  T18S1.Caption:='0';
  T16S1.Caption:='0';
  T14S1.Caption:='0';
  T12S1.Caption:='0';
  T20S2.Caption:='0';
  T18S2.Caption:='0';
  T16S2.Caption:='0';
  T14S2.Caption:='0';
  T12S2.Caption:='0';
  T20S3.Caption:='0';
  T18S3.Caption:='0';
  T16S3.Caption:='0';
  T14S3.Caption:='0';
  T12S3.Caption:='0';
  T20S4.Caption:='0';
  T18S4.Caption:='0';
  T16S4.Caption:='0';
  T14S4.Caption:='0';
```

```
T12S4.Caption:='0';
T20S5.Caption:='0';
T18S5.Caption:='0';
T16S5.Caption:='0';
T14S5.Caption:='0';
T12S5.Caption:='0';
T20S1.Height:=20;
T18S1.Height:=20;
T16S1.Height:=20;
T14S1.Height:=20;
T12S1.Height:=20;
T20S2.Height:=20;
T18S2.Height:=20;
T16S2.Height:=20;
T14S2.Height:=20;
T12S2.Height:=20;
T20S3.Height:=20;
T18S3.Height:=20;
T16S3.Height:=20;
T14S3.Height:=20;
T12S3.Height:=20;
T20S4.Height:=20;
T18S4.Height:=20;
T16S4.Height:=20;
T14S4.Height:=20;
T12S4.Height:=20;
T20S5.Height:=20;
T18S5.Height:=20;
T16S5.Height:=20;
T14S5.Height:=20;
T12S5.Height:=20;
T20S1.Left:=62;
T18S1.Left:=62;
T16S1.Left:=62;
T14S1.Left:=62;
T12S1.Left:=62;
T20S2.Left:=62;
T18S2.Left:=62;
T16S2.Left:=62;
T14S2.Left:=62;
T12S2.Left:=62;
T20S3.Left:=62;
T18S3.Left:=62;
T16S3.Left:=62;
T14S3.Left:=62;
T12S3.Left:=62;
T20S4.Left:=62;
T18S4.Left:=62;
T16S4.Left:=62;
T14S4.Left:=62;
T12S4.Left:=62;
T20S5.Left:=62;
T18S5.Left:=62;
T16S5.Left:=62;
```

```
  T14S5.Left:=62;
  T12S5.Left:=62;
End;

procedure TForm1.JogValueChanged(Sender: TObject; Value: Double;
  CurrentMode: TAdvSmoothJogWheelModeType);
begin
  if Value<5 Then Value:=5;
  if CheckSolution=3 then
    Sol2[CheckSolution].SG2:=Trunc(Value);
  if CheckSolution=4 then
    Sol1[CheckSolution].SR1:=Trunc(Value);
  if ((CheckSolution=1) OR (CheckSolution=2) OR (CheckSolution=5)) then
      Sol2[CheckSolution].SR2:=Trunc(Value);
end;

procedure TForm1.btnCalibratePClick(Sender: TObject);
Var Init : Integer;
begin
  Memo1.Visible:=False;
  ProgressBar1.Maximum:=(CIterations*25)+1;
  ProgressBar1.Step:=1;
  if Not Logged then
  Begin
    ADOConnection1.ConnectionString:=Login.CString;//    'Provider=MSDASQL.1;Password=douglasjohn;Persist
Security Info=True;User ID=Douglas;Data Source=UCTMSQLE1;Initial Catalog=uctexp1';
    ADOConnection1.Connected:=True;
    Logged:=True;
  End;
  DataBaseFunctions.CreateCalTable(login.VNumber);
  DataBaseFunctions.CreateSumTableParvo;
  PanelCValues.Top:=475;
  PanelCValues.Left:=650;
  //InitialiseMatrix;
  MatrixPanel.Visible:=True;
  if Not Panel then
    MatrixPanel.top:=133;
  PanelCValues.Visible:=Panel;
  LblFlickerHz.Visible:=True;
  Banner1.Caption.Text:='Colour Calibration';
  PRun:=0;
  Phase:='CP';
  PDataIndx:=0;
  for Init := 1 to 1000 do
  Begin
    PCalData[Init].Timing:=0;
    PCalData[Init].ColourSolution:=0;
    PCalData[Init].C1R:=0;
    PCalData[Init].C1G:=0;
    PCalData[Init].C1B:=0;
    PCalData[Init].C2R:=0;
    PCalData[Init].C2G:=0;
    PCalData[Init].C2B:=0;
    PCalData[Init].KeyPress:=False;
```

```
  End;
  BtnCalibrateP.Visible:=False;
  AdoPCalTable.TableName:='ParvoCal'+login.VNumber;
  AdoPCalTable.Open;
  AdoPCalTable.Active:=True;
  Form1.KeyPreview:=true;
//  ScrollBar1.Visible:=True;
  RPanel.Visible:=False;
//  ProgressBar1.Position:=0;
  RunPTimes;
  Started:=true;
end;

Procedure TForm1.ColourCalibrate;
Begin
  Inc(Iterations);
  Pressed:=False;
  if Solution=1 then
    Begin
      if ColourDown then
       Begin
        if Red1>S1C2R-S1Threshold then Red1:=Red1-1;
        if Red1<=S1C2R-S1Threshold then ColourDown:=False;
       End;
       if Not(ColourDown) then
       Begin
         if Red1<S1C2R+S1Threshold then Red1:=Red1+1;
         if Red1=S1C2R+S1Threshold then ColourDown:=True;
       End;
       if Iterations<=CIterations then
         First.Enabled:=True
           Else
             RunP.Enabled:=True;
    End;
  if Solution=2 then
    Begin
      if ColourDown then
       Begin
        if Red1>S2C2R-S2Threshold then Red1:=Red1-1;
        if Red1<=S2C2R-S2Threshold then ColourDown:=False;
       End;
       if Not(ColourDown) then
       Begin
         if Red1<S2C2R+S2Threshold then Red1:=Red1+1;
         if Red1=S2C2R+S2Threshold then ColourDown:=True;
       End;
       if Iterations<=CIterations then
         First.Enabled:=True
           Else
             RunP.Enabled:=True;
    End;
  if Solution=3 then
    Begin
      if ColourDown then
```

```
      Begin
       if Green1>S3C2G-S3Threshold then Green1:=Green1-1;
       if Green1<=S3C2G-S3Threshold then ColourDown:=False;
      End;
      if Not(ColourDown) then
      Begin
        if Green1<S3C2G+S3Threshold then Green1:=Green1+1;
        if Green1=S3C2G+S3Threshold then ColourDown:=True;
      End;
      if Iterations<=CIterations then
        First.Enabled:=True
          Else
            RunP.Enabled:=True;
  End;
 if Solution=4 then
  Begin
    if ColourDown then
      Begin
      if Red>S4C1R-S4Threshold then Red:=Red-1;
       if Red<=S4C1R-S4Threshold then ColourDown:=False;
      End;
      if Not(ColourDown) then
      Begin
        if Red<S4C1R+S4Threshold then Red:=Red+1;
        if Red=S4C1R+S4Threshold then ColourDown:=True;
      End;
      if Iterations<=CIterations then
        First.Enabled:=True
          Else
            RunP.Enabled:=True;
  End;
 if Solution=5 then
  Begin
    if ColourDown then
      Begin
       if Red1>S5C2R-S5Threshold then Red1:=Red1-1;
       if Red1<=S5C2R-S5Threshold then ColourDown:=False;
      End;
      if Not(ColourDown) then
      Begin
        if Red1<S5C2R+S5Threshold then Red1:=Red1+1;
        if Red1=S5C2R+S5Threshold then ColourDown:=True;
      End;
      if Iterations<=CIterations then
        First.Enabled:=True
          Else
            RunP.Enabled:=True;
  End;

End;

procedure TForm1.CycleTimer(Sender: TObject);
begin
 RunPTimes;
```

```
  Cycle.Enabled:=false;
end;

procedure TForm1.ScrollBar1Scroll(Sender: TObject; ScrollCode: TScrollCode;
  var ScrollPos: Integer);
begin
  Stimulus.Color:=RGB(ScrollBar1.Position,ScrollBar1.Position,ScrollBar1.Position);
end;

procedure TForm1.SecondTimer(Sender: TObject);
begin
  Stimulus.Font.Color:=RGB(Red1,Green1,Blue1);
  First.enabled:=True;
  Second.Enabled:=False;
end;

procedure TForm1.FirstTimer(Sender: TObject);
begin
  ColourCalibrate;
  Stimulus.Font.Color:=RGB(Red,Green,Blue);
  First.enabled:=False;
  Second.Enabled:=True;
end;

procedure TForm1.SizerTimer(Sender: TObject);
begin
  if (Target.Width<TargetWidth) OR (Target.Height<TargetHeight)  then
  Begin
    Inc(Iteration,1);
 //   SFactor:=SFactor-Iteration;
    Target.Width:=Target.Width+SFactor;             //+4    +2
    Target.Height:=Target.Height+SFactor;           //+4    +2
    Target.Left:=Target.Left-(SFactor DIV 2);        //-2    -1 Target.Left:=Target.Left-(SFactor DIV 2);
    Target.Top:=Target.Top-(SFactor DIV 2);         //-2    -1  Target.Top:=Target.Top-(SFactor DIV 2);
 //   Target.Height:=Target.Height + Round(1.03* Target.Height);
  End;
  if (Target.Width>=TargetWidth) OR (Target.Width>=TargetHeight)then
    BEgin
      Sizer.Enabled:=false;
      Target.Hide;
      Timer2.Enabled:=True;
    End;
end;

// this enables the main engine 'Buttonnext' after opening the phase 2 input file
// but keeping the previously recorded names and sending the result to the main
// test output file
procedure TForm1.BtnVTestClick(Sender: TObject);
Var Resp,Total,TLoop : Integer;
    Done : Boolean;
begin
  Done:=False;
  if TextChanged then
    Begin
```

```
    if PicNo<=i Then
      Begin
        Resp:=StrToInt(EdColour.Text);
        if Resp=VScores[PicNo].Correct then
          Begin
            VScores[PicNo].Score:=1;
            LblVResult.Font.Color:=ClGreen;
            LblVResult.Caption:='Correct';
          End
           Else
             Begin
              VScores[PicNo].Score:=0;
              LblVResult.Font.Color:=ClRed;
              LblVResult.Caption:='Incorrect '+'you should see '+IntToStr(VScores[PicNo].Correct);
             End;
         if PicNo=i then
          Done:=True
            Else
              Done:=false;
        //   TextChanged:=False;
       End;
  if Not Done then
    Begin
      LblInstruct.Visible:=True;
      EdColour.Visible:=True;
      PicNo:=PicNo+1;
      ImageColour.Picture.LoadFromFile(CList[PicNo]);
      EdColour.SetFocus;
      EdColour.Text:='';
      TextChanged:=False;
      ImageColour.Visible:=true;
    End;
    TextChanged:=False;
  End;
if Done then
  Begin
    Total:=0;
    for TLoop:= 0 to i do
    Begin
      Total:=Total+VScores[TLoop].Score;
    End;
    ADONameTable1.Locate('ID',Login.VNumber,[]);
    With ADONameTable1 Do
      Begin
        Edit;
        Fields.FieldByName('V1').Value:=VScores[0].Score;
        Fields.FieldByName('V2').Value:=VScores[1].Score;
        Fields.FieldByName('V3').Value:=VScores[2].Score;
        Fields.FieldByName('V4').Value:=VScores[3].Score;
        Fields.FieldByName('V5').Value:=VScores[4].Score;
        Fields.FieldByName('V6').Value:=VScores[5].Score;
        Fields.FieldByName('V7').Value:=VScores[6].Score;
        Fields.FieldByName('V8').Value:=VScores[7].Score;
        Fields.FieldByName('V9').Value:=VScores[8].Score;
```

```
        Fields.FieldByName('Colour Blind').Value:=Total;
        UpDateRecord;
        Post;
      End;
      ColourMem.Clear;
      ColourMem.Visible:=true;
      if Total=9 then
        Begin
          ColourMem.Lines.Add('Your colour vision appears to be normal: 9/9 correct.');
          ColourMem.Lines.Add('However, since this is an experimental screening');
          ColourMem.Lines.Add('instrument it may not identify all cases and types ');
          ColourMem.Lines.Add('of colour blindness. If you have concerns about your ');
          ColourMem.Lines.Add('vision, please consult a professional. You may also ');
          ColourMem.Lines.Add('contact any of the research assistants for advice, ');
          ColourMem.Lines.Add('or information.');
        End;
      if (Total <9) AND (Total >=7) then
        ColourMem.Lines.Add('You made some errors: '+IntToStr(Total)+'/9 correct');
      if (Total <7) AND (Total >=5) then
        ColourMem.Lines.Add('You made some errors: '+IntToStr(Total)+'/9 correct');
      if (Total <5) then
        ColourMem.Lines.Add('You made many errors: '+IntToStr(Total)+'/9 correct');
      if Total<9 then
        Begin
          ColourMem.Lines.Add('If you made more than 1 or 2 errors, it is possible that');
          ColourMem.Lines.Add('you have some degree of colour blindness.');
          ColourMem.Lines.Add('If you were not able to see some of the numbers, it');
          ColourMem.Lines.Add('would be advisable to have a proper test for ');
          ColourMem.Lines.Add('colour blindness. However, since this is an ');
          ColourMem.Lines.Add('experimental screening instrumentit may not ');
          ColourMem.Lines.Add('identify all cases and types of colour blindness.');
          ColourMem.Lines.Add('If you have concerns about your vision, please consult');
          ColourMem.Lines.Add('a professional.');
          ColourMem.Lines.Add('You may also contact any of the research assistants ');
          ColourMem.Lines.Add('for advice, or information.');
        End;
        LblVResult.Visible:=False;
        LblInstruct.Visible:=False;
        ImageColour.Visible:=False;
        EdColour.Visible:=False;
        BtnVTest.Visible:=false;
        BtnContinue.Visible:=True;
    End;
    Application.ProcessMessages;
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
  CalcPAverages;
end;

procedure TForm1.BtnEndTestStartGraphClick(Sender: TObject);
begin
  Progressbar1.Visible:=false;
```

```
  RPanel.Visible:=False;
  BtnEndTestStartGraph.Visible:=false;
  BtnShowBarGraph.Visible:=True;
  Banner1.Visible:=true;
  Banner2.Visible:=true;
  Banner2.Alignment:=TaCenter;
  Banner1.Caption.Text:='End of Test';
  Banner2.Caption:='Click Graph for summary';
  BitBtn1.Visible:=true;
  Label1.Visible:=false;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
begin
  Field.Caption:='+';
  Field.Show;
  If Correct.Visible Then Correct.Visible:=false;
  If Incorrect.Visible Then Incorrect.Visible:=false;

  // this is the Calibrate M phase routine
  IF (Phase='CM') AND NOT (Finished) Then
    DisplayCM;

  IF (Phase='CM') and Finished Then
    Begin
    Label1.Caption:='End of brightness calibration proedure';
    End;

  // this is the main test phase routine
 if (Phase<>'CM') then
   Begin
     IF (StrToInt(Phase)<5) AND NOT(Finished) then
     DisplayM(Letter,Last);

     IF (StrToInt(Phase)=5) AND Finished THEN
     Label1.Caption:='End of Testing';
   End;
  Timer1.Enabled:=false;
END;

procedure TForm1.Timer2Timer(Sender: TObject);
begin
  Target.Hide;
  Field.Caption:='';
  Timer3.Enabled:=True;
  Timer2.Enabled:=false;
end;

procedure TForm1.Timer3Timer(Sender: TObject);
begin
  Field.Visible:=True;
  Field.Caption:='+';
  Timer3.Enabled:=false;
  Pressed:=True;
```

```
  Timer1.Enabled:=True;
end;

procedure TForm1.Timer4Timer(Sender: TObject);
begin
Correct.Visible:=False;
InCorrect.Visible:=False;
Timer4.Enabled:=False;
end;

procedure TForm1.Timer5Timer(Sender: TObject);
begin
 if (Phase<> 'CM') And (IsSizer) then
 Begin
  Iteration:=0;
  Field.Hide;
  Pressed:=False;
  Target.Width:=15;
  Target.Height:=15;
  Target.Left:=TargetLeft;//432;
  Target.Top:=TargetTop;//287;
  Target.Show;
  StartClock;
  Ts:=DateTimeToTimeStamp(now);
  Sizer.Enabled:=True;
  Timer5.Enabled:=False;
 End
 Else
  Begin
    Target.Width:=TargetWidth;
    Target.Height:=TargetHeight;
    Target.Left:=TargetLeft-(TargetWidth DIV 2);
    Target.Top:=(TargetTop+45)-(TargetHeight Div 2);
    Iteration:=0;
    Pressed:=False;
    Target.Show;
    StartClock;
    Ts:=DateTimeToTimeStamp(now);
    Timer5.Enabled:=False;
    Timer2.Enabled:=True;
  End;
end;

Begin
FileFunctions.ExtractPFiles;
FileFunctions.ExtractFileNames;
//FLarge,FSmall,FSmallerCM,FBiggerCM : String;
FileFunctions.GetFileNames;
FileFunctions.OpenInPutFiles(LTargets,LNon,LPracticeTargets,LPracNonT);
MainT:=LTargets+LNon;
ArrayLength:=LPracticeTargets+LPracNonT;
SetLength(Pract,ArrayLength+1);
Sorting.SortPractice(LPracticeTargets,LPracNonT,PracTargetArr,PracNonT,Pract);
Setlength(Both,MainT+1);
```

```
Sorting.SortAll(LTargets,LNon,PracTargets,PracNonTargets,Both);
FileFunctions.InitScores;
Logged:=False;
Started:=False;
End.
```

```
unit Unit1;
// Amended 13 May 2012
interface

uses
  System.SysUtils, System.Types, System.UITypes, System.Classes, System.Variants,
  StrUtils, FMX.Controls, FMX.Forms, FMX.Dialogs, FMX.Objects, FMX.Edit,
  FMX.Filter.Effects, FMX.Effects, FMX.Layouts, FMX.ExtCtrls, FMX.ListBox,
  FMX.Ani, FMX.Colors, FMX.Types;

type
  TMyPixelDescriptor = record
    Blue: Byte;
    Green: Byte;
    Red: Byte;
  end;

  PMyPixelArray = ^TMyPixelArray;
  TMyPixelArray = array[0..32767] of TMyPixelDescriptor;
  TMyPictureArray = array of TMyPixelArray;
  TForm1 = class(TForm)
    Image1: TImage;
    Button1: TButton;
    SR: TSpinBox;
    SG: TSpinBox;
    SB: TSpinBox;
    DR: TSpinBox;
    DG: TSpinBox;
    DB: TSpinBox;
    FSR: TSpinBox;
    FSG: TSpinBox;
    FSB: TSpinBox;
    FDR: TSpinBox;
    FDG: TSpinBox;
    FDB: TSpinBox;
    CheckBox1: TCheckBox;
    CheckBox2: TCheckBox;
    Background: TGroupBox;
    ForeGround: TGroupBox;
    BevelEffect1: TBevelEffect;
    BevelEffect2: TBevelEffect;
    BevelEffect3: TBevelEffect;
    Button2: TButton;
    BevelEffect4: TBevelEffect;
    OpenDialog1: TOpenDialog;
    Button3: TButton;
    BevelEffect5: TBevelEffect;
    ImageViewer1: TImageViewer;
    Panel1: TPanel;
    Button4: TButton;
    BevelEffect6: TBevelEffect;
    Expander1: TExpander;
    Button5: TButton;
```

```
BevelEffect7: TBevelEffect;
SaveDialog1: TSaveDialog;
ArcDial1: TArcDial;
GlowEffect1: TGlowEffect;
CheckBox3: TCheckBox;
CheckBox4: TCheckBox;
Timer1: TTimer;
TrackBar1: TTrackBar;
ColorAnimation1: TColorAnimation;
ComboBox1: TComboBox;
ListBoxItem1: TListBoxItem;
ListBoxItem2: TListBoxItem;
ListBoxItem3: TListBoxItem;
ListBoxItem4: TListBoxItem;
ListBoxItem5: TListBoxItem;
ListBoxItem6: TListBoxItem;
ListBoxItem7: TListBoxItem;
ListBoxItem8: TListBoxItem;
ListBoxItem9: TListBoxItem;
ListBoxItem10: TListBoxItem;
ListBoxItem11: TListBoxItem;
ListBoxItem12: TListBoxItem;
Label1: TLabel;
ColorPicker1: TColorPicker;
ColorPicker2: TColorPicker;
Label2: TLabel;
procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Button5Click(Sender: TObject);
procedure Button4Click(Sender: TObject);
procedure ArcDial1Change(Sender: TObject);
procedure ArcDial1DblClick(Sender: TObject);
procedure CheckBox3Click(Sender: TObject);
procedure CheckBox4Click(Sender: TObject);
procedure Timer1Timer(Sender: TObject);
procedure TrackBar1Change(Sender: TObject);
procedure ComboBox1Change(Sender: TObject);
Procedure BlueGreen;
Procedure RedGreen;
Procedure Line;
Procedure MColours;
Procedure DBlueGreen;
Procedure DRedGreen;
Procedure Snake1;
Procedure LRedGreen;
Procedure YellowCyan;
Procedure BrightCyanGreen;
Procedure EscherBlue;
Procedure EscherBlueY;
Procedure ReloadAndChange;
procedure ColorPicker1Click(Sender: TObject);
procedure Expander1Resize(Sender: TObject);
procedure ColorPicker2Click(Sender: TObject);
```

```
  private
    { Private declarations }
  public
     RotateA: Integer;
    { Public declarations }
  end;

var
  Form1: TForm1;
  Opened : Boolean;
  FNam : ShortString;
  Const Folder= 'C:\CPT\Files\ShoeBox\Apple2.bmp';

implementation
Uses Thread;


{$R *.fmx}

procedure TForm1.ArcDial1Change(Sender: TObject);
begin
  ImageViewer1.RotationAngle:=360-ArcDial1.Value;
end;

procedure TForm1.ArcDial1DblClick(Sender: TObject);
begin
  ArcDial1.Value:=0;
end;

procedure TForm1.Button1Click(Sender: TObject);
Var AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,AToBB: Integer;
  ABox1,ABox2: Boolean;
begin
  AFR:=Trunc(Form1.Sr.Value);
  AFG:=Trunc(Form1.Sg.Value);
  AFB:=Trunc(Form1.Sb.Value);
  AToR:=Trunc(Form1.Dr.Value);
  AToG:=Trunc(Form1.Dg.Value);
  AToB:=Trunc(Form1.Db.Value);
  AFBR:=Trunc(Form1.FSR.Value);
  AFBG:=Trunc(Form1.FSG.Value);
  AFBB:=Trunc(Form1.FSB.Value);
  AToBR:=Trunc(Form1.FDR.Value);
  AToBG:=Trunc(Form1.FDG.Value);
  AToBB:=Trunc(Form1.FDB.Value);
  ABox1:=Form1.CheckBox1.IsChecked;
  ABox2:=Form1.CheckBox2.IsChecked;
  ImageThread.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB,ABox1,ABox2,FNam);
//  OnTerminate
//  TFOrm3.Label2.Caption:='Done';
end;
```

```
procedure TForm1.Button2Click(Sender: TObject);
begin
  if Opendialog1.Execute then
    Image1.Bitmap.LoadFromFile(Opendialog1.FileName);
  Form1.ImageViewer1.Bitmap.Assign(Form1.Image1.Bitmap);
  Form1.Image1.Visible:=False;
  ComboBox1.Enabled:=True;
end;


procedure TForm1.Button3Click(Sender: TObject);
begin
  Image1.Bitmap.LoadFromFile(Opendialog1.FileName);
  Form1.ImageViewer1.Bitmap.Assign(Form1.Image1.Bitmap);
  Form1.Image1.Visible:=False;
end;

procedure TForm1.Button4Click(Sender: TObject);
begin
  Form1.Image1.Bitmap.Assign(Form1.ImageViewer1.Bitmap);
  if SaveDialog1.Execute then
    Form1.Image1.Bitmap.SaveToFile(SaveDialog1.FileName);
end;

Procedure TForm1.ReloadAndChange;
Var AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,AToBB: Integer;
  ABox1,ABox2: Boolean;
begin
  Image1.Bitmap.LoadFromFile(Opendialog1.FileName);
  Form1.ImageViewer1.Bitmap.Assign(Form1.Image1.Bitmap);
  Form1.Image1.Visible:=False;
  AFR:=Trunc(Form1.Sr.Value);
  AFG:=Trunc(Form1.Sg.Value);
  AFB:=Trunc(Form1.Sb.Value);
  AToR:=Trunc(Form1.Dr.Value);
  AToG:=Trunc(Form1.Dg.Value);
  AToB:=Trunc(Form1.Db.Value);
  AFBR:=Trunc(Form1.FSR.Value);
  AFBG:=Trunc(Form1.FSG.Value);
  AFBB:=Trunc(Form1.FSB.Value);
  AToBR:=Trunc(Form1.FDR.Value);
  AToBG:=Trunc(Form1.FDG.Value);
  AToBB:=Trunc(Form1.FDB.Value);
  ABox1:=Form1.CheckBox1.IsChecked;
  ABox2:=Form1.CheckBox2.IsChecked;
  ImageThread.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB,ABox1,ABox2,FNam);
end;

procedure TForm1.Button5Click(Sender: TObject);
Var AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,AToBB: Integer;
  ABox1,ABox2: Boolean;
begin
```

```
  Image1.Bitmap.LoadFromFile(Opendialog1.FileName);
  Form1.ImageViewer1.Bitmap.Assign(Form1.Image1.Bitmap);
  Form1.Image1.Visible:=False;
  AFR:=Trunc(Form1.Sr.Value);
  AFG:=Trunc(Form1.Sg.Value);
  AFB:=Trunc(Form1.Sb.Value);
  AToR:=Trunc(Form1.Dr.Value);
  AToG:=Trunc(Form1.Dg.Value);
  AToB:=Trunc(Form1.Db.Value);
  AFBR:=Trunc(Form1.FSR.Value);
  AFBG:=Trunc(Form1.FSG.Value);
  AFBB:=Trunc(Form1.FSB.Value);
  AToBR:=Trunc(Form1.FDR.Value);
  AToBG:=Trunc(Form1.FDG.Value);
  AToBB:=Trunc(Form1.FDB.Value);
  ABox1:=Form1.CheckBox1.IsChecked;
  ABox2:=Form1.CheckBox2.IsChecked;
  ImageThread.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB,ABox1,ABox2,FNam);
end;

procedure TForm1.CheckBox3Click(Sender: TObject);
begin
 if CheckBox4.IsChecked then
    CheckBox4.IsChecked:=False;
    RotateA:=Round(TrackBar1.Value);
    Timer1.Enabled:=True;
end;

procedure TForm1.CheckBox4Click(Sender: TObject);
begin
 if CheckBox3.IsChecked then
    CheckBox3.IsChecked:=False;
    RotateA:=Round(TrackBar1.Value);
    Timer1.Enabled:=True;
end;

Procedure TForm1.BlueGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=150;
 DB.Value:=0;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=0;
 FDG.Value:=150;
 FDB.Value:=70;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
```

```
End;

Procedure TForm1.RedGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=141;
 DB.Value:=0;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=150;
 FDG.Value:=0;
 FDB.Value:=0;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.Line;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=10;
 DG.Value:=10;
 DB.Value:=10;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=20;
 FDG.Value:=20;
 FDB.Value:=20;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.MColours;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=38;//119;
 DG.Value:=38;//119;
 DB.Value:=38;//119;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=31;//133;
 FDG.Value:=31;//133;
 FDB.Value:=31;//133;
```

```
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.DBlueGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=200;
 DB.Value:=0;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=0;
 FDG.Value:=200;
 FDB.Value:=100;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.DRedGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=10;
 DB.Value:=0;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=20;
 FDG.Value:=0;
 FDB.Value:=0;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.Snake1;
Begin
 FSR.Value:=190;
 FSG.Value:=190;
 FSB.Value:=190;
 FDR.Value:=215;
 FDG.Value:=215;
 FDB.Value:=0;
 SR.Value:=86;
 SG.Value:=86;
 SB.Value:=86;
```

```
 DR.Value:=0;
 DG.Value:=99;
 DB.Value:=255;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.LRedGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=200;
 DB.Value:=0;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=255;
 FDG.Value:=0;
 FDB.Value:=0;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.YellowCyan;
Begin
 SR.Value:=0;
 SG.Value:=0;
 SB.Value:=0;
 DR.Value:=0;
 DG.Value:=255;
 DB.Value:=255;
 FSR.Value:=255;
 FSG.Value:=255;
 FSB.Value:=255;
 FDR.Value:=255;
 FDG.Value:=255;
 FDB.Value:=0;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.BrightCyanGreen;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=0;
 DG.Value:=255;
 DB.Value:=0;
```

```
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=0;
 FDG.Value:=254;
 FDB.Value:=255;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.EscherBlue;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=100;
 DG.Value:=183;
 DB.Value:=230;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=95;
 FDG.Value:=121;
 FDB.Value:=185;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

Procedure TForm1.EscherBlueY;
Begin
 SR.Value:=255;
 SG.Value:=255;
 SB.Value:=255;
 DR.Value:=90;
 DG.Value:=165;
 DB.Value:=220;
 FSR.Value:=0;
 FSG.Value:=0;
 FSB.Value:=0;
 FDR.Value:=115;
 FDG.Value:=165;
 FDB.Value:=137;
 CheckBox1.Ischecked:=True;
 CheckBox2.Ischecked:=True;
 ReloadAndChange;
End;

procedure TForm1.Expander1Resize(Sender: TObject);
begin
  if Not(Expander1.IsExpanded) then
    ComboBox1.Visible:=True
    Else
```

```
     ComboBox1.Visible:=False;
end;

procedure TForm1.ColorPicker1Click(Sender: TObject);
Var NewColor: Integer;
   RR,GG,BB : Byte;
   CStr,RS,GS,BS : ShortString;
begin
   CheckBox1.IsChecked:=false;
   NewColor:=ColorPicker1.Color;
   CStr:=IntToHex(NewColor,8);
   BS:=MidStr(CStr,7,2);
   GS:=MidStr(CStr,5,2);
   RS:=MidStr(CStr,3,2);
   Label1.Text:='R '+RS+' G '+GS+' B '+BS;
   DR.Value:=StrToInt('$'+RS);
   DG.Value:=StrToInt('$'+GS);
   DB.Value:=StrToInt('$'+BS);
   CheckBox1.IsChecked:=true;
end;

procedure TForm1.ColorPicker2Click(Sender: TObject);
Var NewColor: Integer;
   RR,GG,BB : Byte;
   CStr,RS,GS,BS : ShortString;
begin
   CheckBox2.IsChecked:=false;
   NewColor:=ColorPicker2.Color;
   CStr:=IntToHex(NewColor,8);
   BS:=MidStr(CStr,7,2);
   GS:=MidStr(CStr,5,2);
   RS:=MidStr(CStr,3,2);
   Label2.Text:='R '+RS+' G '+GS+' B '+BS;
   FDR.Value:=StrToInt('$'+RS);
   FDG.Value:=StrToInt('$'+GS);
   FDB.Value:=StrToInt('$'+BS);
   CheckBox2.IsChecked:=true;
end;

procedure TForm1.ComboBox1Change(Sender: TObject);
begin
  CheckBox1.IsChecked:=false;
  CheckBox2.IsChecked:=false;
  if ComboBox1.ItemIndex=0 then  BlueGreen;
  if ComboBox1.ItemIndex=1 then  RedGreen;
  if ComboBox1.ItemIndex=2 then  Line;
  if ComboBox1.ItemIndex=3 then  MColours;
  if ComboBox1.ItemIndex=4 then  DBlueGreen;
  if ComboBox1.ItemIndex=5 then  DRedGreen;
  if ComboBox1.ItemIndex=6 then  Snake1;
  if ComboBox1.ItemIndex=7 then  LRedGreen;
  if ComboBox1.ItemIndex=8 then  YellowCyan;
  if ComboBox1.ItemIndex=9 then  BrightCyanGreen;
  if ComboBox1.ItemIndex=10 then  EscherBlue;
```

```
  if ComboBox1.ItemIndex=11 then  EscherBlueY;
end;



procedure TForm1.Timer1Timer(Sender: TObject);
begin
  if CheckBox3.IsChecked then
    Begin
      ArcDial1.Value:=ArcDial1.Value-RotateA;
    End;
  if CheckBox4.IsChecked then
    Begin
      ArcDial1.Value:=ArcDial1.Value+RotateA;
    End;
end;

procedure TForm1.TrackBar1Change(Sender: TObject);
begin
  RotateA:=Round(TrackBar1.Value);
end;

end.
```

```
unit IATWinFormUnit;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Buttons, jpeg, ExtCtrls, ComCtrls, DB, ADODB, DBCtrls,
  ImgList;

type
  TTWinForm1 = class(TForm)
    AScreen: TButton;
    AScreen2: TButton;
    BkClose: TBitBtn;
    Code: TLabel;
    Final: TButton;
    Finished: TButton;
    GMask: TImage;
    GPrime: TImage;
    IAT1: TButton;
    IAT2: TButton;
    IatStart: TButton;
    Image2: TImage;
    Label1: TLabel;
    LCon: TLabel;
    Mask: TLabel;
    Name1: TEdit;
    NextCue: TLabel;
    PictureBox1: TImage;
    Prime: TLabel;
    ProgressBar1: TProgressBar;
    RCon: TLabel;
    Results: TButton;
    Stimulus: TLabel;
    T1: TLabel;
    T2: TLabel;
    T3: TLabel;
    T4: TLabel;
    T5: TLabel;
    Timer1: TTimer;
    Timer2: TTimer;
    Timer3: TTimer;
    Title: TLabel;
    Title2: TLabel;
    Trial: TButton;
    Button1: TButton;
    Label2: TLabel;
    Timer4: TTimer;
    Shape1: TShape;
    Timer5: TTimer;
    Memo1: TRichEdit;
    Disclaimer: TMemo;
    AgeField: TEdit;
```

```
LblAge: TLabel;
RBFemale: TRadioButton;
RBMale: TRadioButton;
Gender: TLabel;
FLanguage: TFlowPanel;
LangIA: TRadioButton;
LangAfrikaans: TRadioButton;
LangEng: TRadioButton;
LangEurope: TRadioButton;
LangME: TRadioButton;
LangHisp: TRadioButton;
LangEast: TRadioButton;
LblQ1: TLabel;
ADOConnection1: TADOConnection;
ADOTable1: TADOTable;
BtnInitialise: TButton;
Memo2: TMemo;
Memo3: TMemo;
Memo4: TMemo;
ImageList1: TImageList;
ProgressBar2: TProgressBar;
ADONameTable1: TADOTable;
ADOCommand1: TADOCommand;
ADOIndTable: TADOTable;
ADOSummaryTable: TADOTable; //
procedure AScreenClick(Sender: TObject);
procedure AScreen2Click(Sender: TObject);
procedure IatStartClick(Sender: TObject);
procedure IAT1Click(Sender: TObject);
procedure TrialClick(Sender: TObject);
procedure FormKeyDown(Sender: TObject; var KeyW: Word; Shift: TShiftState);
procedure Timer1Timer(Sender: TObject);
procedure Timer2Timer(Sender: TObject);
procedure Timer3Timer(Sender: TObject);
procedure FinalClick(Sender: TObject);
procedure IAT2Click(Sender: TObject);
procedure ResultsClick(Sender: TObject);
Procedure StartTimer;
Procedure UpdateScreen;
Procedure UpdatePrompt;

Procedure DoFirstBlock124(VAR NextCue,Stimulus,T1,T2,T3,Prime,Mask: Tlabel;
VAR Counter,Score: Integer; VAR Ts:TTimeStamp; VAR Time1,Time2,Total: Longint;
Lcon, Rcon : TLabel; B1,B2,B4 : Boolean; Timer1,Timer2,
Timer3: Ttimer; Progressbar1 : TProgressBar;
Image2: TImage);

Procedure DoBlock124(VAR Stimulus,T1,T2,T3,T4,t5: TLabel;
VAR Counter: Integer; VAR Ts:TTimeStamp; VAR Time1,Time2,Total: Longint;
VAR Key: Char; Var Score: Integer; Timer1: Ttimer;
ProgressBar1 : TProgressBar; Image2 : TImage);

Procedure Trials124(VAR Stimulus,T1,T2,T3,T4,T5,Lcon,Rcon: TLabel;
VAR Counter,Block, Total: Integer; VAR B1: Boolean; Key: Char; VAR Score: Integer;
```

```
   Timer1: Ttimer; ProgressBar1: TProgressBar;
   Image2 : TImage);

   Procedure DoFirstBlock35(VAR Stimulus,Nextcue,T1,T2,T3: TLabel;
   VAR Counter,Score: Integer; VAR Ts:TTimeStamp; VAR Time1,Time2,Total: Longint;
   Lcon, Rcon: TLabel; B1,B2,B4 : Boolean;
   Image2 : TImage;  Timer1,Timer2,Timer3,Timer5: TTimer;
   Progressbar1 : TProgressBar);

   Procedure DoBlock35(VAR Stimulus,Start,T1,T2,T3,T4,t5: TLabel;
   VAR Counter: Integer; VAR Ts:TTimeStamp; VAR Time1,Time2,Total: Longint;
   VAR Key: Char; Var Score: Integer; Image2 : TImage;
   Timer1,Timer2,Timer3 : TTimer;
   Progressbar1 : TProgressbar);

   Procedure Trials35(VAR Stimulus,Start,T1,T2,T3,T4,T5,Lcon,Rcon: TLabel;
   VAR Counter,Block,
   Total: Integer; VAR B1: Boolean; Key: Char; VAR Score: Integer;
   Image2 : TImage; Timer1, Timer2, Timer3 : TTimer; Progressbar1 : TProgressBar);
   procedure Button1Click(Sender: TObject);
   procedure Timer4Timer(Sender: TObject);
   procedure Timer5Timer(Sender: TObject);
   Procedure GetFiles;
   procedure BtnInitialiseClick(Sender: TObject);
   procedure FormClose(Sender: TObject; var Action: TCloseAction);

 private
   { Private declarations }
 public
   { Public declarations }
 end;

 Type
    Results = Record
    ID : String[9];
    Block: Integer;
    Date: String[11];
    Time: String[11];
    Gender: String[1];
    Age : Integer;
    QResponse: Integer;
    Colour: String[1];
    SubTimer1 : String[4];
    SubTimer2 : String[4];
    Counter: Integer;
    Diff : Integer;
    Item : String[20];
    SubItem : String[20];
    Flag: String[1];
    BlockOrder : Integer;
    end;

var
 TWinForm1: TTWinForm1;
```

```
  Q1,Q2,Q3,Q4,Q5,Q1to5, Namecode : STRING;
  Concepts  : TextFile;
  P,Maskfield : String;
  Key: Char;
  Block1,Block2,Block3,Block4,Block5,B1,B2,B3,B4,B5,Pic,Starting,NoKey,
  SPNEB,SPSSEB,SPSPREB,SPSTRSEB,SPSTRREB: Boolean;
  Counter, Block,Score,PrimeW, PrimeB, Block1Order, Block2Order, Block3Order,
  Block4Order, Block5Order,BT1,BT2,BT3,BT4,BT5: Integer;
  TS: TTimeStamp;
  Time1,Time2,Diff,Total:Longint;
  EndTime: Single;
  Gend : ShortString;
  Age,VRating : Integer;
  screens  : String;
  ClockS : Extended;
  ETime : Extended;
  FileLocation: ShortString;
  FileLocationP: ShortString;
  FileLoc: ShortString;
  Parameters,TimerValues: ShortString;
  IndScores: Array[1..140] Of Results;
  GlobalCounter: Integer;

implementation
Uses FileFunctions,Login, COntrolLogic,ArrayFunctions,ReportUnit,ResultUnit,CalcBlocks,DBFunctions;

Procedure StartClock; External
'StopWatch.dll';
Procedure StopClock(Var EndTime: Single); External
'StopWatch.dll';

{$R *.dfm}


Procedure Initialise(VAR Block1,Block2,Block3,Block4,Block5,B1,B2,B3,B4,
B5,Starting: Boolean; Var Counter,Time1,Time2,Score: Integer);
BEGIN
  Block1:=False;
  Block2:=False;
  Block3:=False;
  Block4:=False;
  Block5:=False;
  B1:=False ;
  B2:=False;
  B3:=False;
  B4:=False;
  B5:=False;
  Counter:=0;
  Time1:=0;
  Time2:=0;
  Score:=0;
  Starting:=True;
  GlobalCounter:=0;
END;
```

```
Procedure ChangePrimeColour(Prime : Tlabel);
BEGIN
  IF CCyan THEN Prime.Color:=ClAqua;
  IF NOT (CCyan) THEN Prime.Color:=ClRed;
END;

procedure TTWinForm1.AScreen2Click(Sender: TObject);
Var Flag : Boolean;
begin
  NextCue.Caption:='PRESS ENTER TO CONTINUE';
  NextCue.Visible:=false;
  Memo1.Visible:=false;
  Name1.Visible:=true;
  Title2.Visible:=True;
  Code.Visible:=true;
  Name1.Enabled:=False;
  FLanguage.Visible:=True;
  AgeField.Visible:=True;
  LblAge.Visible:=True;
  LblQ1.Visible:=true;
  Gender.Visible:=true;
  RBMale.Visible:=true;
  RBFemale.Visible:=True;
  Name1.MaxLength:=9;
  Gend:='N';
  Namecode:=Login.VNumber;
  if LangIA.Checked then
    VRating:=1;
  if LangAfrikaans.Checked then
    VRating:=2;
  if LangEng.checked then
    VRating:=3;
  if LangEurope.Checked then
    VRating:=4;
  if LangMe.Checked then
    VRating:=5;
  if LangHisp.Checked then
    VRating:=6;
  if LangEast.Checked then
    VRating:=7;
  if RBMale.Checked then
    Gend:='M';
  if RBFemale.Checked then
    Gend:='F';
    Try
     IF (AgeField.Text<>'') THEN
       Age:=StrToInt(AgeField.Text);
       Flag:=True;
      except
        ShowMessage('Wrong value for AGE  ''' +AgeField.Text +'''   - use whole numbers only!');
        AgeField.SetFocus;
  end;
```

```
   IF (NameCode<>'') AND (Length(Namecode)=9) AND (Gend<>'N') AND (Flag=True)
   And (Age<>0) and (VRating<>0) THEN
     BEGIN
     ADOConnection1.Connected:=True;
     ADONameTable1.Open;
     ADONameTable1.Active:=True;
     ADONameTable1.Locate('ID',Login.VNumber,[]);
     With ADONameTable1 Do
       Begin
         Edit;
         Fields.FieldByName('Gender').Value:=Gend;
         Fields.FieldByName('Age').Value:=Age;
         Fields.FieldByName('QResponse').Value:=VRating;
         UpDateRecord;
         Post;
       End;
       DBFunctions.CreateIndTable(login.VNumber);
       DBFunctions.CreateSummaryTable;
       Name1.Visible:=false;
       Code.Visible:=False;
       FLanguage.Visible:=false;
       LblAge.Visible:=false;
       LblQ1.Visible:=false;
       Gender.Visible:=false;
       RBMale.Visible:=false;
       RBFemale.Visible:=false;
       Ascreen2.Visible:=False;
       AgeField.Visible:=false;
       Nextcue.Caption:='Press ENTER or click "Agree" to continue...';
       Disclaimer.Visible:=True;
       Nextcue.Visible:=true;
       AScreen.Visible:=True;
       AScreen.Caption:='Agree';
       AScreen.SetFocus;
     End;
end;

procedure TTWinForm1.AScreenClick(Sender: TObject);
Var S: TFileStream;
begin
 Disclaimer.Visible:=false;
 S:=TFileStream.Create(SSA,fmOpenRead);
 Memo1.Visible:=True;
 Memo1.Lines.LoadFromStream(S);
 Title.Caption:='';
 Title2.Caption:='';
 S.Free;
 Ascreen.Visible := False;
 IatStart.Visible:=True;
 IatStart.SetFocus;
end;

procedure TTWinForm1.Button1Click(Sender: TObject);
begin
```

```
  Button1.Visible:=FALSE;
  Label2.Visible:=False;
end;

Procedure CheckPicture24(VAR Image2: TImage;
VAR Stimulus: Tlabel);
VAR JoinName : String;
BEGIN
 Stimulus.Caption:='';
 Stimulus.Visible:=False;
 JoinName:=Concat(FileLocation,JoinArray[Counter],'.','bmp');
 Image2.Picture.LoadFromFile(JoinName);
 Image2.Visible:=True;  //true?
END;

Procedure PrimeImage35(GPrime : TImage);
VAR LFilename : String;
BEGIN
IF (B3) THEN
  BEGIN
    IF (IDArray40[Counter]='W') THEN
      BEgin
        LFilename:=Concat(FileLocationP,JoinArray40[Counter],'.bmp');
        GPrime.Picture.LoadFromFile(LFilename);
      End;
    IF (IDArray40[Counter]='B') Then
      BEGIN
        Lfilename:=Concat(FilelocationP,JoinArray40[Counter],'.bmp');
        GPrime.Picture.LoadFromFile(Lfilename);
      End;
  End;
IF (B5) THEN
  BEGIN
    IF (IDArray40[Counter]='W') THEN
      BEgin
        LFilename:=Concat(FileLocationP,JoinArray40[Counter],'.bmp');
        GPrime.Picture.LoadFromFile(LFilename);
      End;
    IF (IDArray40[Counter]='B') Then
      BEGIN
        Lfilename:=Concat(FilelocationP,JoinArray40[Counter],'.bmp');
        GPrime.Picture.LoadFromFile(Lfilename);
      End;
  End;
//  GPrime.Visible:=True;
END;

Procedure FinishTrial124(VAR Stimulus,Lcon,Rcon,T1,T2,T3,T4,T5 : TLabel;
VAR B1,B2,B4: Boolean; Var Score: Integer; VAR IAT2: TButton;
ProgressBar1 : TProgressBar; Image2 : TImage);
VAR Average: Longint; Percent: String; PercentR: Real;
BEGIN
  Progressbar1.Position:=1;
  Progressbar1.Visible:=False;
```

```
 T3.Caption:='';
 Stimulus.Caption:='';
 Stimulus.Visible:=False;
 Image2.Visible:=False;
 Rcon.Visible:=False;
 Lcon.Visible:=False;
 Average:=Total DIV(Counter-1);
 T1.Caption:=IntToStr(Average);
 T1.Visible:=True;
 T2.Visible:=True;
 T4.Visible:=True;
 T5.Visible:=True;
 PercentR:=(Score/20)*100;
 If B1 Then  PercentR:=(Score/(PL+UL))*100;
 IF B2 OR B4 Then PercentR:=(Score/(CL+FL))*100;
 Str(PercentR:3:0,Percent);
 T5.Caption:=Percent;
// finished Block 2 and its not counterbalanced - go to Block 3 or 5
 If B2 AND Not B2Second Then
   Begin
     B2:=False;
     If Not B3Second Then
       Begin
         B3:=True;
         Lcon.Caption:=Conceptarr[5];
         Rcon.Caption:=Conceptarr[6];
       End;
     If B3Second Then
       Begin
         B5:=True;
         Lcon.Caption:=Conceptarr[9];
         Rcon.Caption:=Conceptarr[10];
       End;
   End;

  // Finished Block 2 and it IS counterbalanced - go to Block 3 or 5
  If B2 And B2Second Then
    Begin
      B2 :=False;
      If Not B3Second Then // goto block 5
        Begin
          B5 :=True;
          Lcon.Caption:=Conceptarr[9];
          Rcon.Caption:=Conceptarr[10];
        End;
      If B3Second Then // goto block 3
        Begin
          B3:= True;
          Lcon.Caption:=Conceptarr[5];
          Rcon.Caption:=Conceptarr[6];
        End;
   End;

   // finished block 4 and it is not counterbalanced Goto 3 or 5
```

```
  If B4 and Not B2Second Then
    Begin
      B4 :=False;
      If Not B3Second Then
        Begin
          B5 := True;
          Lcon.Caption:=Conceptarr[9];
          Rcon.Caption:=Conceptarr[10];
        End;
      If B3Second Then
        Begin
          B3:=True;
          Lcon.Caption:=Conceptarr[5];
          Rcon.Caption:=Conceptarr[6];
        End;
    End;

    //Finished Block 4 and it is counterbalanced Goto 3 or 5
    If B4 and B2Second Then
      Begin
        B4:=False;
        If Not B3Second then
          Begin
            B3:=True;
            Lcon.Caption:=Conceptarr[5];
            Rcon.Caption:=Conceptarr[6];
          End;
        If B3Second Then
          Begin
            B5 :=True;
            Lcon.Caption:=Conceptarr[9];
            Rcon.Caption:=Conceptarr[10];
          End;
      End;

    // finished block 1
IF (B1=True) THEN
BEGIN
  B1:=False;
  If B2second Then
    Begin
      B4:=True;
      Stimulus.Caption:='';
      Lcon.Caption:=Conceptarr[7];
      Rcon.Caption:=Conceptarr[8];
    End;
  If Not B2Second Then
    Begin
      B2:=True;
      Lcon.Caption:=Conceptarr[3];
      Rcon.Caption:=Conceptarr[4];
    End;
End;
```

```
  Counter:=0;
  Score:=0;
  IAT2.Visible:=True;
  IAT2.SetFocus;
END;

Procedure SetTimer(Timer1,Timer2,Timer3 : Ttimer);
BEGIN
  Timer2.Interval:=TS2;   //30       300    Exposure for prime word
  Timer3.Interval:=TS3;   //50        50
END;


Procedure TTWinForm1.DoFirstBlock124(VAR NextCue,Stimulus,T1,T2,T3,Prime,Mask: Tlabel;
VAR Counter,Score: Integer; VAR Ts:TTimeStamp; VAR Time1,Time2,Total: Longint;
Lcon, Rcon : TLabel; B1,B2,B4 : Boolean; Timer1,Timer2,
Timer3: Ttimer; Progressbar1 : TProgressBar;
Image2: TImage);
BEGIN
  Stimulus.Visible:=true;
  NextCue.Visible:=false;
  Score:=0;
  Progressbar1.Visible:=True;
  ProgressBar1.Min:=0;
  IF (B1=true) THEN
  BEGIN
    ArrayFunctions.RandomsortP(Pleasant,Pl);
    ArrayFunctions.RandomsortU(Unpleasant,Ul);
    ArrayFunctions.JoinB1;
    Lcon.Caption:=Conceptarr[1];
    Rcon.Caption:=Conceptarr[2];
    ProgressBar1.Max:=(PL+UL);
  END;
  IF (B2=True) THEN
  BEGIN
  ArrayFunctions.RandomsortF(Blacks,Fl);
  ArrayFunctions.RandomsortC(Whites,Cl);
  ArrayFunctions.JoinB2;
  ProgressBar1.Max:=FL+CL;
  END;
  IF (B4=True) THEN
  BEGIN
    RandomsortF(Blacks,Fl);
    RandomsortC(Whites,Cl);
    ArrayFunctions.JoinB4;
    ProgressBar1.Max:=FL+CL;
  END;
  Total:=0;
  Counter:=Counter+1;
  IF (B1) AND(B1Subliminal) THEN
    BEGIN
      Stimulus.visible:=false;
      SetTimer(Timer1,Timer2,Timer3);
      StartTimer;
```

```
   End;
  IF (B1) AND NOT(B1Subliminal) THEN
   BEGIN
     Stimulus.Visible:=True;
     Stimulus.Caption:=Joinarray[Counter];
       StartClock;
   END;
  IF (B2) AND (B2Subliminal) THEN
  BEGIN
     Stimulus.visible:=false;
     SetTimer(Timer1,Timer2,Timer3);
     StartTimer;
   End;
   IF (B2) AND NOT(B2Subliminal)  THEN
   BEGIN
     If B2Text then
       Begin
         Stimulus.Visible:=True;
         Stimulus.Caption:=Joinarray[Counter];
         StartClock;
       End;
     If Not B2Text Then
       Begin
         Image2.Visible:=True;
         CheckPicture24(Image2,Stimulus);
         StartClock;
       End;
   END;

   IF (B4) AND(B4Subliminal) THEN
   BEGIN
     Stimulus.visible:=false;
     SetTimer(Timer1,Timer2,Timer3);
     StartTimer;
   End;

   IF (B4) AND NOT(B4Subliminal) THEN
   BEGIN
     IF B4Text Then
       Begin
         Stimulus.Visible:=True;
         Stimulus.Caption:=Joinarray[Counter];
         StartClock;
       End;
      If Not B2Text Then
       Begin
         Image2.Visible:=True;
         CheckPicture24(Image2,Stimulus);
         StartClock;
       End;
   END;
END;

Procedure Wrong(T3: TLabel);
```

```
BEGIN
  T3.Visible:=True;
  T3.Caption:='X';
END;

Procedure TTWinForm1.DoBlock124(VAR Stimulus,T1,T2,T3,T4,t5: TLabel;
VAR Counter: Integer; VAR Ts:TTimeStamp; VAR Time1,Time2,Total: Longint;
VAR Key: Char; Var Score: Integer; Timer1: Ttimer;
 ProgressBar1 : TProgressBar; Image2 : TImage);
VAR Flag: Integer;
BEGIN
  Diff:=Round(EndTime);
  Label2.Caption:=IntToStr(Diff);
  IF (B1) THEN
    Block1arr[Counter]:=Diff;
  IF (B2) THEN
    Block2arr[Counter]:=Diff;
  IF (B4) THEN
    Block4arr[Counter]:=Diff;
  Total:=Total+Diff;
  Flag:=0;
  IF ControlArray20[Counter]=Key THEN
    Flag:=1;
  ReportUnit.ReportBlocks(B1,B2,B3,B4,B5,Flag);
  Score:=Score+Flag;
  Counter:=Counter+1;
  Progressbar1.StepBy(1);
  if B1 then
    Begin
      IF (COunter<=BT1) THEN
        BEGIN
         Ts:=DateTimeToTimeStamp(now);
         Stimulus.Caption:=Joinarray[Counter];
         StartClock;
        End;
      End;

  if B2 AND (Counter<=BT2) then
  Begin
    IF (B2Subliminal) THEN
      BEGIN
        Stimulus.visible:=false;
        StartTimer;
      End;
    If B2Text AND NOT(B2Subliminal) then
        Begin
          Stimulus.Caption:=Joinarray[Counter];
          StartClock;
        End;
     If Not B2Text then
        Begin
          CheckPicture24(Image2,Stimulus);
          Image2.Visible:=True;
          Ts:=DateTimeToTimeStamp(now);
```

```
        StartClock;
      End;
    End;

  IF (B4) AND (Counter<=BT4) THEN
    BEGIN
     if (B4Subliminal) then
      begin
        Stimulus.visible:=false;
        SetTimer(Timer1,Timer2,Timer3);
        StartTimer;
      End;
     IF NOT(B4Subliminal) THEN
      BEGIN
        If B4Text then
          Begin
            Stimulus.Caption:=Joinarray[Counter];
            StartClock;
          End;
        If Not B4Text then
          Begin
            CheckPicture24(Image2,Stimulus);
            Image2.Visible:=True;
            Ts:=DateTimeToTimeStamp(now);
            StartClock;
          End;
      END;
    END;
END;


// this procedure scores the keypress response
Procedure TTWinForm1.Trials124(VAR Stimulus,T1,T2,T3,T4,T5,Lcon,Rcon: TLabel;
 VAR Counter,Block, Total: Integer; VAR B1: Boolean; Key: Char; VAR Score: Integer;
  Timer1: Ttimer; ProgressBar1: TProgressBar;
  Image2 : TImage);
BEGIN
  T3.Caption:='';
  T3.Visible:=False;
  IF ControlArray20[Counter]<>Key THEN
  BEGIN
  Wrong(T3);
  END;
    IF ControlArray20[Counter]=Key Then
    DOBlock124(Stimulus,T1,T2,T3,T4,T5,Counter,Ts,Time1,Time2,Total,Key,Score,
    Timer1,ProgressBar1,Image2);
END;

Procedure SetTimer35(Timer1,Timer2,Timer3,Timer5 : TTimer);
BEGIN
  Timer2.Interval:=GS2;
  Timer3.Interval:=GS3;
  Timer5.Interval:=DoubleMask;
END;
```

```
Procedure CheckPicture35(VAR Image2: TImage;
VAR Stimulus: TLabel);
VAR JoinName : String;
Begin
    Stimulus.Caption:='';
    Stimulus.Visible:=False;
    IF (IDArray40[Counter]='W')THEN
     BEGIN
      if COR then
       JoinName:=Concat(FileLocationP,JoinArray40[Counter],'.','bmp')
         Else
          JoinName:=Concat(FileLocation,JoinArray40[Counter],'.','bmp')
     End;
    IF (IDArray40[Counter]='B') THEN
     BEGIN
      if COR then
       JoinName:=Concat(FileLocationP,Joinarray40[Counter],'.','bmp')
         Else
          JoinName:=Concat(FileLocation,Joinarray40[Counter],'.','bmp')
     END;
    Image2.Picture.LoadFromFile(JoinName);
    Image2.Visible:=False;
END;


Procedure TTWinForm1.DoFirstBlock35(VAR Stimulus,Nextcue,T1,T2,T3: TLabel;
VAR Counter,Score: Integer; VAR Ts:TTimeStamp; VAR Time1,Time2,Total: Longint;
Lcon, Rcon: TLabel; B1,B2,B4 : Boolean;
Image2 : TImage;  Timer1,Timer2,Timer3,Timer5: TTimer;
 Progressbar1 : TProgressBar);
BEGIN
  Progressbar1.Visible:=True;
  ProgressBar1.Min:=0;
  ProgressBar1.Max:=BT3;
  SetTimer35(Timer1,Timer2,Timer3,Timer5);
  PrimeW:=0;
  PrimeB:=0;
  Score:=0;
  Pic:=True;
  IF (B3=true) THEN
  BEGIN
    RandomsortP(Pleasant,Pl);
    RandomsortU(Unpleasant,Ul);
    RandomsortF(Blacks,Fl);
    RandomsortC(Whites,Cl);
    ArrayFunctions.Join35;
  END;
  IF (B5=True) THEN
  BEGIN
    RandomsortP(Pleasant,Pl);
    RandomsortU(Unpleasant,Ul);
    RandomsortF(Blacks,Fl);
    RandomsortC(Whites,Cl);
```

```
  ArrayFunctions.Join53;
End;
Total:=0;
Counter:=Counter+1;
Stimulus.visible:=false;
NextCue.visible:=False;
IF (IDArray40[Counter]='W') THEN PrimeW:=PrimeW+1;
IF (IDArray40[Counter]='B') THEN PrimeB:=PrimeB+1;
IF (IDArray40[Counter]='P') OR (IDArray40[Counter]='U') OR B3Text Or B5Text Then
   BEGIN
      Pic:=False;
      Stimulus.Visible:=True;
      Stimulus.Caption:=Joinarray40[Counter];
      StartClock;
    // DotStartTiming;
   END;
   IF (B3) AND (B3Subliminal) AND (Pic)  THEN
     BEGIN
     Stimulus.Visible:=False;
     if Not Priming then Image2.Visible:=False;
     if DMask then
     Begin
       GMask.Visible:=True;
       Nokey:=True;
       Timer5.Enabled:=True;
     End;
     If Not Dmask then StartTimer;
     End;
   IF (B5) AND (B5Subliminal) AND (Pic) THEN
     BEGIN
     Stimulus.visible:=false;
     if Not Priming then Image2.Visible:=False;
     if DMask then
     Begin
       GMask.Visible:=True;
       NoKey:=True;
       Timer5.Enabled:=True;
     End;
     If Not DMask then StartTimer;
     End;
  IF (B3) AND NOT(B3Subliminal) AND (Pic)  THEN
  BEGIN
    CheckPicture35(Image2, Stimulus);
    Image2.Visible:=True;
    StartClock;
  // DotStartTiming;
  End;
  IF (B5) AND NOT(B5Subliminal) AND (Pic)  THEN
  BEGIN
    CheckPicture35(Image2, Stimulus);
    Image2.Visible:=True;
    StartClock;
  // DotStartTiming;
  End;
```

```
END;

Procedure TTWinForm1.DoBlock35(VAR Stimulus,Start,T1,T2,T3,T4,t5: TLabel;
VAR Counter: Integer; VAR Ts:TTimeStamp; VAR Time1,Time2,Total: Longint;
VAR Key: Char; Var Score: Integer; Image2 : TImage;
Timer1,Timer2,Timer3 : TTimer;
Progressbar1 : TProgressbar);
VAR Flag: Integer;
BEGIN
 Diff:=Round(EndTime);
  Label2.Caption:=IntToStr(Diff);
  IF (B3) THEN
    Block3Arr[Counter]:=Diff;
  IF (B5) THEN
    Block5Arr[Counter]:=Diff;
  Total:=Total+Diff;
  Flag:=0;
  Pic:=True;
  IF ControlArray40[Counter]=Key THEN
    Flag:=1;
    ReportUnit.ReportBlocks(B1,B2,B3,B4,B5,Flag);
    Score:=Score+Flag;
    Counter:=Counter+1;
    Stimulus.Visible:=false;
  If Counter<=BT3 Then ProgressBar1.StepBy(1);
  IF Counter<=BT3 Then
    Begin
      IF(IDArray40[Counter]='W') THEN PrimeW:=PrimeW+1;
      IF(IDArray40[Counter]='B') THEN PrimeB:=PrimeB+1;
      IF(IDArray40[Counter]='P') OR (IDArray40[Counter]='U') Or B3Text Or B5Text
      THEN Pic:=False;
    End;
    IF (COunter<=BT3) AND (Pic=false) THEN
  BEGIN
    Stimulus.Caption:=Joinarray40[Counter];
    Stimulus.Visible:=True;
    Image2.Visible:=False;
    StartClock;
  END;
  If (B3) AND (Counter<=BT3) AND (Pic=True) AND (B3Subliminal) THEN
  BEGIN
    Stimulus.Visible:=false;
    Image2.Visible:=False;
    if DMask then
    Begin
      GMask.Visible:=True;
      NoKey:=True;
      Timer5.Enabled:=True
    End;
    If Not DMask then StartTimer;
  END;
  If (B5) AND (Counter<BT5) AND (Pic=True) AND (B5Subliminal) THEN
    BEGIN
    Stimulus.Visible:=false;
```

```
      Image2.Visible:=False;
       if DMask then
       Begin
       GMask.Visible:=True;
       NoKey:=True;
       Timer5.Enabled:=True
       End;
       If Not Dmask then StartTimer;
    End;
  IF (B3) AND (Counter<=BT3) AND NOT(B3Subliminal) AND (Pic)  THEN
    BEGIN
      CheckPicture35(Image2, Stimulus);
      Image2.Visible:=True;
      StartClock;
    End;
  IF(B5) AND (Counter<=BT5) AND NOT(B5Subliminal) AND (Pic) THEN
   BEGIN
      CheckPicture35(Image2, Stimulus);
      Image2.Visible:=True;
      StartClock;
    End;
END;


Procedure TTWinForm1.Trials35(VAR Stimulus,Start,T1,T2,T3,T4,T5,Lcon,Rcon: TLabel;
VAR Counter,Block,
Total: Integer; VAR B1: Boolean; Key: Char; VAR Score: Integer;
Image2 : TImage;
 Timer1, Timer2, Timer3 : TTimer;
 Progressbar1 : TProgressBar);
BEGIN
  T3.Caption:='';
  IF ControlArray40[Counter]<>Key THEN
  BEGIN
  Wrong(T3);
  END;
  IF ControlArray40[Counter]=Key Then
    DOBlock35(Stimulus,Start,T1,T2,T3,T4,T5,Counter,Ts,Time1,Time2,Total,Key,Score,
    Image2,Timer1,Timer2,Timer3,Progressbar1);
END;

Procedure FinishTrial35(Stimulus,Lcon,Rcon,T1,T2,T3,T4,T5 : TLabel;
VAR B1,B2,B4: Boolean; Var Score: Integer; VAR IAT2,Final: TButton;
 Image2 : TImage; Progressbar1 : Tprogressbar);
VAR Average: Longint;  Percent: String; PercentR: Real;
BEGIN
  Progressbar1.Position:=1;
  Progressbar1.Visible:=false;
  T3.Caption:='';
  Stimulus.Caption:='';
  Stimulus.Visible:=False;
  Image2.Visible:=False;
  Rcon.Visible:=False;
  Lcon.Visible:=False;
```

```
Average:=Total DIV(Counter-1);
T1.Caption:=IntToStr(Average);
T1.Visible:=True;
T2.Visible:=True;
T4.Visible:=True;
T5.Visible:=True;
PercentR:=(Score/BT3)*100;
Str(PercentR:3:0,Percent);
T5.Caption:=Percent;

If B5 and Not B3Second Then
  Begin
    B5:=False;
    Final.Visible:=True;
    Final.SetFocus;
  End;

 IF B5 And B3Second Then
   Begin
     B5:= False;
     If Not B2Second Then
       Begin
         B4:=True;
         Lcon.Caption:=Conceptarr[7];
         Rcon.Caption:=Conceptarr[8];
         IAT2.Visible:=True;
         IAT2.SetFocus;
       End;
     If B2Second Then
       Begin
         B2 :=True;
         Lcon.Caption:=Conceptarr[3];
         Rcon.Caption:=Conceptarr[4];
         IAT2.Visible:=True;
         IAT2.SetFocus;
       End;
   End;

If B3 and Not B3Second Then
  Begin
    B3 := False;
    If Not B2Second then
      Begin
        B4:=True;
        Lcon.Caption:=Conceptarr[7];
        Rcon.Caption:=Conceptarr[8];
        IAT2.Visible:=True;
        IAT2.SetFocus;
      End;
    If B2Second then
      Begin
        B2:=True;
        Lcon.Caption:=Conceptarr[3];
        Rcon.Caption:=Conceptarr[4];
```

```
        IAT2.Visible:=True;
        IAT2.SetFocus;
       End;
    End;

  IF B3 and B3Second Then
  Begin
    B3 :=False;
    Final.Visible:=True;
    Final.SetFocus;
  End;
  Counter:=0;
  Score:=0;
END;

procedure TTWinForm1.FinalClick(Sender: TObject);
Var S : TFileStream;
begin
  PictureBox1.Visible:=false;
  T3.Visible:=false;
  BkClose.Visible:=False;
  Memo1.Clear;
  Memo1.Visible:=True;
  S:=TFileStream.Create(SS7,fmOpenRead);
  Memo1.Lines.LoadFromStream(S);
  S.Free;
  FileFunctions.JoinOutPutFiles;
  Final.Visible:=False;
  Nextcue.Visible:=False;
  T1.Visible:=False;
  T2.Visible:=False;
  T4.Visible:=False;
  T5.Visible:=False;
  Results.Visible:=True;
  Results.setFocus;
end;

procedure TTWinForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
ADOTable1.Close;
ADOTable1.Active:=false;
ADONameTable1.Close;
ADONameTable1.Active:=false;
ADOIndTable.Close;
ADOIndTable.Active:=false;
ADOSummaryTable.Close;
ADOSummaryTable.Active:=false;
ADOConnection1.Close;
ADOConnection1.Connected:=False;
end;

procedure TTWinForm1.FormKeyDown(Sender: TObject; var KeyW: Word;
  Shift: TShiftState);
begin
```

```
 Key:='#';
  CASE KeyW OF
   80 : Key:='P';
   81 : Key:='Q';
   32 : Key:=' ';
 End;
 If (Starting Or NoKey) Then Key:='#';

 IF (Key='P') OR (Key='Q') OR (Key=' ') THEN
   BEGIN
     If (Key='P') OR (Key='Q') Then
     Begin
     Shape1.Visible:=True;
     Timer4.Enabled:=True;
     End;

     NextCue.Visible:=False;
    IF (Key=' ') and (Counter=0)  and (B1=True) THEN
     BEGIN
       PictureBox1.Visible:=True;
       DoFirstBlock124(NextCue,Stimulus,T1,T2,T3,Prime,Mask,Counter,Score,Ts,Time1,Time2,Total,
       Lcon,Rcon,B1,B2,B4,Timer1,Timer2,Timer3,Progressbar1,Image2);
     END;
   IF ((Key='P') OR (Key='Q'))  AND ((Counter>=1) AND (Counter<=BT1)) AND (B1=True) THEN
     BEGIN
       StopClock(EndTime);
//       EndTime:=Int(ETime);
//       EndTime:=DotUtils.dotTimeElapsed;
       PictureBox1.Visible:=False;
       Trials124(Stimulus,T1,T2,T3,T4,T5,Lcon,Rcon,Counter,Block,Total,B1,Key,
       Score,Timer1,ProgressBar1,Image2);
     END;
   IF ((Key='P') OR (Key='Q'))  AND (Counter>BT1) AND (B1=True) THEN
    BEGIN
       FinishTrial124(Stimulus,Lcon,Rcon,T1,T2,T3,T4,T5,B1,B2,B4,Score,IAT2,
       Progressbar1,Image2);
     END;
   IF (Key=' ') AND (Counter=0) And (B2=True) THEN
     BEGIN
       PictureBox1.Visible:=True;
       DoFirstBlock124(NextCue,Stimulus,T1,T2,T3,Prime,Mask,Counter,Score,Ts,Time1,Time2,Total,Lcon,
       Rcon,B1,B2,B4,Timer1,Timer2,Timer3,Progressbar1,Image2);
     END;
   IF ((Key='P') OR (Key='Q'))  AND (Counter>=1) AND (Counter<=BT2) AND (B2=True) THEN
     BEGIN
       StopClock(EndTime);
//       EndTime:=Int(ETime);
//       EndTime:=DotUtils.dotTimeElapsed;
       Picturebox1.Visible:=false;
       Trials124(Stimulus,T1,T2,T3,T4,T5,Lcon,Rcon,Counter,Block,Total,B1,Key,
       Score,Timer1,ProgressBar1,Image2);
     END;
   IF ((Key='P') OR (Key='Q'))  AND (Counter>BT2) AND (B2=True) THEN
    BEGIN
```

```
      FinishTrial124(Stimulus,Lcon,Rcon,T1,T2,T3,T4,T5,B1,B2,B4,Score,IAT2,
      Progressbar1,Image2);
    END;
    IF (Key=' ') AND (Counter=0) AND (B3=True) THEN
      BEGIN
      Picturebox1.Visible:=true;
        DOFirstBlock35(Stimulus,Nextcue,T1,T2,T3,Counter,Score,Ts,Time1,Time2,Total,Lcon,
        Rcon,B1,B2,B4,Image2,Timer1,Timer2,Timer3,Timer5,Progressbar1);
      END;
    IF ((Key='P') OR (Key='Q'))  AND (Counter>=1) AND (Counter<=BT3) AND (B3=True) THEN
      BEGIN
      StopClock(EndTime);
//      EndTime:=Int(ETime);
//      EndTime:=DotUtils.dotTimeElapsed;
      Picturebox1.Visible:=false;
      Trials35(Stimulus,Nextcue,T1,T2,T3,T4,T5,Lcon,Rcon,Counter,Block,Total,B1,Key,
       Score,Image2, Timer1,Timer2,Timer3,Progressbar1);
      END;
    IF ((Key='P') OR (Key='Q'))  AND (Counter>BT3) AND (B3=True) THEN
      BEGIN
        FinishTrial35(Stimulus,Lcon,Rcon,T1,T2,T3,T4,T5,B1,B2,B4,Score,IAT2,Final,Image2,
        Progressbar1);
      END;
    IF (Key=' ') AND (Counter=0) And (B4=True) THEN
       BEGIN
         PictureBox1.Visible:=True;
         DoFirstBlock124(NextCue,Stimulus,T1,T2,T3,Prime,Mask,Counter,Score,Ts,Time1,Time2,Total,
         Lcon,Rcon,B1,B2,B4,Timer1,Timer2,Timer3,Progressbar1,Image2);
       END;
    IF ((Key='P') OR (Key='Q'))  AND (Counter>=1) AND (Counter<=BT4) AND (B4=True) THEN
       BEGIN
         StopClock(EndTime);
//         EndTime:=Int(ETime);
//         EndTime:=DotUtils.dotTimeElapsed;
         PictureBox1.Visible:=False;
         Trials124(Stimulus,T1,T2,T3,T4,T5,Lcon,Rcon,Counter,Block,Total,B1,Key,
         Score,Timer1,ProgressBar1,Image2);
       END;
    IF ((Key='P') OR (Key='Q'))  AND (Counter>BT4)  AND (B4=True) THEN
      BEGIN
        FinishTrial124(Stimulus,Lcon,Rcon,T1,T2,T3,T4,T5,B1,B2,B4,Score,IAT2,Progressbar1,Image2);
      END;
    IF (Key=' ') AND (Counter=0) AND (B5=True) THEN
      BEGIN
        PictureBox1.Visible:=True;
        DOFirstBlock35(Stimulus,NextCue,T1,T2,T3,Counter,Score,Ts,Time1,Time2,Total,Lcon,
        Rcon,B1,B2,B4,Image2,Timer1,Timer2,Timer3,Timer5,Progressbar1);
      END;
    IF ((Key='P') OR (Key='Q'))  AND (Counter>=1) AND (Counter<=BT5) AND (B5=True) THEN
      BEGIN
        StopClock(EndTime);
//        EndTime:=Int(ETime);
//        EndTime:=DotUtils.dotTimeElapsed;
        PictureBox1.Visible:=False;
```

```
        Trials35(Stimulus,Nextcue,T1,T2,T3,T4,T5,Lcon,Rcon,Counter,Block,Total,B1,
        Key,Score,Image2, Timer1,Timer2,Timer3,Progressbar1);
       END;
     IF ((Key='P') OR (Key='Q'))  AND (Counter>BT5) AND (B5=True) THEN
       BEGIN
        FinishTrial35(Stimulus,Lcon,Rcon,T1,T2,T3,T4,T5,B1,B2,B4,Score,IAT2,Final,Image2,
        Progressbar1);
       END;
  End;
end;

procedure TTWinForm1.IAT1Click(Sender: TObject);
Var S: TFileStream;
begin
// Disclaimer.Visible:=False;
 Nextcue.Visible:=false;
 Memo1.Clear;
 S:=TFileStream.Create(SS2,fmOpenRead);
 Memo1.Visible:=True;
 Memo1.Lines.LoadFromStream(S);
 S.Free;
 IAT1.Visible:=false;
 Trial.Visible:=True;
 Trial.SetFocus;
 LCon.Caption:=ConceptArr[1];
 RCon.Caption:=ConceptArr[2];
end;


procedure TTWinForm1.IAT2Click(Sender: TObject);
var S: TFileStream;
begin
  Picturebox1.Visible:=false;
  T3.Visible:=false;
  Memo1.Clear;
  Memo1.Visible:=true;
  IF B2 Then
  BEGIN
    S:=TFileStream.Create(SS3,fmOpenRead);
    Memo1.Visible:=True;
    Memo1.Lines.LoadFromStream(S);
    S.Free;
  End;
  IF B3 Then
    Begin
      S:=TFileStream.Create(SS4,fmOpenRead);
      Memo1.Visible:=True;
      Memo1.Lines.LoadFromStream(S);
      S.Free;
    End;
  IF B4 Then
  Begin
    S:=TFileStream.Create(SS5,fmOpenRead);
    Memo1.Visible:=True;
```

```
    Memo1.Lines.LoadFromStream(S);
    S.Free;
  End;
  IF B5 Then
  Begin
    S:=TFileStream.Create(SS6,fmOpenRead);
    Memo1.Visible:=True;
    Memo1.Lines.LoadFromStream(S);
    S.Free;
  End;
  IAT2.Visible:=false;
  Trial.Visible:=True;
  Trial.SetFocus;
  Nextcue.Visible:=false;
  T1.Visible:=False;
  T2.Visible:=False;
  T4.Visible:=False;
  T5.Visible:=False;
end;

procedure TTWinForm1.IatStartClick(Sender: TObject);
Var S: TFileStream;
begin
 NextCue.Caption:='Press ENTER or click NEXT to continue...';
 Nextcue.Visible:=false;
 Disclaimer.Visible:=False;
 Memo1.Clear;
 S:=TFileStream.Create(SS1,fmOpenRead);
 Memo1.Visible:=True;
 Memo1.Lines.LoadFromStream(S);
 S.Free;
 IatStart.visible:=False;
 IAT1.Visible:=true;
 IAT1.SetFocus;
end;

procedure TTWinForm1.ResultsClick(Sender: TObject);
Var Block1Avg,Block2Avg,Block3Avg,Block4Avg,Block5Avg,Block1std,Block2Std,
Block3Std,Block4Std,Block5Std,DStat5, DStat4, Block24Std, Block35Std : Double;
DisplayV,Disp1,Disp2,Verdict: String;
Var LP : Integer;
Var EventDate : ShortString;
  begin
  Memo1.Clear;
  Memo1.Lines.Add('END OF TEST, CLICK "CLOSE" TO FINISH TEST');
  FileFunctions.CloseFiles;
  ReportUnit.WriteIndDBFile;
  CalcBlocks.CalcBlock1(Block1Avg,Block1Std);
  CalcBlocks.CalcBlock2(Block2Avg,Block2Std);
  CalcBlocks.CalcBlock3(Block3Avg,Block3Std);
  CalcBlocks.CalcBlock4(Block4Avg,Block4Std);
  CalcBlocks.CalcBlock5(Block5Avg,Block5Std);
  ArrayFunctions.Combine24;
  ArrayFunctions.Combine35;
```

```
CalcBlocks.GetStd24(Block24Std);
CalcBlocks.GetStd35(Block35Std);
Dstat5:=(Block3Avg-Block5Avg)/Block35Std;
DStat4:=(Block2Avg-Block4Avg)/Block24Std;
Disp1:=FloatToStr(Round(Block1Avg));
Disp2:=FloatToStr(Block1Std);
DisplayV:=Concat('Block1 Average: ',Disp1,'    Block 1 Std deviation: ',Disp2);
ResultUnit.Results.Memo1.Clear;
ResultUnit.Results.Memo1.Lines.Append('Results of IAT test as follows:');
ResultUnit.Results.Memo1.Lines.Append(DisplayV);
//display block 2
Disp1:=FloatToStr(Round(Block2Avg));
Disp2:=FloatToStr(Block2Std);
DisplayV:=Concat('Block2 Average: ',Disp1,'    Block 2 Std deviation: ',Disp2);
ResultUnit.Results.Memo1.Lines.Append(DisplayV);
//display block 3
Disp1:=FloatToStr(Round(Block3Avg));
Disp2:=FloatToStr(Block3Std);
DisplayV:=Concat('Block3 Average: ',Disp1,'    Block 3 Std deviation: ',Disp2);
ResultUnit.Results.Memo1.Lines.Append(DisplayV);
//display block 4
Disp1:=FloatToStr(Round(Block4Avg));
Disp2:=FloatToStr(Block4Std);
DisplayV:=Concat('Block4 Average: ',Disp1,'    Block 4 Std deviation: ',Disp2);
ResultUnit.Results.Memo1.Lines.Append(DisplayV);
//display block 5
Disp1:=FloatToStr(Round(Block5Avg));
Disp2:=FloatToStr(Block5Std);
DisplayV:=Concat('Block5 Average: ',Disp1,'    Block 5 Std deviation: ',Disp2);
ResultUnit.Results.Memo1.Lines.Append(DisplayV);
//Display block 2-4 index
Disp1:=FloatToStr(Dstat4);
DisplayV:=Concat('Block2 - Block4 index: ',Disp1);
ResultUnit.Results.Memo1.Lines.Append(DisplayV);
// Display Block 3-5 index
Disp1:=FloatToStr(Dstat5);
DisplayV:=Concat('Block3 - Block5 index: ',Disp1);
ResultUnit.Results.Memo1.Lines.Append(DisplayV);
EventDate:=DateTimeToStr(Now);
With ADOSummaryTable Do
Begin
  ADOSummaryTable.TableName:='summary';
  ADOSummaryTable.Open;
  ADOSummaryTable.Active:=true;
  ADOSummaryTable.Append;
  Fields.FieldByName('ID').Value:=IndScores[1].ID;
  Fields.FieldByName('Date').AsDateTime:=StrToDateTime(EventDate);
  Fields.FieldByName('Age').Value:=IndScores[1].Age;
  Fields.FieldByName('Gender').Value:=IndScores[1].Gender;
  Fields.FieldByName('QResponse').Value:=IndScores[1].QResponse;
  Fields.FieldByName('Colour').Value:=IndScores[1].Colour;
  Fields.FieldByName('SubTimer1').Value:=IndScores[1].SubTimer1;
  Fields.FieldByName('SubTimer2').Value:=IndScores[1].SubTimer2;
  Fields.FieldByName('Block1 Average').Value:=(Round(Block1Avg));
```

```
    Fields.FieldByName('Block1 SD').Value:=(Int(Block1Std));
    Fields.FieldByName('Block2 Average').Value:=(Round(Block2Avg));
    Fields.FieldByName('Block2 SD').Value:=(Int(Block2Std));
    Fields.FieldByName('Block3 Average').Value:=(Round(Block3Avg));
    Fields.FieldByName('Block3 SD').Value:=(Int(Block3Std));
    Fields.FieldByName('Block4 Average').Value:=(Round(Block4Avg));
    Fields.FieldByName('Block4 SD').Value:=(Int(Block4Std));
    Fields.FieldByName('Block5 Average').Value:=(Round(Block5Avg));
    Fields.FieldByName('Block5 SD').Value:=(Int(Block5Std));
    Fields.FieldByName('Block2 4 D').Value:=DStat4;
    Fields.FieldByName('D Score').Value:=DStat5;
    UpDateRecord;
    Post;
  End;
  Begin
    If (Dstat5 <0) And (DStat5 >=-0.3) Then Verdict:=Memo4.Lines[1];//'slight preference for whites.';
    If (Dstat5 <-0.3) And (DStat5 >=-0.5) Then Verdict:=Memo4.Lines[2];//'moderate preference for whites.';
    If (Dstat5 <-0.5) And (DStat5 >=-0.75) Then Verdict:=Memo4.Lines[3];//'strong preference for whites.';
    If (Dstat5 <-0.75) Then Verdict:=Memo4.Lines[4];//'very strong preference for whites.';
    If (Dstat5 >0) And (DStat5 <=0.3) Then Verdict:=Memo4.Lines[5];//'slight preference for blacks.';
    If (Dstat5 >0.3) And (DStat5 <=0.5) Then Verdict:=Memo4.Lines[6];//'moderate preference for blacks.';
    If (Dstat5 >0.5) And (DStat5 <=0.75) Then Verdict:=Memo4.Lines[7];//'strong preference for blacks.';
    If (Dstat5 >0.75) Then Verdict:=Memo4.Lines[8];//'very strong preference for blacks.';
    If ((Dstat5<0) AND (DStat5>-0.3)) OR ((Dstat5>0) AND (DStat5<0.3)) Then Verdict:=Memo4.Lines[0];// 'neutral
attitude towards white/black people.'; ?
  End;
  Cursor:=crdefault;
  NextCue.Caption:='Success! CLICK "CLOSE" TO FINISH TEST';
  DisplayV:=Concat('Your result suggests a ',Verdict);
  ResultUnit.Results.Memo1.Lines.Append(DisplayV);
  ResultUnit.Results.Memo1.Visible:=True;
  ResultUnit.Results.Visible:=True;
  ResultUnit.Results.Chart1.Visible:=True;
  ResultUnit.Results.Series1.Clear;
  ResultUnit.Results.Series1.AddBar(Block1Avg,'Block 1',788536);
  ResultUnit.Results.Series1.AddBar(Block2Avg,'Block 2',8745662);
  ResultUnit.Results.Series1.AddBar(Block3Avg,'Block 3',45668);
  ResultUnit.Results.Series1.AddBar(Block4Avg,'Block 4',789965);
  ResultUnit.Results.Series1.AddBar(Block5Avg,'Block 5',10000);
  for Lp := 1 to BT5 do
    Begin
      ResultUnit.Results.Series3.AddXY(Lp,Block5Arr[Lp]);
    End;
  for Lp := 1 to BT3 do
    Begin
      ResultUnit.Results.Series2.AddXY(Lp,Block3Arr[Lp]);
    End;
 //   ResultUnit.Results.TeeFunction1.AddPoints(ResultUnit.Results.Series2);
 //   ResultUnit.Results.TeeFunction1.Calculate(ResultUnit.Results.Series2,1,40);
 //   ResultUnit.Results.TeeFunction1.Calculate(ResultUnit.Results.Series3,1,40);
    Results.Visible:=False;
    BkClose.Visible:=True;
end;
```

```
Procedure GetMaskStimuli(Mask: tLabel);
Var LoopStreet : Integer;
Letter : Char;
Begin
  Maskfield:='';
  For LoopStreet:= 1 TO 9 DO
    Begin
      Letter:= Chr(Random(15)+96);
      Maskfield:=Concat(Maskfield,Letter);
    End;
    Mask.Caption:=Maskfield;
  End;

Procedure TTWinForm1.StartTimer;
Begin
  IF (B1) Then
  BEGIN
    GetMaskStimuli(Mask);
    Prime.Caption:=JoinArray[Counter];
    Prime.Visible:=True;
  End;
  IF (B2) THEN
  BEGIN
    GetMaskStimuli(Mask);
    Prime.Visible:=True;
    IF (ControlArray20[Counter]='P') Then
    Prime.Caption:=Subjoin[Counter];
    IF (ControlArray20[Counter]='Q') Then
    Prime.Caption:=Subjoin[Counter];
  End;
  IF (B3)Then
  BEGIN
     CheckPicture35(Image2, Stimulus);
     PrimeImage35(GPrime);
     GPrime.Visible:=True; //  wwwwwwwwwwwwwww
    End;
  IF (B4) Then
  BEGIN
    GetMaskStimuli(Mask);
    Prime.Visible:=True;
    IF (ControlArray20[Counter]='Q')
    Then  Prime.Caption:=Subjoin[Counter];
    IF (ControlArray20[Counter]='P')
    Then Prime.Caption:=Subjoin[Counter];
  End;
   If (B5) Then
   BEGIN
     CheckPicture35(Image2, Stimulus);
     PrimeImage35(Gprime);
    GPrime.Visible:=True;  // wwwwwwwwwwwwwwwwwwww
   End;
   Timer2.Enabled:=true;
End;
```

```
procedure TTWinForm1.Timer1Timer(Sender: TObject);
begin
  IF (B1) Then
  BEGIN
    GetMaskStimuli(Mask);
    Prime.Caption:=JoinArray[Counter];
    Prime.Visible:=True;
  End;
  IF (B2) THEN
  BEGIN
    GetMaskStimuli(Mask);
    Prime.Visible:=True;
    IF (ControlArray20[Counter]='P') Then
    Prime.Caption:=Subjoin[Counter];
    IF (ControlArray20[Counter]='Q') Then
    Prime.Caption:=Subjoin[Counter];
  End;
  IF (B3)Then
  BEGIN
     CheckPicture35(Image2, Stimulus);
     PrimeImage35(GPrime);
     GPrime.Visible:=True; //  wwwwwwwwwwwwwww
    End;
  IF (B4) Then
  BEGIN
    GetMaskStimuli(Mask);
    Prime.Visible:=True;
    IF (ControlArray20[Counter]='Q')
    Then  Prime.Caption:=Subjoin[Counter];
    IF (ControlArray20[Counter]='P')
    Then Prime.Caption:=Subjoin[Counter];
  End;
   If (B5) Then
   BEGIN
     CheckPicture35(Image2, Stimulus);
     PrimeImage35(Gprime);
     GPrime.Visible:=True;  // wwwwwwwwwwwwwwwwwwww
   End;
   Timer2.Enabled:=true;
end;

procedure TTWinForm1.Timer2Timer(Sender: TObject);
begin
   IF (B1) OR (B2) or (B4) THEN
   BEGIN
     Mask.Visible:=true;
     Prime.visible:=false;
     Timer3.Enabled:=true;
   End;
   IF (B3)OR (B5) THEN
   BEGIN
     GPrime.Visible:=False;
     IF MaskFlag then Gmask.Visible:=True;
     Timer3.Enabled:=true;
```

```
   End;
   Timer2.Enabled:=False;
end;

procedure TTWinForm1.Timer3Timer(Sender: TObject);
begin
    Mask.Visible:=false;
    IF (B1) OR (B2) OR (B4) Then
    BEGIN
      Stimulus.Caption:=Joinarray[Counter];
      Stimulus.Visible:=True;
    End;
    IF (B3) OR(B5) THEN
    BEGIN
      Gmask.Visible:=False;
      if Priming then
      Image2.Visible:=True;
    End;
    Timer3.Enabled:=false;
    NoKey:=False;
    StartClock;
end;

procedure TTWinForm1.Timer4Timer(Sender: TObject);
begin
Shape1.Visible:=False;
Timer4.Enabled:=false;
end;

procedure TTWinForm1.Timer5Timer(Sender: TObject);
begin
  GMask.Visible:=False;
  Timer5.Enabled:=False;
  StartTimer;
end;

procedure TTWinForm1.TrialClick(Sender: TObject);
begin
  TWinForm1.KeyPreview:=True;
  Lcon.Visible:=True;
  Rcon.visible:=True;
  BkClose.Visible:=false;
  Memo1.Visible:=False;
  Trial.Visible:=False;
  NextCue.Caption:='Press SPACE to begin';
  NextCue.Visible:=True;
  If (Starting) THEN
  BEGIN
    B1:=True;
    Starting:=False;
  END;
 end;

Procedure TTWinForm1.UpdateScreen;
```

```
Begin
  BtnInitialise.ImageIndex:=0;
  BtnInitialise.Enabled:=false;
  ProgressBar2.Visible:=True;
  NextCue.Caption:='PLEASE WAIT WHILE FILES ARE PREPARED';
  NextCue.Visible:=True;
  TWinForm1.Cursor:=crHourGlass;
  Application.ProcessMessages;
End;

Procedure TTWinForm1.UpdatePrompt;
Begin
    Application.ProcessMessages;
    Memo1.Visible:=false;
    Progressbar2.Max:=140;
    Progressbar2.position:=0;
    Progressbar2.Visible:=true;
    NextCue.Caption:='PLEASE WAIT A MOMENT - UPLOADING RESULTS';
    NextCue.Visible:=True;
    TWinform1.Cursor:=crHourGlass;
End;

procedure TTWinForm1.BtnInitialiseClick(Sender: TObject);
begin
  UpdateScreen;
  GetFiles;
  ProgressBar2.Visible:=false;
  BtnInitialise.Visible:=False;
  Name1.Text:=Login.VNumber;
end;

Procedure TTWinForm1.GetFiles;
Var blob:  TBlobField;
    Table : TMemo;
    FLocation,FName : ShortString;
Begin
  Application.ProcessMessages;
  ADOConnection1.ConnectionString:=Login.CString;
  ADOConnection1.Connected:=true;
  ADOTable1.TableName:='textfields';
  ADOTable1.Open;
  ADOTable1.Active:=true;
  With ADOTable1 DO
  Begin
    Table:=TMemo.Create(Self);
    Table.Text:=ADOTable1.Fields.FieldByName('parameters').Text;
    Memo2.Text:=Table.Text;
    OutPut:=Memo2.Lines[1];
    FileLoc:=Memo2.Lines[0];
    FLocation:=(FileLoc+'Parameters.txt');
    Parameters:=FLocation;
    Table.Lines.SaveToFile(FLocation);
    Table.Text:='';
    FLocation:=(FileLoc+'times.txt');
```

```
TimerValues:=FLocation;
Table.Text:=ADOTable1.Fields.FieldByName('times').Text;
Table.Lines.SaveToFile(FLocation);
Table.Text:='';
Questions:=(FileLoc+'Questions.txt');
Table.Text:=ADOTable1.Fields.FieldByName('questions').Text;
Memo4.Text:=Table.Text;
LblQ1.Caption:=Memo4.Lines[0];
LangIA.Caption:=Memo4.Lines[1];
LangAfrikaans.Caption:=Memo4.Lines[2];
LangEng.Caption:=Memo4.Lines[3];
LangEurope.Caption:=Memo4.Lines[4];
LangME.Caption:=Memo4.Lines[5];
LangHisp.Caption:=Memo4.Lines[6];
LangEast.Caption:=Memo4.Lines[7];
Table.Lines.SaveToFile(Questions);
Table.Text:='';
FLocation:=(FileLoc+'Verdicts.txt');
Table.Text:=ADOTable1.Fields.FieldByName('verdicts').Text;
Memo4.Text:=Table.Text;
Table.Lines.SaveToFile(FLocation);
Table.Text:='';
FName:=Memo2.Lines[2];
Table.Text:=ADOTable1.Fields.FieldByName('PWE').Text;
Table.Lines.SaveToFile(FName);
Table.Text:='';
FName:=Memo2.Lines[3];
Table.Text:=ADOTable1.Fields.FieldByName('UWE').Text;
Table.Lines.SaveToFile(FName);
Table.Text:='';
FName:=Memo2.Lines[4];
Table.Text:=ADOTable1.Fields.FieldByName('IGItems').Text;
Memo3.Text:=Table.Text;
Table.Lines.SaveToFile(FName);
Table.Text:='';
FName:=Memo2.Lines[5];
Table.Text:=ADOTable1.Fields.FieldByName('OGItems').Text;
Memo3.Lines.Append(Table.Text);
Table.Lines.SaveToFile(FName);
Table.Text:='';
FName:=Memo2.Lines[6];
Table.Text:=ADOTable1.Fields.FieldByName('concepts').Text;
Table.Lines.SaveToFile(FName);
Table.Text:='';
FName:=Memo2.Lines[8];
Table.Text:=ADOTable1.Fields.FieldByName('SubPItems').Text;
Table.Lines.SaveToFile(FName);
Table.Text:='';
FName:=Memo2.Lines[7];
Table.Text:=ADOTable1.Fields.FieldByName('SubUItems').Text;
Table.Lines.SaveToFile(FName);
Table.Text:='';
FName:=Memo2.Lines[9];
Table.Text:=ADOTable1.Fields.FieldByName('SubPOItems').Text;
```

```
  Memo3.Lines.Append(Table.Text);
  Table.Lines.SaveToFile(FName);
  Table.Text:='';
  FName:=Memo2.Lines[10];
  Table.Text:=ADOTable1.Fields.FieldByName('SubUOItems').Text;
  Memo3.Lines.Append(Table.Text);
  Table.Lines.SaveToFile(FName);
  Table.Free;
End;
With ADOTable1 DO
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='settings';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('M1') as TBlobField;
    FName:=Memo2.Lines[15];
    Blob.SaveToFile(FName); //('C:\IAT\Files\M1.rtf');
    blob := ADOTable1.fields.FieldByName('M2') as TBlobField;
    FName:=Memo2.Lines[16];
    Blob.SaveToFile(FName);              //Blob.SaveToFile('C:\IAT\Files\M2.rtf');
    blob := ADOTable1.fields.FieldByName('M3') as TBlobField;
    FName:=Memo2.Lines[17];
    Blob.SaveToFile(FName);  // Blob.SaveToFile('C:\IAT\Files\M3.rtf');
    blob := ADOTable1.fields.FieldByName('M4') as TBlobField;
    FName:=Memo2.Lines[18];
    Blob.SaveToFile(FName);//   Blob.SaveToFile('C:\IAT\Files\M4.rtf');
    blob := ADOTable1.fields.FieldByName('M5') as TBlobField;
    FName:=Memo2.Lines[19];
    Blob.SaveToFile(FName);//     Blob.SaveToFile('C:\IAT\Files\M5.rtf');
    blob := ADOTable1.fields.FieldByName('M6') as TBlobField;
    FName:=Memo2.Lines[20];
    Blob.SaveToFile(FName);//     Blob.SaveToFile('C:\IAT\Files\M6.rtf');
    blob := ADOTable1.fields.FieldByName('M7') as TBlobField;
    FName:=Memo2.Lines[21];
    Blob.SaveToFile(FName);//     Blob.SaveToFile('C:\IAT\Files\M7.rtf');
    blob := ADOTable1.fields.FieldByName('MA') as TBlobField;
    FName:=Memo2.Lines[14];
    Blob.SaveToFile(FName);//     Blob.SaveToFile('C:\IAT\Files\MA.rtf');
    blob := ADOTable1.fields.FieldByName('IN1') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[0]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('IN2') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[1]+'.bmp');
    Blob.SaveToFile(FName);
End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='in3to5';
    ADOTable1.Open;
```

```
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('IN3') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[2]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('IN4') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[3]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('IN5') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[4]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='in6to10';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('IN6') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[5]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('IN7') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[6]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('IN8') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[7]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('IN9') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[8]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('IN10') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[9]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='out1to3';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('out1') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[10]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('out2') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[11]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('out3') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[12]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
```

```
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='out4to6';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('out4') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[13]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('out5') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[14]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('out6') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[15]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='out7to10';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('out7') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[16]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('out8') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[17]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('out9') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[18]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('out10') as TBlobField;
    FName:=(Memo2.Lines[11]+Memo3.Lines[19]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subp1to3';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subp1') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[0]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp2') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[1]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp3') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[2]+'.bmp');
```

```
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subp4to6';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subp4') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[3]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp5') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[4]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp6') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[5]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subp7to10';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subp7') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[6]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp8') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[7]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp9') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[8]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp10') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[9]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subp11to13';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subp11') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[10]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp12') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[11]+'.bmp');
```

```
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp13') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[12]+'.bmp');
    Blob.SaveToFile(FName);
   End;
 ProgressBar2.StepBy(5);
 With ADOTable1 Do
   Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subp14to16';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subp14') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[13]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp15') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[14]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp16') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[15]+'.bmp');
    Blob.SaveToFile(FName);
   End;
   ProgressBar2.StepBy(5);
    With ADOTable1 Do
   Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subp17to20';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subp17') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[16]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp18') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[17]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp19') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[18]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp20') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[19]+'.bmp');
    Blob.SaveToFile(FName);
   End;
 ProgressBar2.StepBy(5);
 With ADOTable1 Do
   Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subu1to3';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subu1') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[0]+'.bmp');
```

```
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu2') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[1]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu3') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[2]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subu4to6';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subu4') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[3]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu5') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[4]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu6') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[5]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subu7to10';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subu7') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[6]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu8') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[7]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu9') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[8]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu10') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[9]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subu11to13';
    ADOTable1.Open;
```

```
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subu11') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[10]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu12') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[11]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu13') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[12]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subu14to16';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subu14') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[13]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu15') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[14]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu16') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[15]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='subu17to20';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subu17') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[16]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu18') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[17]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu19') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[18]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu20') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[19]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
```

```
    ADOTable1.Active:=false;
    ADOTable1.TableName:='s1to3';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subp11') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[20]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp12') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[21]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp13') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[22]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='s4to6';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subp14') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[23]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp15') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[24]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp16') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[25]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='s7to10';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subp17') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[26]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp18') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[27]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp19') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[28]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subp20') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[29]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
```

```
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='s1to3b';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subu11') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[30]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu12') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[31]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu13') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[32]+'.bmp');
    Blob.SaveToFile(FName);
  End;
  ProgressBar2.StepBy(5);
    With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='s4to6b';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subu14') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[33]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu15') as TBlobField;
    FName:=(Memo2.Lines[13]+Memo3.Lines[34]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu16') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[35]+'.bmp');
    Blob.SaveToFile(FName);
  End;
ProgressBar2.StepBy(5);
With ADOTable1 Do
  Begin
    ADOTable1.Close;
    ADOTable1.Active:=false;
    ADOTable1.TableName:='s7to10b';
    ADOTable1.Open;
    ADOTable1.Active:=true;
    blob := ADOTable1.fields.FieldByName('subu17') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[36]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu18') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[37]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu19') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[38]+'.bmp');
    Blob.SaveToFile(FName);
    blob := ADOTable1.fields.FieldByName('subu20') as TBlobField;
    FName:=(Memo2.Lines[12]+Memo3.Lines[39]+'.bmp');
```

```
      Blob.SaveToFile(FName);
      ADOTable1.Close;
      ADOTable1.Active:=false;
    End;
    ProgressBar2.StepBy(5);
    TWinForm1.Cursor:=crDefault;
    AScreen2.Visible:=True;
    NextCue.Caption:='Click "NEXT" TO CONTINUE';

  SetPriorityClass(GetCurrentProcess, HIGH_PRIORITY_CLASS);
  SetThreadPriority(GetCurrentThread, THREAD_PRIORITY_TIME_CRITICAL);
  Block1Order:=1;
  Block2Order:=2;
  Block3Order:=3;
  Block4Order:=4;
  Block5Order:=5;
  FileFunctions.OpenFiles;
  FileFunctions.Readarrays;
  FileFunctions.InitArrays;
  ArrayFunctions.RandomsortP(Pleasant,Pl);
  ArrayFunctions.RandomsortU(Unpleasant,Ul);
  ArrayFunctions.RandomsortF(Blacks,Fl);
  ArrayFunctions.RandomsortC(Whites,Cl);
  Initialise(Block1,Block2,Block3,BLock4,Block5,B1,B2,B3,B4,B5,Starting,Counter,
  Time1,Time2,Score);
  NoKey:=False;
  BT1:=PL+UL;
  BT2:=FL+CL;
  BT3:=PL+UL+FL+CL;
  BT4:=FL+CL;
  BT5:=PL+UL+FL+CL;
  SPNEB:=False;
  SPSSEB:=False;
  SPSPREB:=False;
  SPSTRSEB:=False;
  SPSTRREB:=False;
End;

end.
```

```
unit Unit3;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ADODB, DB, ExtCtrls, DBCtrls, Grids, DBGrids,
  AdvSmoothStatusIndicator, Buttons, Mask, ImgList;

type
  TForm3 = class(TForm)
    ADOConnection1: TADOConnection;
    ADOCommand1: TADOCommand;
    ComboBox1: TComboBox;
    Button1: TButton;
    GroupBox1: TGroupBox;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label17: TLabel;
    Label18: TLabel;
    Label19: TLabel;
    Label20: TLabel;
    Label21: TLabel;
    Label22: TLabel;
    Label23: TLabel;
    Label24: TLabel;
    Label25: TLabel;
    Label26: TLabel;
    Label27: TLabel;
    Label28: TLabel;
    Label29: TLabel;
    Label30: TLabel;
    Label31: TLabel;
    Label32: TLabel;
    Label33: TLabel;
    Label34: TLabel;
    Label35: TLabel;
    Label36: TLabel;
    Button2: TButton;
```

ComboBox2: TComboBox;
Label37: TLabel;
Label38: TLabel;
Label39: TLabel;
Label40: TLabel;
Label41: TLabel;
DBGrid1: TDBGrid;
DBNavigator1: TDBNavigator;
DataSource1: TDataSource;
ADOTable1: TADOTable;
Button3: TButton;
ADOConnection2: TADOConnection;
ADOCommand2: TADOCommand;
Button4: TButton;
T20S1: TAdvSmoothStatusIndicator;
T20S2: TAdvSmoothStatusIndicator;
T20S3: TAdvSmoothStatusIndicator;
T20S4: TAdvSmoothStatusIndicator;
T20S5: TAdvSmoothStatusIndicator;
Label42: TLabel;
Label43: TLabel;
Label44: TLabel;
Label45: TLabel;
Label46: TLabel;
Label47: TLabel;
S1C2: TShape;
S2C2: TShape;
S3C2: TShape;
S1C1: TShape;
S2C1: TShape;
S3C1: TShape;
S4C1: TShape;
S4C2: TShape;
S5C1: TShape;
S5C2: TShape;
Button5: TButton;
Button6: TButton;
BitBtn1: TBitBtn;
Label48: TLabel;
Label49: TLabel;
Label50: TLabel;
PC1: TShape;
PC2: TShape;
E1: TDBEdit;
E2: TDBEdit;
E3: TDBEdit;
E4: TDBEdit;
E5: TDBEdit;
E6: TDBEdit;
E7: TDBEdit;
E37: TDBEdit;
E8: TDBEdit;
E9: TDBEdit;
E10: TDBEdit;

E11: TDBEdit;
E12: TDBEdit;
E13: TDBEdit;
E14: TDBEdit;
AVM: TDBEdit;
E15: TDBEdit;
E16: TDBEdit;
E17: TDBEdit;
E18: TDBEdit;
E19: TDBEdit;
E20: TDBEdit;
E21: TDBEdit;
E22: TDBEdit;
E23: TDBEdit;
E24: TDBEdit;
E25: TDBEdit;
E26: TDBEdit;
E27: TDBEdit;
E28: TDBEdit;
E29: TDBEdit;
E30: TDBEdit;
UVM: TDBEdit;
E38: TDBEdit;
E39: TDBEdit;
E40: TDBEdit;
E41: TDBEdit;
E31: TDBEdit;
E32: TDBEdit;
E33: TDBEdit;
E34: TDBEdit;
E35: TDBEdit;
E36: TDBEdit;
LVM: TDBEdit;
S1C1R: TDBEdit;
S1C1G: TDBEdit;
S1C1B: TDBEdit;
S2C1R: TDBEdit;
S2C1G: TDBEdit;
S2C1B: TDBEdit;
S3C1R: TDBEdit;
S3C1G: TDBEdit;
S3C1B: TDBEdit;
S4C1R: TDBEdit;
S4C1G: TDBEdit;
S4C1B: TDBEdit;
S5C1R: TDBEdit;
S5C1G: TDBEdit;
S5C1B: TDBEdit;
S1C2R: TDBEdit;
S1C2G: TDBEdit;
S1C2B: TDBEdit;
S2C2R: TDBEdit;
S2C2G: TDBEdit;
S2C2B: TDBEdit;

S3C2R: TDBEdit;
S3C2G: TDBEdit;
S3C2B: TDBEdit;
S4C2R: TDBEdit;
S4C2G: TDBEdit;
S4C2B: TDBEdit;
S5C2R: TDBEdit;
S5C2G: TDBEdit;
S5C2B: TDBEdit;
s1t: TDBEdit;
S2T: TDBEdit;
S3T: TDBEdit;
S4T: TDBEdit;
S5T: TDBEdit;
FormR: TDBEdit;
FormG: TDBEdit;
FormB: TDBEdit;
FieldR: TDBEdit;
FieldG: TDBEdit;
FieldB: TDBEdit;
StimulusR: TDBEdit;
StimulusG: TDBEdit;
StimulusB: TDBEdit;
Cal1R: TDBEdit;
Cal1G: TDBEdit;
Cal1B: TDBEdit;
Cal2R: TDBEdit;
Cal2G: TDBEdit;
Cal2B: TDBEdit;
PC1R: TDBEdit;
PC2R: TDBEdit;
PC1G: TDBEdit;
PC2G: TDBEdit;
PC1B: TDBEdit;
PC2B: TDBEdit;
ChkPrescribed: TCheckBox;
ChkUserCal: TCheckBox;
ChkSizer: TCheckBox;
DBCal1: TDBText;
DBPrescribed1: TDBText;
DBSizer: TDBEdit;
DBPrescribed: TDBEdit;
DBCal: TDBEdit;
ChkBypass: TCheckBox;
DBbyPass: TDBEdit;
OpenDialog1: TOpenDialog;
SaveDialog1: TSaveDialog;
BtnOpenFile: TButton;
BtnSaveFile: TButton;
ImageList1: TImageList;
procedure ComboBox1Change(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure ComboBox2Change(Sender: TObject);
procedure Button2Click(Sender: TObject);

```
    procedure Button3Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure T20S1Click(Sender: TObject);
    procedure T20S2Click(Sender: TObject);
    procedure T20S3Click(Sender: TObject);
    procedure T20S4Click(Sender: TObject);
    procedure T20S5Click(Sender: TObject);
    Procedure UpdateColour;
    Procedure UpdateFields;
    procedure DBGrid1MouseLeave(Sender: TObject);
    procedure Button5Click(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
    procedure Button6Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
    procedure DataSource1DataChange(Sender: TObject; Field: TField);
    procedure ChkSizerClick(Sender: TObject);
    procedure ChkUserCalClick(Sender: TObject);
    procedure ChkPrescribedClick(Sender: TObject);
    procedure ChkBypassClick(Sender: TObject);
    procedure BtnSaveFileClick(Sender: TObject);
    procedure BtnOpenFileClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form3: TForm3;
  Var CString : WideString;
  Var SList : TStringlist;

implementation

{$R *.dfm}

Procedure CreateTableParticipants;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`participants` (';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`ID` varchar(9) DEFAULT NULL,';
  SString:=SString+'`Gender` varchar(1) DEFAULT NULL,';
  SString:=SString+'`Age` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Visual Rating` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Colour Blind` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Date` datetime DEFAULT NULL,';
  SString:=SString+'`V1` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V2` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V3` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V4` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V5` int(10) unsigned DEFAULT NULL,';
```

```
  SString:=SString+'`V6` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V7` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V8` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V9` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`First Language` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form3.ADOCommand1.CommandText:=SString;
  Form3.ADOCommand1.Execute;
End;

procedure TForm3.BitBtn1Click(Sender: TObject);
begin
 ADOTable1.Active:=false;
 ADoTable1.Close;
 ADOCommand1.Free;
 ADOCommand2.Free;
 ADOConnection1.Close;
 ADOConnection2.Close;
end;

procedure TForm3.BtnOpenFileClick(Sender: TObject);
Var Idx : Integer;
    EventDate : ShortString;
begin
  Idx:=0;
  SList:=TStringList.Create;
  if OpenDialog1.Execute then
    SList.LoadFromFile(OpenDialog1.FileName);
  if SList.Count>0 then
  begin
    ADOTable1.TableName:='settings';
    ADOTable1.Open;
    ADOTable1.Active:=True;
      With ADOTable1 Do
        Begin
          Edit;
          Fields.FieldByName('Timer1').Value:=StrToInt(SList[Idx]);
          Inc(Idx);
          Fields.FieldByName('Timer2').Value:=StrToInt(SList[Idx]);
          Inc(Idx);
          Fields.FieldByName('Timer3').Value:=StrToInt(SList[Idx]);
          Inc(Idx);
          Fields.FieldByName('Mask').Value:=SList[Idx];
          Inc(Idx);
          Fields.FieldByName('BGC1').Value:=SList[Idx];
          Inc(Idx);
          Fields.FieldByName('BGC2').Value:=SList[Idx];
          Inc(Idx);
          Fields.FieldByName('SizerInterval').Value:=StrToInt(SList[Idx]);
          Inc(Idx);
          Fields.FieldByName('Overide').Value:=SList[Idx];
          Inc(Idx);
          Fields.FieldByName('Random').Value:=SList[Idx];
```

```
Inc(Idx);
Fields.FieldByName('PracticeLight').Value:=SList[Idx];
Inc(Idx);
Fields.FieldByName('MainLight').Value:=SList[Idx];
Inc(Idx);
Fields.FieldByName('Calibrate').Value:=SList[Idx];
Inc(Idx);
Fields.FieldByName('Title').Value:=SList[Idx];
Inc(Idx);;
Fields.FieldByName('Graphics').Value:=SList[Idx];
Inc(Idx);
Fields.FieldByName('Panel').Value:=SList[Idx];
Inc(Idx);
Fields.FieldByName('CIterations').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('TurnaroundM').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('MaxRuns').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('P1Time').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('P2Time').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('P3Time').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('P4Time').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('P5Time').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FromBGR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FromBGG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FromBGB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FromFGR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FromFGG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FromFGB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Equiliminance').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToBGR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToBGG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToBGB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToFGR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToFGG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToFGB').Value:=StrToInt(SList[Idx]);
```

```
Inc(Idx);
Fields.FieldByName('SizerFactor').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Sizer').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('TargetWidth').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('TargetHeight').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('TargetLeft').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('TargetTop').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C1B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C2B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2C1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2C1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2C1B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2C2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2C2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2C2B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C1B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C2B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4C1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4C1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4C1B').Value:=StrToInt(SList[Idx]);
```

```
Inc(Idx);
Fields.FieldByName('S4C2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4C2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4C2B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C1B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C2B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1T').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2T').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3T').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4T') .Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5T').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('UVM') .Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('LVM') .Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('AVM').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FormR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FormG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FormB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FieldR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FieldG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FieldB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('StimulusR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('StimulusG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('StimulusB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('User Calibration').Value:=StrToInt(SList[Idx]);
```

```
        Inc(Idx);
        Fields.FieldByName('Cal1R').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('Cal1G').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('Cal1B').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('Cal2R').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('Cal2G').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('Cal2B').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('Prescribed Solution').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('Bypass Calibration').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('PC1R').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('PC1G').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('PC1B').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('PC2R').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('PC2G').Value:=StrToInt(SList[Idx]);
        Inc(Idx);
        Fields.FieldByName('PC2B').Value:=StrToInt(SList[Idx]);
        EventDate:=DateTimeToStr(Now);
        Fields.FieldByName('Date').AsDateTime:=StrToDateTime(EventDate);
        UpDateRecord;
        Post;
      End;
      SList.Free;
  end;
end;

procedure TForm3.BtnSaveFileClick(Sender: TObject);
Var EventDate : ShortString;
begin
  SList:=TStringlist.Create;
  Slist.Add(E1.Text);
  Slist.Add(E2.Text);
  Slist.Add(E3.Text);
  Slist.Add(E4.Text);
  Slist.Add(E5.Text);
  Slist.Add(E6.Text);
  Slist.Add(E7.Text);
  Slist.Add(E8.Text);
  Slist.Add(E9.Text);
  Slist.Add(E10.Text);
  Slist.Add(E11.Text);
  Slist.Add(E12.Text);
  Slist.Add(E13.Text);
```

```
Slist.Add(E14.Text);
Slist.Add(E15.Text);
Slist.Add(E16.Text);
Slist.Add(E17.Text);
Slist.Add(E18.Text);
Slist.Add(E19.Text);
Slist.Add(E20.Text);
Slist.Add(E21.Text);
Slist.Add(E22.Text);
Slist.Add(E23.Text);
Slist.Add(E24.Text);
Slist.Add(E25.Text);
Slist.Add(E26.Text);
Slist.Add(E27.Text);
Slist.Add(E28.Text);
Slist.Add(E29.Text);
Slist.Add(E30.Text);
Slist.Add(E31.Text);
Slist.Add(E32.Text);
Slist.Add(E33.Text);
Slist.Add(E34.Text);
Slist.Add(E35.Text);
Slist.Add(E36.Text);
Slist.Add(E37.Text);
   if Form3.ChkSizer.Checked then
     Slist.Add('1')
       Else
        Slist.Add('0');
Slist.Add(E38.Text);
Slist.Add(E39.Text);
Slist.Add(E40.Text);
Slist.Add(E41.Text);
Slist.Add(S1C1R.Text);
Slist.Add(S1C1G.Text);
Slist.Add(S1C1B.Text);
Slist.Add(S1C2R.Text);
Slist.Add(S1C2G.Text);
Slist.Add(S1C2B.Text);
Slist.Add(S2C1R.Text);
Slist.Add(S2C1G.Text);
Slist.Add(S2C1B.Text);
Slist.Add(S2C2R.Text);
Slist.Add(S2C2G.Text);
Slist.Add(S2C2B.Text);
Slist.Add(S3C1R.Text);
Slist.Add(S3C1G.Text);
Slist.Add(S3C1B.Text);
Slist.Add(S3C2R.Text);
Slist.Add(S3C2G.Text);
Slist.Add(S3C2B.Text);
Slist.Add(S4C1R.Text);
Slist.Add(S4C1G.Text);
Slist.Add(S4C1B.Text);
Slist.Add(S4C2R.Text);
```

```
Slist.Add(S4C2G.Text);
Slist.Add(S4C2B.Text);
Slist.Add(S5C1R.Text);
Slist.Add(S5C1G.Text);
Slist.Add(S5C1B.Text);
Slist.Add(S5C2R.Text);
Slist.Add(S5C2G.Text);
Slist.Add(S5C2B.Text);
Slist.Add(S1T.Text);
Slist.Add(S2T.Text);
Slist.Add(S3T.Text);
Slist.Add(S4T.Text);
Slist.Add(S5T.Text);
Slist.Add(UVM.Text);
Slist.Add(LVM.Text);
Slist.Add(AVM.Text);
Slist.Add(FormR.Text);
Slist.Add(FormG.Text);
Slist.Add(FormB.Text);
Slist.Add(FieldR.Text);
Slist.Add(FieldG.Text);
Slist.Add(FieldB.Text);
Slist.Add(StimulusR.Text);
Slist.Add(StimulusG.Text);
Slist.Add(StimulusB.Text);
  if Form3.ChkUserCal.Checked then
    Slist.Add('1')
      Else
       Slist.Add('0');
Slist.Add(Cal1R.Text);
Slist.Add(Cal1G.Text);
Slist.Add(Cal1B.Text);
Slist.Add(Cal2R.Text);
Slist.Add(Cal2G.Text);
Slist.Add(Cal2B.Text);
   if Form3.ChkPrescribed.Checked then
     Slist.Add('1')
       Else
        Slist.Add('0');
   if Form3.ChkByPass.Checked then
     Slist.Add('1')
       Else
        Slist.Add('0');
 Slist.Add(PC1R.Text);
 Slist.Add(PC1G.Text);
 Slist.Add(PC1B.Text);
 Slist.Add(PC2R.Text);
 Slist.Add(PC2G.Text);
 Slist.Add(PC2B.Text);
 EventDate:=DateTimeToStr(Now);
 Slist.Add(EventDate);
 if SaveDialog1.Execute then
  SList.SaveToFile(SaveDialog1.FileName);
 SList.Free;
```

```
end;

procedure TForm3.Button1Click(Sender: TObject);
begin
  CreateTableParticipants;
end;

Procedure CreateTableCalibration;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`settings` (';
  SString:=SString+'`Timer1` int(10) unsigned NOT NULL,';
  SString:=SString+'`Timer2` int(10) unsigned NOT NULL,';
  SString:=SString+'`Timer3` int(10) unsigned NOT NULL,';
  SString:=SString+'`Mask` varchar(10) DEFAULT NULL,';
  SString:=SString+'`BGC1` varchar(10) DEFAULT NULL,';
  SString:=SString+'`BGC2` varchar(10) DEFAULT NULL,';
  SString:=SString+'`SizerInterval` int(10) unsigned NOT NULL,';
  SString:=SString+'`Overide` varchar(10) DEFAULT NULL,';
  SString:=SString+'`Random` varchar(10) DEFAULT NULL,';
  SString:=SString+'`PracticeLight` varchar(10) DEFAULT NULL,';
  SString:=SString+'`MainLight` varchar(10) DEFAULT NULL,';
  SString:=SString+'`Calibrate` varchar(11) DEFAULT NULL,';
  SString:=SString+'`Title` varchar(30) DEFAULT NULL,';
  SString:=SString+'`Graphics` varchar(10) DEFAULT NULL,';
  SString:=SString+'`Panel` varchar(10) DEFAULT NULL,';
  SString:=SString+'`CIterations` int(10) unsigned NOT NULL,';
  SString:=SString+'`TurnaroundM` int(10) unsigned NOT NULL,';
  SString:=SString+'`MaxRuns` int(10) unsigned NOT NULL,';
  SString:=SString+'`P1Time` int(10) unsigned NOT NULL,';
  SString:=SString+'`P2Time` int(10) unsigned NOT NULL,';
  SString:=SString+'`P3Time` int(10) unsigned NOT NULL,';
  SString:=SString+'`P4Time` int(10) unsigned NOT NULL,';
  SString:=SString+'`P5Time` int(10) unsigned NOT NULL,';
  SString:=SString+'`FromBGR` int(10) unsigned NOT NULL,';
  SString:=SString+'`FromBGG` int(10) unsigned NOT NULL,';
  SString:=SString+'`FromBGB` int(10) unsigned NOT NULL,';
  SString:=SString+'`ToBGR` int(10) unsigned NOT NULL,';
  SString:=SString+'`ToBGG` int(10) unsigned NOT NULL,';
  SString:=SString+'`ToBGB` int(10) unsigned NOT NULL,';
  SString:=SString+'`FromFGR` int(10) unsigned NOT NULL,';
  SString:=SString+'`FromFGG` int(10) unsigned NOT NULL,';
  SString:=SString+'`FromFGB` int(10) unsigned NOT NULL,';
  SString:=SString+'`ToFGR` int(10) unsigned NOT NULL,';
  SString:=SString+'`ToFGG` int(10) unsigned NOT NULL,';
  SString:=SString+'`ToFGB` int(10) unsigned NOT NULL,';
  SString:=SString+'`Equiliminance` int(10) unsigned NOT NULL,';
  SString:=SString+'`Sizer` TINYINT(1) unsigned NOT NULL,';
  SString:=SString+'`SizerFactor` int(10) unsigned NOT NULL,';
  SString:=SString+'`TargetWidth` int(10) unsigned NOT NULL,';
  SString:=SString+'`TargetHeight` int(10) unsigned NOT NULL,';
  SString:=SString+'`TargetLeft` int(10) unsigned NOT NULL,';
  SString:=SString+'`TargetTop` int(10) unsigned NOT NULL,';
```

```
SString:=SString+'`S1C1R` int(10) unsigned NOT NULL,';
SString:=SString+'`S1C1G` int(10) unsigned NOT NULL,';
SString:=SString+'`S1C1B` int(10) unsigned NOT NULL,';
SString:=SString+'`S1C2R` int(10) unsigned NOT NULL,';
SString:=SString+'`S1C2G` int(10) unsigned NOT NULL,';
SString:=SString+'`S1C2B` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C1R` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C1G` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C1B` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C2R` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C2G` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C2B` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C1R` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C1G` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C1B` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C2R` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C2G` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C2B` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C1R` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C1G` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C1B` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C2R` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C2G` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C2B` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C1R` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C1G` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C1B` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C2R` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C2G` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C2B` int(10) unsigned NOT NULL,';
SString:=SString+'`S1T` int(10) unsigned NOT NULL,';
SString:=SString+'`S2T` int(10) unsigned NOT NULL,';
SString:=SString+'`S3T` int(10) unsigned NOT NULL,';
SString:=SString+'`S4T` int(10) unsigned NOT NULL,';
SString:=SString+'`S5T` int(10) unsigned NOT NULL,';
SString:=SString+'`UVM` int(10) unsigned NOT NULL,';
SString:=SString+'`LVM` int(10) unsigned NOT NULL,';
SString:=SString+'`AVM` int(10) unsigned NOT NULL,';
SString:=SString+'`FormR` int(10) unsigned NOT NULL,';
SString:=SString+'`FormG` int(10) unsigned NOT NULL,';
SString:=SString+'`FormB` int(10) unsigned NOT NULL,';
SString:=SString+'`FieldR` int(10) unsigned NOT NULL,';
SString:=SString+'`FieldG` int(10) unsigned NOT NULL,';
SString:=SString+'`FieldB` int(10) unsigned NOT NULL,';
SString:=SString+'`StimulusR` int(10) unsigned NOT NULL,';
SString:=SString+'`StimulusG` int(10) unsigned NOT NULL,';
SString:=SString+'`StimulusB` int(10) unsigned NOT NULL,';
SString:=SString+'`User Calibration` TINYINT(1) unsigned NOT NULL,';
SString:=SString+'`Cal1R` int(10) unsigned NOT NULL,';
SString:=SString+'`Cal1G` int(10) unsigned NOT NULL,';
SString:=SString+'`Cal1B` int(10) unsigned NOT NULL,';
SString:=SString+'`Cal2R` int(10) unsigned NOT NULL,';
SString:=SString+'`Cal2G` int(10) unsigned NOT NULL,';
SString:=SString+'`Cal2B` int(10) unsigned NOT NULL,';
```

```
  SString:=SString+'`Prescribed Solution` TINYINT(1) unsigned NOT NULL,';
  SString:=SString+'`Bypass Calibration` TINYINT(1) unsigned NOT NULL,';
  SString:=SString+'`PC1R` int(10) unsigned NOT NULL,';
  SString:=SString+'`PC1G` int(10) unsigned NOT NULL,';
  SString:=SString+'`PC1B` int(10) unsigned NOT NULL,';
  SString:=SString+'`PC2R` int(10) unsigned NOT NULL,';
  SString:=SString+'`PC2G` int(10) unsigned NOT NULL,';
  SString:=SString+'`PC2B` int(10) unsigned NOT NULL,';
  SString:=SString+'`Date` datetime DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Timer1`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form3.ADOCommand1.CommandText:=SString;
  Form3.ADOCommand1.Execute;
End;

procedure TForm3.Button2Click(Sender: TObject);
begin
  CreateTableCalibration;
end;

procedure TForm3.Button3Click(Sender: TObject);
Var EventDate : ShortString;
begin
  ADOTable1.TableName:='settings';
  ADOTable1.Open;
  ADOTable1.Active:=True;
    With ADOTable1 Do
    Begin
      Append;
      Fields.FieldByName('Timer1').Value:=StrToInt(E1.Text);
      Fields.FieldByName('Timer2').Value:=StrToInt(E2.Text);
      Fields.FieldByName('Timer3').Value:=StrToInt(E3.Text);
      Fields.FieldByName('Mask').Value:=E4.Text;
      Fields.FieldByName('BGC1').Value:=E5.Text;
      Fields.FieldByName('BGC2').Value:=E6.Text;
      Fields.FieldByName('SizerInterval').Value:=StrToInt(E7.Text);
      Fields.FieldByName('Overide').Value:=E8.Text;
      Fields.FieldByName('Random').Value:=E9.Text;
      Fields.FieldByName('PracticeLight').Value:=E10.Text;
      Fields.FieldByName('MainLight').Value:=E11.Text;
      Fields.FieldByName('Calibrate').Value:=E12.Text;
      Fields.FieldByName('Title').Value:=E13.Text;
      Fields.FieldByName('Graphics').Value:=E14.Text;
      Fields.FieldByName('Panel').Value:=E15.Text;
      Fields.FieldByName('CIterations').Value:=StrToInt(E16.Text);
      Fields.FieldByName('TurnaroundM').Value:=StrToInt(E17.Text);
      Fields.FieldByName('MaxRuns').Value:=StrToInt(E18.Text);
      Fields.FieldByName('P1Time').Value:=StrToInt(E19.Text);
      Fields.FieldByName('P2Time').Value:=StrToInt(E20.Text);
      Fields.FieldByName('P3Time').Value:=StrToInt(E21.Text);
      Fields.FieldByName('P4Time').Value:=StrToInt(E22.Text);
      Fields.FieldByName('P5Time').Value:=StrToInt(E23.Text);
      Fields.FieldByName('FromBGR').Value:=StrToInt(E24.Text);
      Fields.FieldByName('FromBGG').Value:=StrToInt(E25.Text);
```

```
Fields.FieldByName('FromBGB').Value:=StrToInt(E26.Text);
Fields.FieldByName('FromFGR').Value:=StrToInt(E27.Text);
Fields.FieldByName('FromFGG').Value:=StrToInt(E28.Text);
Fields.FieldByName('FromFGB').Value:=StrToInt(E29.Text);
Fields.FieldByName('Equiliminance').Value:=StrToInt(E30.Text);
Fields.FieldByName('ToBGR').Value:=StrToInt(E31.Text);
Fields.FieldByName('ToBGG').Value:=StrToInt(E32.Text);
Fields.FieldByName('ToBGB').Value:=StrToInt(E33.Text);
Fields.FieldByName('ToFGR').Value:=StrToInt(E34.Text);
Fields.FieldByName('ToFGG').Value:=StrToInt(E35.Text);
Fields.FieldByName('ToFGB').Value:=StrToInt(E36.Text);
Fields.FieldByName('SizerFactor').Value:=StrToInt(E37.Text);
if Form3.ChkSizer.Checked then
  Fields.FieldByName('Sizer').Value:=1
    Else
     Fields.FieldByName('Sizer').Value:=0;
Fields.FieldByName('TargetWidth').Value:=StrToInt(E38.Text);
Fields.FieldByName('TargetHeight').Value:=StrToInt(E39.Text);
Fields.FieldByName('TargetLeft').Value:=StrToInt(E40.Text);
Fields.FieldByName('TargetTop').Value:=StrToInt(E41.Text);
Fields.FieldByName('S1C1R').Value:=StrToInt(S1C1R.Text);
Fields.FieldByName('S1C1G').Value:=StrToInt(S1C1G.Text);
Fields.FieldByName('S1C1B').Value:=StrToInt(S1C1B.Text);
Fields.FieldByName('S1C2R').Value:=StrToInt(S1C2R.Text);
Fields.FieldByName('S1C2G').Value:=StrToInt(S1C2G.Text);
Fields.FieldByName('S1C2B').Value:=StrToInt(S1C2B.Text);
Fields.FieldByName('S2C1R').Value:=StrToInt(S2C1R.Text);
Fields.FieldByName('S2C1G').Value:=StrToInt(S2C1G.Text);
Fields.FieldByName('S2C1B').Value:=StrToInt(S2C1B.Text);
Fields.FieldByName('S2C2R').Value:=StrToInt(S2C2R.Text);
Fields.FieldByName('S2C2G').Value:=StrToInt(S2C2G.Text);
Fields.FieldByName('S2C2B').Value:=StrToInt(S2C2B.Text);
Fields.FieldByName('S3C1R').Value:=StrToInt(S3C1R.Text);
Fields.FieldByName('S3C1G').Value:=StrToInt(S3C1G.Text);
Fields.FieldByName('S3C1B').Value:=StrToInt(S3C1B.Text);
Fields.FieldByName('S3C2R').Value:=StrToInt(S3C2R.Text);
Fields.FieldByName('S3C2G').Value:=StrToInt(S3C2G.Text);
Fields.FieldByName('S3C2B').Value:=StrToInt(S3C2B.Text);
Fields.FieldByName('S4C1R').Value:=StrToInt(S4C1R.Text);
Fields.FieldByName('S4C1G').Value:=StrToInt(S4C1G.Text);
Fields.FieldByName('S4C1B').Value:=StrToInt(S4C1B.Text);
Fields.FieldByName('S4C2R').Value:=StrToInt(S4C2R.Text);
Fields.FieldByName('S4C2G').Value:=StrToInt(S4C2G.Text);
Fields.FieldByName('S4C2B').Value:=StrToInt(S4C2B.Text);
Fields.FieldByName('S5C1R').Value:=StrToInt(S5C1R.Text);
Fields.FieldByName('S5C1G').Value:=StrToInt(S5C1G.Text);
Fields.FieldByName('S5C1B').Value:=StrToInt(S5C1B.Text);
Fields.FieldByName('S5C2R').Value:=StrToInt(S5C2R.Text);
Fields.FieldByName('S5C2G').Value:=StrToInt(S5C2G.Text);
Fields.FieldByName('S5C2B').Value:=StrToInt(S5C2B.Text);
Fields.FieldByName('S1T').Value:=StrToInt(S1T.Text);
Fields.FieldByName('S2T').Value:=StrToInt(S2T.Text);
Fields.FieldByName('S3T').Value:=StrToInt(S3T.Text);
Fields.FieldByName('S4T') .Value:=StrToInt(S4T.Text);
```

```
    Fields.FieldByName('S5T').Value:=StrToInt(S5T.Text);
    Fields.FieldByName('UVM') .Value:=StrToInt(UVM.Text);
    Fields.FieldByName('LVM') .Value:=StrToInt(LVM.Text);
    Fields.FieldByName('AVM').Value:=StrToInt(AVM.Text);
    Fields.FieldByName('FormR').Value:=StrToInt(FormR.Text);
    Fields.FieldByName('FormG').Value:=StrToInt(FormG.Text);
    Fields.FieldByName('FormB').Value:=StrToInt(FormB.Text);
    Fields.FieldByName('FieldR').Value:=StrToInt(FieldR.Text);
    Fields.FieldByName('FieldG').Value:=StrToInt(FieldG.Text);
    Fields.FieldByName('FieldB').Value:=StrToInt(FieldB.Text);
    Fields.FieldByName('StimulusR').Value:=StrToInt(StimulusR.Text);
    Fields.FieldByName('StimulusG').Value:=StrToInt(StimulusG.Text);
    Fields.FieldByName('StimulusB').Value:=StrToInt(StimulusB.Text);
    if Form3.ChkUserCal.Checked then
      Fields.FieldByName('User Calibration').Value:=1
        Else
         Fields.FieldByName('User Calibration').Value:=0;
    Fields.FieldByName('Cal1R').Value:=StrToInt(Cal1R.Text);
    Fields.FieldByName('Cal1G').Value:=StrToInt(Cal1G.Text);
    Fields.FieldByName('Cal1B').Value:=StrToInt(Cal1B.Text);
    Fields.FieldByName('Cal2R').Value:=StrToInt(Cal2R.Text);
    Fields.FieldByName('Cal2G').Value:=StrToInt(Cal2G.Text);
    Fields.FieldByName('Cal2B').Value:=StrToInt(Cal2B.Text);
    if Form3.ChkPrescribed.Checked then
      Fields.FieldByName('Prescribed Solution').Value:=1
        Else
         Fields.FieldByName('Prescribed Solution').Value:=0;
    if Form3.ChkByPass.Checked then
      Fields.FieldByName('Bypass Calibration').Value:=1
        Else
         Fields.FieldByName('Bypass Calibration').Value:=0;
    Fields.FieldByName('PC1R').Value:=StrToInt(PC1R.Text);
    Fields.FieldByName('PC1G').Value:=StrToInt(PC1G.Text);
    Fields.FieldByName('PC1B').Value:=StrToInt(PC1B.Text);
    Fields.FieldByName('PC2R').Value:=StrToInt(PC2R.Text);
    Fields.FieldByName('PC2G').Value:=StrToInt(PC2G.Text);
    Fields.FieldByName('PC2B').Value:=StrToInt(PC2B.Text);
    EventDate:=DateTimeToStr(Now);
    Fields.FieldByName('Date').AsDateTime:=StrToDateTime(EventDate);
    UpDateRecord;
    Post;
   End;
end;

Procedure TForm3.UpdateFields;
Begin
  if ADOTable1.Active then
   With ADOTable1 Do
    Begin
      S1C1R.Text:=Fields.FieldByName('S1C1R').Value;
      S1C1G.Text:=Fields.FieldByName('S1C1G').Value;
      S1C1B.Text:=Fields.FieldByName('S1C1B').Value;
      S1C2R.Text:=Fields.FieldByName('S1C2R').Value;
      S1C2G.Text:=Fields.FieldByName('S1C2G').Value;
```

```
S1C2B.Text:=Fields.FieldByName('S1C2B').Value;
S2C1R.Text:=Fields.FieldByName('S2C1R').Value;
S2C1G.Text:=Fields.FieldByName('S2C1G').Value;
S2C1B.Text:=Fields.FieldByName('S2C1B').Value;
S2C2R.Text:=Fields.FieldByName('S2C2R').Value;
S2C2G.Text:=Fields.FieldByName('S2C2G').Value;
S2C2B.Text:=Fields.FieldByName('S2C2B').Value;
S3C1R.Text:=Fields.FieldByName('S3C1R').Value;
S3C1G.Text:=Fields.FieldByName('S3C1G').Value;
S3C1B.Text:=Fields.FieldByName('S3C1B').Value;
S3C2R.Text:=Fields.FieldByName('S3C2R').Value;
S3C2G.Text:=Fields.FieldByName('S3C2G').Value;
S3C2B.Text:=Fields.FieldByName('S3C2B').Value;
S4C1R.Text:=Fields.FieldByName('S4C1R').Value;
S4C1G.Text:=Fields.FieldByName('S4C1G').Value;
S4C1B.Text:=Fields.FieldByName('S4C1B').Value;
S4C2R.Text:=Fields.FieldByName('S4C2R').Value;
S4C2G.Text:=Fields.FieldByName('S4C2G').Value;
S4C2B.Text:=Fields.FieldByName('S4C2B').Value;
S5C1R.Text:=Fields.FieldByName('S5C1R').Value;
S5C1G.Text:=Fields.FieldByName('S5C1G').Value;
S5C1B.Text:=Fields.FieldByName('S5C1B').Value;
S5C2R.Text:=Fields.FieldByName('S5C2R').Value;
S5C2G.Text:=Fields.FieldByName('S5C2G').Value;
S5C2B.Text:=Fields.FieldByName('S5C2B').Value;
FormR.Text:=Fields.FieldByName('FormR').Value;
FormG.Text:=Fields.FieldByName('FormG').Value;
FormB.Text:=Fields.FieldByName('FormB').Value;
FieldR.Text:=Fields.FieldByName('FieldR').Value;
FieldG.Text:=Fields.FieldByName('FieldG').Value;
FieldB.Text:=Fields.FieldByName('FieldB').Value;
StimulusR.Text:=Fields.FieldByName('StimulusR').Value;
StimulusG.Text:=Fields.FieldByName('StimulusG').Value;
StimulusB.Text:=Fields.FieldByName('StimulusB').Value;
  if (Fields.FieldByName('Sizer').Value='1') then
     Form3.ChkSizer.Checked:=true
      Else
       Form3.ChkSizer.Checked:=False;
  if (Fields.FieldByName('User Calibration').Value='1') then
     Form3.ChkUserCal.Checked:=true
      Else
       Form3.ChkUserCal.Checked:=False;
Cal1R.Text:=Fields.FieldByName('Cal1R').Value;
Cal1G.Text:=Fields.FieldByName('Cal1G').Value;
Cal1B.Text:=Fields.FieldByName('Cal1B').Value;
Cal2R.Text:=Fields.FieldByName('Cal2R').Value;
Cal2G.Text:=Fields.FieldByName('Cal2G').Value;
Cal2B.Text:=Fields.FieldByName('Cal2B').Value;
if (Fields.FieldByName('Prescribed Solution').Value='1') then
     Form3.ChkPrescribed.Checked:=true
      Else
       Form3.ChkPrescribed.Checked:=False;
if (Fields.FieldByName('Bypass Calibration').Value='1') then
     Form3.ChkBypass.Checked:=True
```

```
            Else
              Form3.ChkBypass.Checked:=False;
       PC1R.Text:=Fields.FieldByName('PC1R').Value;
       PC1G.Text:=Fields.FieldByName('PC1G').Value;
       PC1B.Text:=Fields.FieldByName('PC1B').Value;
       PC2R.Text:=Fields.FieldByName('PC2R').Value;
       PC2G.Text:=Fields.FieldByName('PC2G').Value;
       PC2B.Text:=Fields.FieldByName('PC2B').Value;
      End;
//     UpdateColour;
End;

procedure TForm3.Button4Click(Sender: TObject);
begin
  ADOTable1.TableName:='settings';
  ADOTable1.Open;
  ADOTable1.Active:=True;
end;

procedure TForm3.Button5Click(Sender: TObject);
Var EventDate : ShortString;
begin
    If ADOTable1.Active Then
    Begin
      With ADOTable1 Do
       Begin
         Edit;
         Fields.FieldByName('Timer1').Value:=StrToInt(E1.Text);
         Fields.FieldByName('Timer2').Value:=StrToInt(E2.Text);
         Fields.FieldByName('Timer3').Value:=StrToInt(E3.Text);
         Fields.FieldByName('Mask').Value:=E4.Text;
         Fields.FieldByName('BGC1').Value:=E5.Text;
         Fields.FieldByName('BGC2').Value:=E6.Text;
         Fields.FieldByName('SizerInterval').Value:=StrToInt(E7.Text);
         Fields.FieldByName('Overide').Value:=E8.Text;
         Fields.FieldByName('Random').Value:=E9.Text;
         Fields.FieldByName('PracticeLight').Value:=E10.Text;
         Fields.FieldByName('MainLight').Value:=E11.Text;
         Fields.FieldByName('Calibrate').Value:=E12.Text;
         Fields.FieldByName('Title').Value:=E13.Text;
         Fields.FieldByName('Graphics').Value:=E14.Text;
         Fields.FieldByName('Panel').Value:=E15.Text;
         Fields.FieldByName('CIterations').Value:=StrToInt(E16.Text);
         Fields.FieldByName('TurnaroundM').Value:=StrToInt(E17.Text);
         Fields.FieldByName('MaxRuns').Value:=StrToInt(E18.Text);
         Fields.FieldByName('P1Time').Value:=StrToInt(E19.Text);
         Fields.FieldByName('P2Time').Value:=StrToInt(E20.Text);
         Fields.FieldByName('P3Time').Value:=StrToInt(E21.Text);
         Fields.FieldByName('P4Time').Value:=StrToInt(E22.Text);
         Fields.FieldByName('P5Time').Value:=StrToInt(E23.Text);
         Fields.FieldByName('FromBGR').Value:=StrToInt(E24.Text);
         Fields.FieldByName('FromBGG').Value:=StrToInt(E25.Text);
         Fields.FieldByName('FromBGB').Value:=StrToInt(E26.Text);
         Fields.FieldByName('FromFGR').Value:=StrToInt(E27.Text);
```

```
Fields.FieldByName('FromFGG').Value:=StrToInt(E28.Text);
Fields.FieldByName('FromFGB').Value:=StrToInt(E29.Text);
Fields.FieldByName('Equiliminance').Value:=StrToInt(E30.Text);
Fields.FieldByName('ToBGR').Value:=StrToInt(E31.Text);
Fields.FieldByName('ToBGG').Value:=StrToInt(E32.Text);
Fields.FieldByName('ToBGB').Value:=StrToInt(E33.Text);
Fields.FieldByName('ToFGR').Value:=StrToInt(E34.Text);
Fields.FieldByName('ToFGG').Value:=StrToInt(E35.Text);
Fields.FieldByName('ToFGB').Value:=StrToInt(E36.Text);
Fields.FieldByName('SizerFactor').Value:=StrToInt(E37.Text);
if Form3.ChkSizer.Checked then
  Fields.FieldByName('Sizer').Value:=1
    Else
       Fields.FieldByName('Sizer').Value:=0;
Fields.FieldByName('TargetWidth').Value:=StrToInt(E38.Text);
Fields.FieldByName('TargetHeight').Value:=StrToInt(E39.Text);
Fields.FieldByName('TargetLeft').Value:=StrToInt(E40.Text);
Fields.FieldByName('TargetTop').Value:=StrToInt(E41.Text);
Fields.FieldByName('S1C1R').Value:=StrToInt(S1C1R.Text);
Fields.FieldByName('S1C1G').Value:=StrToInt(S1C1G.Text);
Fields.FieldByName('S1C1B').Value:=StrToInt(S1C1B.Text);
Fields.FieldByName('S1C2R').Value:=StrToInt(S1C2R.Text);
Fields.FieldByName('S1C2G').Value:=StrToInt(S1C2G.Text);
Fields.FieldByName('S1C2B').Value:=StrToInt(S1C2B.Text);
Fields.FieldByName('S2C1R').Value:=StrToInt(S2C1R.Text);
Fields.FieldByName('S2C1G').Value:=StrToInt(S2C1G.Text);
Fields.FieldByName('S2C1B').Value:=StrToInt(S2C1B.Text);
Fields.FieldByName('S2C2R').Value:=StrToInt(S2C2R.Text);
Fields.FieldByName('S2C2G').Value:=StrToInt(S2C2G.Text);
Fields.FieldByName('S2C2B').Value:=StrToInt(S2C2B.Text);
Fields.FieldByName('S3C1R').Value:=StrToInt(S3C1R.Text);
Fields.FieldByName('S3C1G').Value:=StrToInt(S3C1G.Text);
Fields.FieldByName('S3C1B').Value:=StrToInt(S3C1B.Text);
Fields.FieldByName('S3C2R').Value:=StrToInt(S3C2R.Text);
Fields.FieldByName('S3C2G').Value:=StrToInt(S3C2G.Text);
Fields.FieldByName('S3C2B').Value:=StrToInt(S3C2B.Text);
Fields.FieldByName('S4C1R').Value:=StrToInt(S4C1R.Text);
Fields.FieldByName('S4C1G').Value:=StrToInt(S4C1G.Text);
Fields.FieldByName('S4C1B').Value:=StrToInt(S4C1B.Text);
Fields.FieldByName('S4C2R').Value:=StrToInt(S4C2R.Text);
Fields.FieldByName('S4C2G').Value:=StrToInt(S4C2G.Text);
Fields.FieldByName('S4C2B').Value:=StrToInt(S4C2B.Text);
Fields.FieldByName('S5C1R').Value:=StrToInt(S5C1R.Text);
Fields.FieldByName('S5C1G').Value:=StrToInt(S5C1G.Text);
Fields.FieldByName('S5C1B').Value:=StrToInt(S5C1B.Text);
Fields.FieldByName('S5C2R').Value:=StrToInt(S5C2R.Text);
Fields.FieldByName('S5C2G').Value:=StrToInt(S5C2G.Text);
Fields.FieldByName('S5C2B').Value:=StrToInt(S5C2B.Text);
Fields.FieldByName('S1T').Value:=StrToInt(S1T.Text);
Fields.FieldByName('S2T').Value:=StrToInt(S2T.Text);
Fields.FieldByName('S3T').Value:=StrToInt(S3T.Text);
Fields.FieldByName('S4T') .Value:=StrToInt(S4T.Text);
Fields.FieldByName('S5T').Value:=StrToInt(S5T.Text);
Fields.FieldByName('UVM') .Value:=StrToInt(UVM.Text);
```

```
        Fields.FieldByName('LVM') .Value:=StrToInt(LVM.Text);
        Fields.FieldByName('AVM').Value:=StrToInt(AVM.Text);
        Fields.FieldByName('FormR').Value:=StrToInt(FormR.Text);
        Fields.FieldByName('FormG').Value:=StrToInt(FormG.Text);
        Fields.FieldByName('FormB').Value:=StrToInt(FormB.Text);
        Fields.FieldByName('FieldR').Value:=StrToInt(FieldR.Text);
        Fields.FieldByName('FieldG').Value:=StrToInt(FieldG.Text);
        Fields.FieldByName('FieldB').Value:=StrToInt(FieldB.Text);
        Fields.FieldByName('StimulusR').Value:=StrToInt(StimulusR.Text);
        Fields.FieldByName('StimulusG').Value:=StrToInt(StimulusG.Text);
        Fields.FieldByName('StimulusB').Value:=StrToInt(StimulusB.Text);
        if Form3.ChkUserCal.Checked then
          Fields.FieldByName('User Calibration').Value:=1
            Else
              Fields.FieldByName('User Calibration').Value:=0;
        Fields.FieldByName('Cal1R').Value:=StrToInt(Cal1R.Text);
        Fields.FieldByName('Cal1G').Value:=StrToInt(Cal1G.Text);
        Fields.FieldByName('Cal1B').Value:=StrToInt(Cal1B.Text);
        Fields.FieldByName('Cal2R').Value:=StrToInt(Cal2R.Text);
        Fields.FieldByName('Cal2G').Value:=StrToInt(Cal2G.Text);
        Fields.FieldByName('Cal2B').Value:=StrToInt(Cal2B.Text);
        if Form3.ChkPrescribed.Checked then
          Fields.FieldByName('Prescribed Solution').Value:=1
            Else
              Fields.FieldByName('Prescribed Solution').Value:=0;
        if Form3.ChkByPass.Checked then
           Fields.FieldByName('Bypass Calibration').Value:=1
            Else
              Fields.FieldByName('Bypass Calibration').Value:=0;
        Fields.FieldByName('PC1R').Value:=StrToInt(PC1R.Text);
        Fields.FieldByName('PC1G').Value:=StrToInt(PC1G.Text);
        Fields.FieldByName('PC1B').Value:=StrToInt(PC1B.Text);
        Fields.FieldByName('PC2R').Value:=StrToInt(PC2R.Text);
        Fields.FieldByName('PC2G').Value:=StrToInt(PC2G.Text);
        Fields.FieldByName('PC2B').Value:=StrToInt(PC2B.Text);
        EventDate:=DateTimeToStr(Now);
        Fields.FieldByName('Date').AsDateTime:=StrToDateTime(EventDate);
        UpDateRecord;
      Post;
    End;
  End;
end;

procedure TForm3.Button6Click(Sender: TObject);
begin
  ADOTable1.Active:=False;
  ADOTable1.Close;
end;

procedure TForm3.ChkBypassClick(Sender: TObject);
begin
   If ADOTable1.Active Then
    Begin
      With ADOTable1 Do
```

```
      Begin
        Edit;
        if ChkByPass.Checked then
          Fields.FieldByName('Bypass Calibration').Value:=1
            Else
              Fields.FieldByName('Bypass Calibration').Value:=0;
      End;
   End;
end;

procedure TForm3.ChkPrescribedClick(Sender: TObject);
begin
  If ADOTable1.Active Then
   Begin
     With ADOTable1 Do
      Begin
        Edit;
        if ChkPrescribed.Checked then
          Fields.FieldByName('Prescribed Solution').Value:=1
            Else
              Fields.FieldByName('Prescribed Solution').Value:=0;
      End;
   End;
end;

procedure TForm3.ChkSizerClick(Sender: TObject);
begin
  If ADOTable1.Active Then
   Begin
     With ADOTable1 Do
      Begin
        Edit;
        if ChkSizer.Checked then
          Fields.FieldByName('Sizer').Value:=1
            Else
              Fields.FieldByName('Sizer').Value:=0;
      End;
   End;
end;

procedure TForm3.ChkUserCalClick(Sender: TObject);
begin
  If ADOTable1.Active Then
   Begin
     With ADOTable1 Do
      Begin
        Edit;
        if ChkUserCal.Checked then
          Fields.FieldByName('User Calibration').Value:=1
            Else
              Fields.FieldByName('User Calibration').Value:=0;
      End;
   End;
end;
```

```
procedure TForm3.ComboBox1Change(Sender: TObject);
begin
 If ADOConnection1.Connected Then
 Begin
   ADOConnection1.Close;
   ADOConnection1.Connected:=False;
 End;
  if ComboBox1.Text='Local Vista'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE2;Initial Catalog=uctexp1';
  if ComboBox1.Text='Mansfield'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE1;Initial Catalog=uctexp1';
  if ComboBox1.Text='Work PC'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE3;Initial Catalog=uctexp1';
  if ComboBox1.Text='UCT Server'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE5;Initial Catalog=uctexp1';
  if ComboBox1.Text='Windows 7 PC'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=DOUGAS-HP;Initial Catalog=uctexp1';
  ADOCOmmand1.ConnectionString:=CString;
  Try
    ADOConnection1.ConnectionString:=CString;
    ADOConnection1.Connected:=True;
  Except
    ShowMessage('Cannot connect');
  End;
  Button1.Enabled:=ADOConnection1.Connected;
  Button4.Enabled:=ADOConnection1.Connected;
  Button6.Enabled:=ADOConnection1.Connected;
  BtnOpenFile.Enabled:=ADOConnection1.Connected;
  BtnSaveFile.Enabled:=ADOConnection1.Connected;
end;

procedure TForm3.ComboBox2Change(Sender: TObject);
begin
 If ADOConnection1.Connected Then
 Begin
   ADOConnection1.Close;
   ADOConnection1.Connected:=False;
 End;
  if ComboBox1.Text='Local Vista'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE2;Initial Catalog=uctexp1';
  if ComboBox2.Text='Mansfield'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE1;Initial Catalog=uctexp1';
  if ComboBox2.Text='Work PC'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE3;Initial Catalog=uctexp1';
  if ComboBox2.Text='UCT Server'
```

```
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE5;Initial Catalog=uctexp1';
  if ComboBox2.Text='Windows 7 PC'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=DOUGAS-HP;Initial Catalog=uctexp1';
  ADOCOmmand1.ConnectionString:=CString;
  Try
    ADOConnection1.ConnectionString:=CString;
    ADOConnection1.Connected:=True;
  Except
    ShowMessage('Cannot connect');
  End;
  Button2.Enabled:=ADOConnection1.Connected;
  Button3.Enabled:=ADOConnection1.Connected;
  Button5.Enabled:=ADOConnection1.Connected;
  Button4.Enabled:=ADOConnection1.Connected;
  Button6.Enabled:=ADOConnection1.Connected;
end;

procedure TForm3.DataSource1DataChange(Sender: TObject; Field: TField);
begin
  if DBSizer.Field.Value='0' then
    ChkSizer.Checked:=false;
  if DBSizer.Field.Value='1' then
    ChkSizer.Checked:=true;
  if DBCal.Field.Value='0' then
    ChkUserCal.Checked:=false;
  if DBCal.Field.Value='1' then
    ChkUserCal.Checked:=True;
  if DBPrescribed.Field.Value='0' then
    ChkPrescribed.Checked:=false;
  if DBPrescribed.Field.Value='1' then
    ChkPrescribed.Checked:=True;
  if DbBypass.Field.Value='1' then
    ChkByPass.Checked:=True;
  if DbBypass.Field.Value='0' then
    ChkByPass.Checked:=False;
    UpdateColour;
//    UpdateFields;
end;

procedure TForm3.DBGrid1MouseLeave(Sender: TObject);
begin
//  UpdateFields;
end;

procedure TForm3.FormClose(Sender: TObject; var Action: TCloseAction);
begin
 ADOTable1.Active:=false;
 ADoTable1.Close;
 ADOCommand1.Free;
 ADOCommand2.Free;
 ADOConnection1.Close;
 ADOConnection2.Close;
```

```
end;

procedure TForm3.FormShow(Sender: TObject);
begin
//  UpdateFields;
end;

Procedure TForm3.UpdateColour;
Var RR,GG,BB : Byte;
Begin
  RR:=StrToInt(S1C1R.Field.Value);
  GG:=StrToInt(S1C1G.Field.Value);
  BB:=StrToInt(S1C1B.Field.Value);
  S1C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S1C2R.Field.Value);
  GG:=StrToInt(S1C2G.Field.Value);
  BB:=StrToInt(S1C2B.Field.Value);
  S1C2.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S2C1R.Field.Value);
  GG:=StrToInt(S2C1G.Field.Value);
  BB:=StrToInt(S2C1B.Field.Value);
  S2C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S2C2R.Field.Value);
  GG:=StrToInt(S2C2G.Field.Value);
  BB:=StrToInt(S2C2B.Field.Value);
  S2C2.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S3C1R.Field.Value);
  GG:=StrToInt(S3C1G.Field.Value);
  BB:=StrToInt(S3C1B.Field.Value);
  S3C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S3C2R.Field.Value);
  GG:=StrToInt(S3C2G.Field.Value);
  BB:=StrToInt(S3C2B.Field.Value);
  S3C2.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S4C1R.Field.Value);
  GG:=StrToInt(S4C1G.Field.Value);
  BB:=StrToInt(S4C1B.Field.Value);
  S4C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S4C2R.Field.Value);
  GG:=StrToInt(S4C2G.Field.Value);
  BB:=StrToInt(S4C2B.Field.Value);
  S4C2.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S5C1R.Field.Value);
  GG:=StrToInt(S5C1G.Field.Value);
  BB:=StrToInt(S5C1B.Field.Value);
  S5C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S5C2R.Field.Value);
  GG:=StrToInt(S5C2G.Field.Value);
  BB:=StrToInt(S5C2B.Field.Value);
  S5C2.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToINt(FormR.Field.Value);
  GG:=StrToINt(FormG.Field.Value);
  BB:=StrToINt(FormB.Field.Value);
  FormR.Color:=RGB(RR,GG,BB);
```

```
  FormG.Color:=RGB(RR,GG,BB);
  FormB.Color:=RGB(RR,GG,BB);
  RR:=StrToINt(FieldR.Field.Value);
  GG:=StrToINt(FieldG.Field.Value);
  BB:=StrToINt(FieldB.Field.Value);
  FieldR.Color:=RGB(RR,GG,BB);
  FieldG.Color:=RGB(RR,GG,BB);
  FieldB.Color:=RGB(RR,GG,BB);
  RR:=StrToINt(StimulusR.Field.Value);
  GG:=StrToINt(StimulusG.Field.Value);
  BB:=StrToINt(StimulusB.Field.Value);
  StimulusR.Color:=RGB(RR,GG,BB);
  StimulusG.Color:=RGB(RR,GG,BB);
  StimulusB.Color:=RGB(RR,GG,BB);
  RR:=StrToINt(PC1R.Field.Value);
  GG:=StrToINt(PC1G.Field.Value);
  BB:=StrToINt(PC1B.Field.Value);
  PC1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToINt(PC2R.Field.Value);
  GG:=StrToINt(PC2G.Field.Value);
  BB:=StrToINt(PC2B.Field.Value);
  PC2.Brush.Color:=RGB(RR,GG,BB);
End;

procedure TForm3.T20S1Click(Sender: TObject);
Var RR,GG,BB : Byte;
begin
  RR:=StrToInt(S1C1R.Text);
  GG:=StrToInt(S1C1G.Text);
  BB:=StrToInt(S1C1B.Text);
  S1C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S1C2R.Text);
  GG:=StrToInt(S1C2G.Text);
  BB:=StrToInt(S1C2B.Text);
  S1C2.Brush.Color:=RGB(RR,GG,BB);
end;

procedure TForm3.T20S2Click(Sender: TObject);
Var RR,GG,BB : Byte;
begin
  RR:=StrToInt(S2C1R.Text);
  GG:=StrToInt(S2C1G.Text);
  BB:=StrToInt(S2C1B.Text);
  S2C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S2C2R.Text);
  GG:=StrToInt(S2C2G.Text);
  BB:=StrToInt(S2C2B.Text);
  S2C2.Brush.Color:=RGB(RR,GG,BB);
end;

procedure TForm3.T20S3Click(Sender: TObject);
Var RR,GG,BB : Byte;
begin
  RR:=StrToInt(S3C1R.Text);
```

```
  GG:=StrToInt(S3C1G.Text);
  BB:=StrToInt(S3C1B.Text);
  S3C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S3C2R.Text);
  GG:=StrToInt(S3C2G.Text);
  BB:=StrToInt(S3C2B.Text);
  S3C2.Brush.Color:=RGB(RR,GG,BB);
end;

procedure TForm3.T20S4Click(Sender: TObject);
Var RR,GG,BB : Byte;
begin
  RR:=StrToInt(S4C1R.Text);
  GG:=StrToInt(S4C1G.Text);
  BB:=StrToInt(S4C1B.Text);
  S4C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S4C2R.Text);
  GG:=StrToInt(S4C2G.Text);
  BB:=StrToInt(S4C2B.Text);
  S4C2.Brush.Color:=RGB(RR,GG,BB);
end;

procedure TForm3.T20S5Click(Sender: TObject);
Var RR,GG,BB : Byte;
begin
  RR:=StrToInt(S5C1R.Text);
  GG:=StrToInt(S5C1G.Text);
  BB:=StrToInt(S5C1B.Text);
  S5C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S5C2R.Text);
  GG:=StrToInt(S5C2G.Text);
  BB:=StrToInt(S5C2B.Text);
  S5C2.Brush.Color:=RGB(RR,GG,BB);
end;

end.
```

```
unit Unit3;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, ADODB, DB, ExtCtrls, DBCtrls, Grids, DBGrids,
  AdvSmoothStatusIndicator, Buttons, Mask, ImgList, AdvSmoothTabPager, ComCtrls,
  ExtDlgs, AdvSmoothPanel, AdvSmoothProgressBar;

type
  TForm3 = class(TForm)
    AdvSmoothTabPager1: TAdvSmoothTabPager;
    AdvSmoothTabPager11: TAdvSmoothTabPage;
    AdvSmoothTabPager12: TAdvSmoothTabPage;
    AdvSmoothTabPager13: TAdvSmoothTabPage;
    ComboBox1: TComboBox;
    Button1: TButton;
    GroupBox1: TGroupBox;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label17: TLabel;
    Label18: TLabel;
    Label19: TLabel;
    Label20: TLabel;
    Label21: TLabel;
    Label22: TLabel;
    Label23: TLabel;
    Label24: TLabel;
    Label25: TLabel;
    Label26: TLabel;
    Label27: TLabel;
    Label28: TLabel;
    Label29: TLabel;
    Label30: TLabel;
    Label31: TLabel;
    Label32: TLabel;
    Label33: TLabel;
    Label34: TLabel;
```

Label35: TLabel;
Label36: TLabel;
Label37: TLabel;
Label38: TLabel;
Label39: TLabel;
Label40: TLabel;
Label41: TLabel;
T20S1: TAdvSmoothStatusIndicator;
T20S2: TAdvSmoothStatusIndicator;
T20S3: TAdvSmoothStatusIndicator;
T20S4: TAdvSmoothStatusIndicator;
T20S5: TAdvSmoothStatusIndicator;
Label42: TLabel;
Label43: TLabel;
Label44: TLabel;
Label45: TLabel;
Label46: TLabel;
Label47: TLabel;
S1C2: TShape;
S2C2: TShape;
S3C2: TShape;
S1C1: TShape;
S2C1: TShape;
S3C1: TShape;
S4C1: TShape;
S4C2: TShape;
S5C1: TShape;
S5C2: TShape;
Label48: TLabel;
Label49: TLabel;
Label50: TLabel;
PC1: TShape;
PC2: TShape;
DBCal1: TDBText;
DBPrescribed1: TDBText;
Button2: TButton;
ComboBox2: TComboBox;
E1: TDBEdit;
E2: TDBEdit;
E3: TDBEdit;
E4: TDBEdit;
E5: TDBEdit;
E6: TDBEdit;
E7: TDBEdit;
E37: TDBEdit;
E8: TDBEdit;
E9: TDBEdit;
E10: TDBEdit;
E11: TDBEdit;
E12: TDBEdit;
E13: TDBEdit;
E14: TDBEdit;
AVM: TDBEdit;
E15: TDBEdit;

E16: TDBEdit;
E17: TDBEdit;
E18: TDBEdit;
E19: TDBEdit;
E20: TDBEdit;
E21: TDBEdit;
E22: TDBEdit;
E23: TDBEdit;
E24: TDBEdit;
E25: TDBEdit;
E26: TDBEdit;
E27: TDBEdit;
E28: TDBEdit;
E29: TDBEdit;
E30: TDBEdit;
UVM: TDBEdit;
E38: TDBEdit;
E39: TDBEdit;
E40: TDBEdit;
E41: TDBEdit;
E31: TDBEdit;
E32: TDBEdit;
E33: TDBEdit;
E34: TDBEdit;
E35: TDBEdit;
E36: TDBEdit;
LVM: TDBEdit;
S1C1R: TDBEdit;
S1C1G: TDBEdit;
S1C1B: TDBEdit;
S2C1R: TDBEdit;
S2C1G: TDBEdit;
S2C1B: TDBEdit;
S3C1R: TDBEdit;
S3C1G: TDBEdit;
S3C1B: TDBEdit;
S4C1R: TDBEdit;
S4C1G: TDBEdit;
S4C1B: TDBEdit;
S5C1R: TDBEdit;
S5C1G: TDBEdit;
S5C1B: TDBEdit;
S1C2R: TDBEdit;
S1C2G: TDBEdit;
S1C2B: TDBEdit;
S2C2R: TDBEdit;
S2C2G: TDBEdit;
S2C2B: TDBEdit;
S3C2R: TDBEdit;
S3C2G: TDBEdit;
S3C2B: TDBEdit;
S4C2R: TDBEdit;
S4C2G: TDBEdit;
S4C2B: TDBEdit;

S5C2R: TDBEdit;
S5C2G: TDBEdit;
S5C2B: TDBEdit;
s1t: TDBEdit;
S2T: TDBEdit;
S3T: TDBEdit;
S4T: TDBEdit;
S5T: TDBEdit;
FormR: TDBEdit;
FormG: TDBEdit;
FormB: TDBEdit;
FieldR: TDBEdit;
FieldG: TDBEdit;
FieldB: TDBEdit;
StimulusR: TDBEdit;
StimulusG: TDBEdit;
StimulusB: TDBEdit;
Cal1R: TDBEdit;
Cal1G: TDBEdit;
Cal1B: TDBEdit;
Cal2R: TDBEdit;
Cal2G: TDBEdit;
Cal2B: TDBEdit;
PC1R: TDBEdit;
PC2R: TDBEdit;
PC1G: TDBEdit;
PC2G: TDBEdit;
PC1B: TDBEdit;
PC2B: TDBEdit;
ChkPrescribed: TCheckBox;
ChkUserCal: TCheckBox;
ChkSizer: TCheckBox;
DBSizer: TDBEdit;
DBPrescribed: TDBEdit;
DBCal: TDBEdit;
ChkBypass: TCheckBox;
DBbyPass: TDBEdit;
DBGrid1: TDBGrid;
DBNavigator1: TDBNavigator;
Button3: TButton;
Button4: TButton;
Button5: TButton;
Button6: TButton;
BitBtn1: TBitBtn;
BtnOpenFile: TButton;
BtnSaveFile: TButton;
ADOConnection1: TADOConnection;
ADOCommand1: TADOCommand;
DataSource1: TDataSource;
ADOTable1: TADOTable;
ADOConnection2: TADOConnection;
ADOCommand2: TADOCommand;
OpenDialog1: TOpenDialog;
SaveDialog1: TSaveDialog;

ImageList1: TImageList;
DBM1: TDBRichEdit;
OpenDialog2: TOpenDialog;
DBM2: TDBRichEdit;
M2Update: TButton;
M1Open: TButton;
AdvSmoothTabPage1: TAdvSmoothTabPage;
DBM3: TDBRichEdit;
M3Update: TButton;
AdvSmoothTabPage2: TAdvSmoothTabPage;
DBM4: TDBRichEdit;
M4Update: TButton;
AdvSmoothTabPage3: TAdvSmoothTabPage;
DBM5: TDBRichEdit;
M5Update: TButton;
AdvSmoothTabPage4: TAdvSmoothTabPage;
DBM6: TDBRichEdit;
M6Update: TButton;
AdvSmoothTabPage5: TAdvSmoothTabPage;
DBM7: TDBRichEdit;
M7Update: TButton;
AdvSmoothTabPage6: TAdvSmoothTabPage;
DBMA: TDBRichEdit;
MAUpdate: TButton;
AdvSmoothTabPage7: TAdvSmoothTabPage;
Label51: TLabel;
Label52: TLabel;
FlowIngroup: TFlowPanel;
DBIn1: TDBImage;
DBIn2: TDBImage;
DBIn3: TDBImage;
DBIn4: TDBImage;
DBIn5: TDBImage;
DBIn6: TDBImage;
DBIn7: TDBImage;
DBIn8: TDBImage;
DBIn9: TDBImage;
DBIn10: TDBImage;
FlowPanel1: TFlowPanel;
DBOut1: TDBImage;
DBOut2: TDBImage;
DBOut3: TDBImage;
DBOut4: TDBImage;
DBOut5: TDBImage;
DBOut6: TDBImage;
DBOut7: TDBImage;
DBOut8: TDBImage;
DBOut9: TDBImage;
DBOut10: TDBImage;
OpenPictureDialog1: TOpenPictureDialog;
ADOConnection3: TADOConnection;
Table2: TADOTable;
DataSource2: TDataSource;
TableIn3to5: TADOTable;

DataSourceI3to5: TDataSource;
DataSourceI6to10: TDataSource;
Table6to10: TADOTable;
TableO1to3: TADOTable;
TableO4to6: TADOTable;
Table7to10: TADOTable;
DataSourceO1to3: TDataSource;
DataSourceO4to6: TDataSource;
DataSourceO7to10: TDataSource;
DBNavigator3: TDBNavigator;
TableIn3to5IDX: TAutoIncField;
TableIn3to5IN3: TBlobField;
TableIn3to5IN4: TBlobField;
TableIn3to5IN5: TBlobField;
Table6to10IDX: TAutoIncField;
Table6to10IN6: TBlobField;
Table6to10IN7: TBlobField;
Table6to10IN8: TBlobField;
Table6to10IN9: TBlobField;
Table6to10IN10: TBlobField;
TableO1to3idx: TAutoIncField;
TableO1to3out1: TBlobField;
TableO1to3out2: TBlobField;
TableO1to3out3: TBlobField;
TableO4to6idx: TAutoIncField;
TableO4to6out4: TBlobField;
TableO4to6out5: TBlobField;
TableO4to6out6: TBlobField;
Table7to10idx: TAutoIncField;
Table7to10out7: TBlobField;
Table7to10out8: TBlobField;
Table7to10out9: TBlobField;
Table7to10out10: TBlobField;
GroupBox2: TGroupBox;
ChcT1: TCheckBox;
ChcT2: TCheckBox;
ChcT3: TCheckBox;
ChcT4: TCheckBox;
ChcT5: TCheckBox;
ChcAuto: TCheckBox;
AdvSmoothTabPage8: TAdvSmoothTabPage;
ADOCommand3: TADOCommand;
ComboBox3: TComboBox;
LEDConnect: TAdvSmoothStatusIndicator;
Button7: TButton;
ProgressBar1: TProgressBar;
BtnConnectBlobs: TButton;
AdvSmoothPanel1: TAdvSmoothPanel;
DBParameters: TDBMemo;
LblParameters: TLabel;
DBTimes: TDBMemo;
LblTimes: TLabel;
AdvSmoothPanel2: TAdvSmoothPanel;
DBSubUnP: TDBMemo;

DBSubGUItems: TDBMemo;
LblOutGroup: TLabel;
DBSubPItems: TDBMemo;
Label54: TLabel;
Label55: TLabel;
DBSubGPM: TDBMemo;
DBPleasantW: TDBMemo;
LblBadItems: TLabel;
DBUnpleasntW: TDBMemo;
Label57: TLabel;
DBIngroup: TDBMemo;
Label58: TLabel;
DBOutGroup: TDBMemo;
Label59: TLabel;
Label53: TLabel;
Label60: TLabel;
Label56: TLabel;
AdvSmoothPanel3: TAdvSmoothPanel;
Label64: TLabel;
DBConcepts: TDBMemo;
DBVerdicts: TDBMemo;
Label61: TLabel;
DBNavigator4: TDBNavigator;
DataSource3: TDataSource;
TableTextFields: TADOTable;
TableTextFieldsIDX: TAutoIncField;
TableTextFieldstimes: TStringField;
TableTextFieldsparameters: TStringField;
TableTextFieldsPWE: TStringField;
TableTextFieldsUWE: TStringField;
TableTextFieldsIGItems: TStringField;
TableTextFieldsOGItems: TStringField;
TableTextFieldsSubPItems: TStringField;
TableTextFieldsSubUItems: TStringField;
TableTextFieldsSubPOItems: TStringField;
TableTextFieldsSubUOItems: TStringField;
TableTextFieldsconcepts: TStringField;
TableTextFieldsverdicts: TStringField;
ChcT6: TCheckBox;
BtnCreateIAT: TButton;
Panel1: TPanel;
AdvSmoothTabPage9: TAdvSmoothTabPage;
FlowPanel2: TFlowPanel;
DBImageP1: TDBImage;
DBImageP2: TDBImage;
DBImageP3: TDBImage;
DBImageP4: TDBImage;
DBImageP5: TDBImage;
DBImageP6: TDBImage;
DBImageP7: TDBImage;
DBImageP8: TDBImage;
DBImageP9: TDBImage;
DBImageP10: TDBImage;
FlowPanel3: TFlowPanel;

DBImageM1: TDBImage;
DBImageM2: TDBImage;
DBImageM3: TDBImage;
DBImageM4: TDBImage;
DBImageM5: TDBImage;
DBImageM6: TDBImage;
DBImageM7: TDBImage;
DBImageM8: TDBImage;
DBImageM9: TDBImage;
DBImageM10: TDBImage;
GroupBox3: TGroupBox;
ChcPSub1to3: TCheckBox;
ChcPSub4to6: TCheckBox;
ChcPSub7to10: TCheckBox;
ChcSubU1to3: TCheckBox;
ChcUSub4to6: TCheckBox;
ChcAutoSub: TCheckBox;
ChcUSub7to10: TCheckBox;
TablePSub1to3: TADOTable;
TablePSub4to6: TADOTable;
TablePSub7to10: TADOTable;
TableUSub1to3: TADOTable;
TableUSub4to6: TADOTable;
TableUSub7to10: TADOTable;
DataSourcePSub1to3: TDataSource;
DataSourcePSub4to6: TDataSource;
DataSourcePSub7to10: TDataSource;
DataSourceUSub1to3: TDataSource;
DataSourceUSub4to6: TDataSource;
DataSourceUSub7to10: TDataSource;
DBNavigator5: TDBNavigator;
Label62: TLabel;
Label63: TLabel;
TablePSub4to6IDX: TAutoIncField;
TablePSub4to6subp4: TBlobField;
TablePSub4to6subp5: TBlobField;
TablePSub4to6subp6: TBlobField;
TablePSub7to10IDX: TAutoIncField;
TablePSub7to10subp7: TBlobField;
TablePSub7to10subp8: TBlobField;
TablePSub7to10subp9: TBlobField;
TablePSub7to10subp10: TBlobField;
TableUSub1to3IDX: TAutoIncField;
TableUSub1to3subu1: TBlobField;
TableUSub1to3subu2: TBlobField;
TableUSub1to3subu3: TBlobField;
TableUSub4to6IDX: TAutoIncField;
TableUSub4to6subu4: TBlobField;
TableUSub4to6subu5: TBlobField;
TableUSub4to6subu6: TBlobField;
TableUSub7to10IDX: TAutoIncField;
TableUSub7to10subu7: TBlobField;
TableUSub7to10subu8: TBlobField;
TableUSub7to10subu9: TBlobField;

TableUSub7to10subu10: TBlobField;
ProgressBar2: TProgressBar;
AdvSmoothTabPage10: TAdvSmoothTabPage;
FlowPanel4: TFlowPanel;
DBImageP11: TDBImage;
DBImageP12: TDBImage;
DBImageP13: TDBImage;
DBImageP14: TDBImage;
DBImageP15: TDBImage;
DBImageP16: TDBImage;
DBImageP17: TDBImage;
DBImageP18: TDBImage;
DBImageP19: TDBImage;
DBImageP20: TDBImage;
FlowPanel5: TFlowPanel;
DBImageM11: TDBImage;
DBImageM12: TDBImage;
DBImageM13: TDBImage;
DBImageM14: TDBImage;
DBImageM15: TDBImage;
DBImageM16: TDBImage;
DBImageM17: TDBImage;
DBImageM18: TDBImage;
DBImageM19: TDBImage;
DBImageM20: TDBImage;
GroupBox4: TGroupBox;
ChcPSub1to3B: TCheckBox;
ChcPSub4to6B: TCheckBox;
ChcPSub7to10B: TCheckBox;
ChcSubU1to3B: TCheckBox;
ChcUSub4to6B: TCheckBox;
ChcAutoSub2: TCheckBox;
ChcUSub7to10B: TCheckBox;
Label65: TLabel;
Label66: TLabel;
subp11to13: TADOTable;
DataSourcesubp11to13: TDataSource;
subp14to16: TADOTable;
AutoIncField2: TAutoIncField;
DataSourcesubp14to16: TDataSource;
subp17to20: TADOTable;
subu10to13: TADOTable;
DataSourcesubu10to13: TDataSource;
subu14to16: TADOTable;
DataSourcesubu14to16: TDataSource;
subu17to20: TADOTable;
DataSourcesubu17to20: TDataSource;
AdvSmoothTabPage11: TAdvSmoothTabPage;
FlowPanel6: TFlowPanel;
DBImageS1: TDBImage;
DBImageS2: TDBImage;
DBImageS3: TDBImage;
DBImageS4: TDBImage;
DBImageS5: TDBImage;

DBImageS6: TDBImage;
DBImageS7: TDBImage;
DBImageS8: TDBImage;
DBImageS9: TDBImage;
DBImageS10: TDBImage;
FlowPanel7: TFlowPanel;
DBImageS1B: TDBImage;
DBImageS2B: TDBImage;
DBImageS3B: TDBImage;
DBImageS4B: TDBImage;
DBImageS5B: TDBImage;
DBImageS6B: TDBImage;
DBImageS7B: TDBImage;
DBImageS8B: TDBImage;
DBImageS9B: TDBImage;
DBImageS10B: TDBImage;
GroupBox5: TGroupBox;
ChcS1to3: TCheckBox;
ChcS4to6: TCheckBox;
ChcS7to10: TCheckBox;
ChcS1to3B: TCheckBox;
ChcS4to6B: TCheckBox;
ChcAutoS: TCheckBox;
ChcS7to10B: TCheckBox;
S1to3: TADOTable;
DataSourceS1TO3: TDataSource;
S4To6: TADOTable;
DataSourceS4To6: TDataSource;
S7TO10: TADOTable;
DataSourceS7TO10: TDataSource;
S1TO3B: TADOTable;
DataSourceS1TO3B: TDataSource;
S4TO6B: TADOTable;
DataSourceS4TO6B: TDataSource;
S7TO10B: TADOTable;
DataSourceS7TO10B: TDataSource;
ProgressBar3: TProgressBar;
ProgressBar4: TProgressBar;
subp14to16subp14: TBlobField;
subp14to16subp15: TBlobField;
subp14to16subp16: TBlobField;
DataSourcesubp17to20: TDataSource;
subp17to20IDX: TAutoIncField;
subp17to20subp17: TBlobField;
subp17to20subp18: TBlobField;
subp17to20subp19: TBlobField;
subp17to20subp20: TBlobField;
subu10to13IDX: TAutoIncField;
subu10to13subu11: TBlobField;
subu10to13subu12: TBlobField;
subu10to13subu13: TBlobField;
subu14to16IDX: TAutoIncField;
subu14to16subu14: TBlobField;
subu14to16subu15: TBlobField;

```
subu14to16subu16: TBlobField;
subu17to20IDX: TAutoIncField;
subu17to20subu17: TBlobField;
subu17to20subu18: TBlobField;
subu17to20subu19: TBlobField;
subu17to20subu20: TBlobField;
S1to3IDX: TAutoIncField;
S1to3subp11: TBlobField;
S1to3subp12: TBlobField;
S1to3subp13: TBlobField;
S4To6IDX: TAutoIncField;
S4To6subp14: TBlobField;
S4To6subp15: TBlobField;
S4To6subp16: TBlobField;
S7TO10IDX: TAutoIncField;
S7TO10subp17: TBlobField;
S7TO10subp18: TBlobField;
S7TO10subp19: TBlobField;
S7TO10subp20: TBlobField;
S4TO6BIDX: TAutoIncField;
S4TO6Bsubu14: TBlobField;
S4TO6Bsubu15: TBlobField;
S4TO6Bsubu16: TBlobField;
S1TO3BIDX: TAutoIncField;
S1TO3Bsubu11: TBlobField;
S1TO3Bsubu12: TBlobField;
S1TO3Bsubu13: TBlobField;
S7TO10BIDX: TAutoIncField;
S7TO10Bsubu17: TBlobField;
S7TO10Bsubu18: TBlobField;
S7TO10Bsubu19: TBlobField;
S7TO10Bsubu20: TBlobField;
TablePSub1to3IDX: TAutoIncField;
TablePSub1to3subp1: TBlobField;
TablePSub1to3subp2: TBlobField;
TablePSub1to3subp3: TBlobField;
subp11to13IDX: TAutoIncField;
subp11to13subp11: TBlobField;
subp11to13subp12: TBlobField;
subp11to13subp13: TBlobField;
DBNavigator2: TDBNavigator;
DBNavigator6: TDBNavigator;
Label67: TLabel;
Label68: TLabel;
DBQuestions: TDBMemo;
Label69: TLabel;
TableTextFieldsquestions: TStringField;
Label70: TLabel;
Label71: TLabel;
Label72: TLabel;
Label73: TLabel;
procedure ComboBox1Change(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure ComboBox2Change(Sender: TObject);
```

```
procedure Button2Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Button4Click(Sender: TObject);
procedure T20S1Click(Sender: TObject);
procedure T20S2Click(Sender: TObject);
procedure T20S3Click(Sender: TObject);
procedure T20S4Click(Sender: TObject);
procedure T20S5Click(Sender: TObject);
Procedure UpdateColour;
Procedure UpdateFields;
procedure DBGrid1MouseLeave(Sender: TObject);
procedure Button5Click(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
procedure Button6Click(Sender: TObject);
procedure FormShow(Sender: TObject);
procedure BitBtn1Click(Sender: TObject);
procedure DataSource1DataChange(Sender: TObject; Field: TField);
procedure ChkSizerClick(Sender: TObject);
procedure ChkUserCalClick(Sender: TObject);
procedure ChkPrescribedClick(Sender: TObject);
procedure ChkBypassClick(Sender: TObject);
procedure BtnSaveFileClick(Sender: TObject);
procedure BtnOpenFileClick(Sender: TObject);
procedure M1OpenClick(Sender: TObject);
procedure M2UpdateClick(Sender: TObject);
procedure M3UpdateClick(Sender: TObject);
procedure M4UpdateClick(Sender: TObject);
procedure M6UpdateClick(Sender: TObject);
procedure M5UpdateClick(Sender: TObject);
procedure M7UpdateClick(Sender: TObject);
procedure MAUpdateClick(Sender: TObject);
procedure DBIn1DblClick(Sender: TObject);
procedure DBIn2DblClick(Sender: TObject);
procedure DBIn3DblClick(Sender: TObject);
procedure DBIn4DblClick(Sender: TObject);
procedure DBIn5DblClick(Sender: TObject);
procedure DBIn6DblClick(Sender: TObject);
procedure DBIn7DblClick(Sender: TObject);
procedure DBIn8DblClick(Sender: TObject);
procedure DBIn9DblClick(Sender: TObject);
procedure DBIn10DblClick(Sender: TObject);
procedure DBOut1DblClick(Sender: TObject);
procedure DBOut2DblClick(Sender: TObject);
procedure DBOut3DblClick(Sender: TObject);
procedure DBOut4DblClick(Sender: TObject);
procedure DBOut5DblClick(Sender: TObject);
procedure DBOut6DblClick(Sender: TObject);
procedure DBOut7DblClick(Sender: TObject);
procedure DBOut8DblClick(Sender: TObject);
procedure DBOut9DblClick(Sender: TObject);
procedure DBOut10DblClick(Sender: TObject);
procedure ComboBox3Change(Sender: TObject);
procedure ADOConnection3AfterConnect(Sender: TObject);
procedure Button7Click(Sender: TObject);
```

```
procedure ChcT1Click(Sender: TObject);
procedure ChcT2Click(Sender: TObject);
procedure ChcT3Click(Sender: TObject);
procedure ChcT4Click(Sender: TObject);
procedure ChcT5Click(Sender: TObject);
procedure ADOConnection3AfterDisconnect(Sender: TObject);
procedure Table2AfterOpen(DataSet: TDataSet);
Procedure Connecting;
procedure BtnConnectBlobsClick(Sender: TObject);
procedure BtnCreateIATClick(Sender: TObject);
Procedure CreateIAT_Tables;
Procedure CreateIAT_TextFields;
procedure ChcT6Click(Sender: TObject);
Procedure CreateIAT_SubTables;
procedure ChcPSub1to3Click(Sender: TObject);
procedure ChcPSub4to6Click(Sender: TObject);
procedure ChcPSub7to10Click(Sender: TObject);
procedure ChcSubU1to3Click(Sender: TObject);
procedure ChcUSub4to6Click(Sender: TObject);
procedure ChcUSub7to10Click(Sender: TObject);
procedure DBImageP1DblClick(Sender: TObject);
procedure DBImageP2DblClick(Sender: TObject);
procedure DBImageP3DblClick(Sender: TObject);
procedure DBImageP4DblClick(Sender: TObject);
procedure DBImageP5DblClick(Sender: TObject);
procedure DBImageP6DblClick(Sender: TObject);
procedure DBImageP7DblClick(Sender: TObject);
procedure DBImageP8DblClick(Sender: TObject);
procedure DBImageP9DblClick(Sender: TObject);
procedure DBImageP10DblClick(Sender: TObject);
procedure DBImageM1DblClick(Sender: TObject);
procedure DBImageM2DblClick(Sender: TObject);
procedure DBImageM3DblClick(Sender: TObject);
procedure DBImageM4DblClick(Sender: TObject);
procedure DBImageM5DblClick(Sender: TObject);
procedure DBImageM6DblClick(Sender: TObject);
procedure DBImageM7DblClick(Sender: TObject);
procedure DBImageM8DblClick(Sender: TObject);
procedure DBImageM9DblClick(Sender: TObject);
procedure DBImageM10DblClick(Sender: TObject);
procedure ChcPSub1to3BClick(Sender: TObject);
procedure ChcPSub4to6BClick(Sender: TObject);
procedure ChcPSub7to10BClick(Sender: TObject);
procedure ChcSubU1to3BClick(Sender: TObject);
procedure ChcUSub4to6BClick(Sender: TObject);
procedure ChcUSub7to10BClick(Sender: TObject);
procedure ChcS1to3Click(Sender: TObject);
procedure ChcS4to6Click(Sender: TObject);
procedure ChcS7to10Click(Sender: TObject);
procedure ChcS1to3BClick(Sender: TObject);
procedure ChcS4to6BClick(Sender: TObject);
procedure ChcS7to10BClick(Sender: TObject);
procedure DBImageP12DblClick(Sender: TObject);
procedure DBImageP13DblClick(Sender: TObject);
```

```
    procedure DBImageP14DblClick(Sender: TObject);
    procedure DBImageP15DblClick(Sender: TObject);
    procedure DBImageP16DblClick(Sender: TObject);
    procedure DBImageP17DblClick(Sender: TObject);
    procedure DBImageP18DblClick(Sender: TObject);
    procedure DBImageP19DblClick(Sender: TObject);
    procedure DBImageP20DblClick(Sender: TObject);
    procedure DBImageM11DblClick(Sender: TObject);
    procedure DBImageM12DblClick(Sender: TObject);
    procedure DBImageM13DblClick(Sender: TObject);
    procedure DBImageM14DblClick(Sender: TObject);
    procedure DBImageM15DblClick(Sender: TObject);
    procedure DBImageM15Click(Sender: TObject);
    procedure DBImageM13Click(Sender: TObject);
    procedure DBImageM16DblClick(Sender: TObject);
    procedure DBImageM17DblClick(Sender: TObject);
    procedure DBImageM18DblClick(Sender: TObject);
    procedure DBImageM19DblClick(Sender: TObject);
    procedure DBImageM20DblClick(Sender: TObject);
    procedure DBImageP11DblClick(Sender: TObject);
    procedure DBImageS1DblClick(Sender: TObject);
    procedure DBImageS2DblClick(Sender: TObject);
    procedure DBImageS3DblClick(Sender: TObject);
    procedure DBImageS4DblClick(Sender: TObject);
    procedure DBImageS5DblClick(Sender: TObject);
    procedure DBImageS6DblClick(Sender: TObject);
    procedure DBImageS7DblClick(Sender: TObject);
    procedure DBImageS8DblClick(Sender: TObject);
    procedure DBImageS9DblClick(Sender: TObject);
    procedure DBImageS10DblClick(Sender: TObject);
    procedure DBImageS1BDblClick(Sender: TObject);
    procedure DBImageS2BDblClick(Sender: TObject);
    procedure DBImageS3BDblClick(Sender: TObject);
    procedure DBImageS4BDblClick(Sender: TObject);
    procedure DBImageS5BDblClick(Sender: TObject);
    procedure DBImageS6BDblClick(Sender: TObject);
    procedure DBImageS7BDblClick(Sender: TObject);
    procedure DBImageS8BDblClick(Sender: TObject);
    procedure DBImageS9BDblClick(Sender: TObject);
    procedure DBImageS10BDblClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form3: TForm3;
  Var CString : WideString;
  Var SList : TStringlist;

implementation

{$R *.dfm}
```

```
Procedure CreateTableParticipants;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`participants` (';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`ID` varchar(9) DEFAULT NULL,';
  SString:=SString+'`Gender` varchar(1) DEFAULT NULL,';
  SString:=SString+'`Age` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Visual Rating` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Colour Blind` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Date` datetime DEFAULT NULL,';
  SString:=SString+'`V1` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V2` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V3` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V4` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V5` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V6` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V7` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V8` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`V9` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`First Language` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form3.ADOCommand1.CommandText:=SString;
  Form3.ADOCommand1.Execute;
End;

Procedure TForm3.Connecting;
Begin
  LEDConnect.Appearance.Fill.Color:=RGB(255,102,0);
End;

procedure TForm3.ADOConnection3AfterConnect(Sender: TObject);
begin
  Connecting;
  BtnConnectBlobs.Enabled:=True;
end;

procedure TForm3.ADOConnection3AfterDisconnect(Sender: TObject);
begin
  ChcT1.Checked:=False;
  ChcT2.Checked:=False;
  ChcT3.Checked:=False;
  ChcT4.Checked:=False;
  ChcT5.Checked:=False;
  ChcT6.Checked:=False;
  ChcPSub1to3.Checked:=False;
  ChcPSub4to6.Checked:=False;
  ChcPSub7to10.Checked:=False;
  ChcSubU1to3.Checked:=False;
  ChcUSub4to6.Checked:=False;
  ChcUSub7to10.Checked:=False;
```

```
  ChcPSub1to3B.Checked:=False;
  ChcPSub4to6B.Checked:=False;
  ChcPSub7to10B.Checked:=False;
  ChcSubU1to3B.Checked:=False;
  ChcUSub4to6B.Checked:=False;
  ChcUSub7to10B.Checked:=False;
  ChcS1to3.Checked:=False;
  ChcS4to6.Checked:=False;
  ChcS7to10.Checked:=False;
  ChcS1to3B.Checked:=False;
  ChcS4to6B.Checked:=False;
  ChcS7to10B.Checked:=False;
  LEDConnect.Appearance.Fill.Color:=ClRed;
end;

procedure TForm3.BitBtn1Click(Sender: TObject);
begin
 ADOTable1.Active:=false;
 ADoTable1.Close;
 ADOCommand1.Free;
 ADOCommand2.Free;
 ADOConnection1.Close;
 ADOConnection2.Close;
end;

procedure TForm3.BtnConnectBlobsClick(Sender: TObject);
begin
  DBNavigator2.Enabled:=True;
  Table2.TableName:='settings';
  Table2.Open;
  Table2.Active:=True;
  if ChcAuto.Checked then
  Form3.ChcT1Click(Sender);
  BtnConnectBlobs.Enabled:=False;
  Button7.Enabled:=True;
end;

Procedure TForm3.CreateIAT_Tables;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`settings` (';
  SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`M1` mediumblob,';
  SString:=SString+'`M2` mediumblob,';
  SString:=SString+'`M3` mediumblob,';
  SString:=SString+'`M4` mediumblob,';
  SString:=SString+'`M5` mediumblob,';
  SString:=SString+'`M6` mediumblob,';
  SString:=SString+'`M7` mediumblob,';
  SString:=SString+'`MA` mediumblob,';
  SString:=SString+'`IN1` longblob,';
  SString:=SString+'`IN2` longblob,';
  SString:=SString+'PRIMARY KEY (`IDX`))';
```

```
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`in3to5` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`IN3` longblob,';
SString:=SString+'`IN4` longblob,';
SString:=SString+'`IN5` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`in6to10` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`IN6` longblob,';
SString:=SString+'`IN7` longblob,';
SString:=SString+'`IN8` longblob,';
SString:=SString+'`IN9` longblob,';
SString:=SString+'`IN10` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`out1to3` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`out1` longblob,';
SString:=SString+'`out2` longblob,';
SString:=SString+'`out3` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`out4to6` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`out4` longblob,';
SString:=SString+'`out5` longblob,';
SString:=SString+'`out6` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`out7to10` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`out7` longblob,';
SString:=SString+'`out8` longblob,';
SString:=SString+'`out9` longblob,';
SString:=SString+'`out10` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
```

```
  Form3.ADOCommand3.CommandText:=SString;
  Form3.ADOCommand3.Execute;
  SString:='';
End;

Procedure TForm3.CreateIAT_TextFields;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`textfields` (';
  SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`times` varchar(30) DEFAULT NULL,';
  SString:=SString+'`parameters` varchar(1240) DEFAULT NULL,';
  SString:=SString+'`PWE` varchar(100) DEFAULT NULL,';
  SString:=SString+'`UWE` varchar(100) DEFAULT NULL,';
  SString:=SString+'`IGItems` varchar(100) DEFAULT NULL,';
  SString:=SString+'`OGItems` varchar(100) DEFAULT NULL,';
  SString:=SString+'`SubPItems` varchar(100) DEFAULT NULL,';
  SString:=SString+'`SubUItems` varchar(100) DEFAULT NULL,';
  SString:=SString+'`SubPOItems` varchar(100) DEFAULT NULL,';
  SString:=SString+'`SubUOItems` varchar(100) DEFAULT NULL,';
  SString:=SString+'`concepts` varchar(200) DEFAULT NULL,';
  SString:=SString+'`verdicts` varchar(500) DEFAULT NULL,';
  SString:=SString+'`questions` varchar(600) DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`IDX`))';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
  Form3.ADOCommand3.CommandText:=SString;
  Form3.ADOCommand3.Execute;
  SString:='';
End;

Procedure TForm3.CreateIAT_SubTables;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subp1to3` (';
  SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`subp1` longblob,';
  SString:=SString+'`subp2` longblob,';
  SString:=SString+'`subp3` longblob,';
  SString:=SString+'PRIMARY KEY (`IDX`))';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
  Form3.ADOCommand3.CommandText:=SString;
  Form3.ADOCommand3.Execute;
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subp4to6` (';
  SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`subp4` longblob,';
  SString:=SString+'`subp5` longblob,';
  SString:=SString+'`subp6` longblob,';
  SString:=SString+'PRIMARY KEY (`IDX`))';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
  Form3.ADOCommand3.CommandText:=SString;
  Form3.ADOCommand3.Execute;
```

```
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subp7to10` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subp7` longblob,';
SString:=SString+'`subp8` longblob,';
SString:=SString+'`subp9` longblob,';
SString:=SString+'`subp10` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subu1to3` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subu1` longblob,';
SString:=SString+'`subu2` longblob,';
SString:=SString+'`subu3` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subu4to6` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subu4` longblob,';
SString:=SString+'`subu5` longblob,';
SString:=SString+'`subu6` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subu7to10` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subu7` longblob,';
SString:=SString+'`subu8` longblob,';
SString:=SString+'`subu9` longblob,';
SString:=SString+'`subu10` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;

SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subp11to13` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subp11` longblob,';
SString:=SString+'`subp12` longblob,';
SString:=SString+'`subp13` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
```

```
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subp14to16` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subp14` longblob,';
SString:=SString+'`subp15` longblob,';
SString:=SString+'`subp16` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subp17to20` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subp17` longblob,';
SString:=SString+'`subp18` longblob,';
SString:=SString+'`subp19` longblob,';
SString:=SString+'`subp20` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subu11to13` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subu11` longblob,';
SString:=SString+'`subu12` longblob,';
SString:=SString+'`subu13` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subu14to16` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subu14` longblob,';
SString:=SString+'`subu15` longblob,';
SString:=SString+'`subu16` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`subu17to20` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subu17` longblob,';
SString:=SString+'`subu18` longblob,';
SString:=SString+'`subu19` longblob,';
SString:=SString+'`subu20` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`S1to3` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
```

```
SString:=SString+'`subp11` longblob,';
SString:=SString+'`subp12` longblob,';
SString:=SString+'`subp13` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`S4to6` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subp14` longblob,';
SString:=SString+'`subp15` longblob,';
SString:=SString+'`subp16` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`S7to10` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subp17` longblob,';
SString:=SString+'`subp18` longblob,';
SString:=SString+'`subp19` longblob,';
SString:=SString+'`subp20` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`S1to3B` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subu11` longblob,';
SString:=SString+'`subu12` longblob,';
SString:=SString+'`subu13` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`S4to6B` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subu14` longblob,';
SString:=SString+'`subu15` longblob,';
SString:=SString+'`subu16` longblob,';
SString:=SString+'PRIMARY KEY (`IDX`))';
SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
Form3.ADOCommand3.CommandText:=SString;
Form3.ADOCommand3.Execute;
SString:='';
SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`S7to10B` (';
SString:=SString+'`IDX` int(10) unsigned NOT NULL AUTO_INCREMENT,';
SString:=SString+'`subu17` longblob,';
SString:=SString+'`subu18` longblob,';
SString:=SString+'`subu19` longblob,';
```

```
  SString:=SString+'`subu20` longblob,';
  SString:=SString+'PRIMARY KEY (`IDX`))';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1 CHECKSUM=1;';
  Form3.ADOCommand3.CommandText:=SString;
  Form3.ADOCommand3.Execute;
  SString:='';
End;

Procedure CreateParticipants;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `iat2`.`participants` (';
  SString:=SString+'`Number` int(10) unsigned NOT NULL AUTO_INCREMENT,';
  SString:=SString+'`ID` varchar(9) NOT NULL,';
  SString:=SString+'`Gender` varchar(1) DEFAULT NULL,';
  SString:=SString+'`Age` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`QResponse` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Date` datetime NOT NULL,';
  SString:=SString+'`Block` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Colour` varchar(1) DEFAULT NULL,';
  SString:=SString+'`SubTimer1` varchar(4) DEFAULT NULL,';
  SString:=SString+'`SubTimer2` varchar(4) DEFAULT NULL,';
  SString:=SString+'`Counter` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Diff` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`Item` varchar(15) DEFAULT NULL,';
  SString:=SString+'`SubItem` varchar(15) DEFAULT NULL,';
  SString:=SString+'`Flag` TINYINT(1) unsigned DEFAULT NULL,';
  SString:=SString+'`BlockOrder` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'`First Language` int(10) unsigned DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Number`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form3.ADOCommand3.CommandText:=SString;
  Form3.ADOCommand3.Execute;
End;

procedure TForm3.BtnCreateIATClick(Sender: TObject);
begin
 CreateIAT_Tables;
 CreateIAT_TextFields;
 CreateIAT_SubTables;
 CreateParticipants;
end;

procedure TForm3.BtnOpenFileClick(Sender: TObject);
Var Idx : Integer;
    EventDate : ShortString;
begin
  Idx:=0;
  SList:=TStringList.Create;
  if OpenDialog1.Execute then
    SList.LoadFromFile(OpenDialog1.FileName);
  if SList.Count>0 then
  begin
```

```
ADOTable1.TableName:='settings';
ADOTable1.Open;
ADOTable1.Active:=True;
  With ADOTable1 Do
    Begin
      Edit;
      Fields.FieldByName('Timer1').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('Timer2').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('Timer3').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('Mask').Value:=SList[Idx];
      Inc(Idx);
      Fields.FieldByName('BGC1').Value:=SList[Idx];
      Inc(Idx);
      Fields.FieldByName('BGC2').Value:=SList[Idx];
      Inc(Idx);
      Fields.FieldByName('SizerInterval').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('Overide').Value:=SList[Idx];
      Inc(Idx);
      Fields.FieldByName('Random').Value:=SList[Idx];
      Inc(Idx);
      Fields.FieldByName('PracticeLight').Value:=SList[Idx];
      Inc(Idx);
      Fields.FieldByName('MainLight').Value:=SList[Idx];
      Inc(Idx);
      Fields.FieldByName('Calibrate').Value:=SList[Idx];
      Inc(Idx);
      Fields.FieldByName('Title').Value:=SList[Idx];
      Inc(Idx);;
      Fields.FieldByName('Graphics').Value:=SList[Idx];
      Inc(Idx);
      Fields.FieldByName('Panel').Value:=SList[Idx];
      Inc(Idx);
      Fields.FieldByName('CIterations').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('TurnaroundM').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('MaxRuns').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('P1Time').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('P2Time').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('P3Time').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('P4Time').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('P5Time').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
      Fields.FieldByName('FromBGR').Value:=StrToInt(SList[Idx]);
      Inc(Idx);
```

```
Fields.FieldByName('FromBGG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FromBGB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FromFGR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FromFGG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FromFGB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Equiliminance').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToBGR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToBGG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToBGB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToFGR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToFGG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('ToFGB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('SizerFactor').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Sizer').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('TargetWidth').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('TargetHeight').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('TargetLeft').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('TargetTop').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C1B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1C2B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2C1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2C1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2C1B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
```

```
Fields.FieldByName('S2C2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2C2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2C2B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C1B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3C2B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4C1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4C1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4C1B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4C2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4C2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4C2B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C1B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5C2B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S1T').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S2T').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S3T').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S4T') .Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('S5T').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('UVM') .Value:=StrToInt(SList[Idx]);
Inc(Idx);
```

```
Fields.FieldByName('LVM') .Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('AVM').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FormR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FormG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FormB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FieldR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FieldG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('FieldB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('StimulusR').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('StimulusG').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('StimulusB').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('User Calibration').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Cal1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Cal1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Cal1B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Cal2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Cal2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Cal2B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Prescribed Solution').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('Bypass Calibration').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('PC1R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('PC1G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('PC1B').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('PC2R').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('PC2G').Value:=StrToInt(SList[Idx]);
Inc(Idx);
Fields.FieldByName('PC2B').Value:=StrToInt(SList[Idx]);
EventDate:=DateTimeToStr(Now);
Fields.FieldByName('Date').AsDateTime:=StrToDateTime(EventDate);
UpDateRecord;
```

```
        Post;
      End;
      SList.Free;
  end;
end;

procedure TForm3.BtnSaveFileClick(Sender: TObject);
Var EventDate : ShortString;
begin
  SList:=TStringlist.Create;
  Slist.Add(E1.Text);
  Slist.Add(E2.Text);
  Slist.Add(E3.Text);
  Slist.Add(E4.Text);
  Slist.Add(E5.Text);
  Slist.Add(E6.Text);
  Slist.Add(E7.Text);
  Slist.Add(E8.Text);
  Slist.Add(E9.Text);
  Slist.Add(E10.Text);
  Slist.Add(E11.Text);
  Slist.Add(E12.Text);
  Slist.Add(E13.Text);
  Slist.Add(E14.Text);
  Slist.Add(E15.Text);
  Slist.Add(E16.Text);
  Slist.Add(E17.Text);
  Slist.Add(E18.Text);
  Slist.Add(E19.Text);
  Slist.Add(E20.Text);
  Slist.Add(E21.Text);
  Slist.Add(E22.Text);
  Slist.Add(E23.Text);
  Slist.Add(E24.Text);
  Slist.Add(E25.Text);
  Slist.Add(E26.Text);
  Slist.Add(E27.Text);
  Slist.Add(E28.Text);
  Slist.Add(E29.Text);
  Slist.Add(E30.Text);
  Slist.Add(E31.Text);
  Slist.Add(E32.Text);
  Slist.Add(E33.Text);
  Slist.Add(E34.Text);
  Slist.Add(E35.Text);
  Slist.Add(E36.Text);
  Slist.Add(E37.Text);
     if Form3.ChkSizer.Checked then
       Slist.Add('1')
         Else
          Slist.Add('0');
  Slist.Add(E38.Text);
  Slist.Add(E39.Text);
  Slist.Add(E40.Text);
```

```
Slist.Add(E41.Text);
Slist.Add(S1C1R.Text);
Slist.Add(S1C1G.Text);
Slist.Add(S1C1B.Text);
Slist.Add(S1C2R.Text);
Slist.Add(S1C2G.Text);
Slist.Add(S1C2B.Text);
Slist.Add(S2C1R.Text);
Slist.Add(S2C1G.Text);
Slist.Add(S2C1B.Text);
Slist.Add(S2C2R.Text);
Slist.Add(S2C2G.Text);
Slist.Add(S2C2B.Text);
Slist.Add(S3C1R.Text);
Slist.Add(S3C1G.Text);
Slist.Add(S3C1B.Text);
Slist.Add(S3C2R.Text);
Slist.Add(S3C2G.Text);
Slist.Add(S3C2B.Text);
Slist.Add(S4C1R.Text);
Slist.Add(S4C1G.Text);
Slist.Add(S4C1B.Text);
Slist.Add(S4C2R.Text);
Slist.Add(S4C2G.Text);
Slist.Add(S4C2B.Text);
Slist.Add(S5C1R.Text);
Slist.Add(S5C1G.Text);
Slist.Add(S5C1B.Text);
Slist.Add(S5C2R.Text);
Slist.Add(S5C2G.Text);
Slist.Add(S5C2B.Text);
Slist.Add(S1T.Text);
Slist.Add(S2T.Text);
Slist.Add(S3T.Text);
Slist.Add(S4T.Text);
Slist.Add(S5T.Text);
Slist.Add(UVM.Text);
Slist.Add(LVM.Text);
Slist.Add(AVM.Text);
Slist.Add(FormR.Text);
Slist.Add(FormG.Text);
Slist.Add(FormB.Text);
Slist.Add(FieldR.Text);
Slist.Add(FieldG.Text);
Slist.Add(FieldB.Text);
Slist.Add(StimulusR.Text);
Slist.Add(StimulusG.Text);
Slist.Add(StimulusB.Text);
  if Form3.ChkUserCal.Checked then
    Slist.Add('1')
      Else
       Slist.Add('0');
Slist.Add(Cal1R.Text);
Slist.Add(Cal1G.Text);
```

```
  Slist.Add(Cal1B.Text);
  Slist.Add(Cal2R.Text);
  Slist.Add(Cal2G.Text);
  Slist.Add(Cal2B.Text);
     if Form3.ChkPrescribed.Checked then
       Slist.Add('1')
         Else
          Slist.Add('0');
     if Form3.ChkByPass.Checked then
       Slist.Add('1')
         Else
          Slist.Add('0');
   Slist.Add(PC1R.Text);
   Slist.Add(PC1G.Text);
   Slist.Add(PC1B.Text);
   Slist.Add(PC2R.Text);
   Slist.Add(PC2G.Text);
   Slist.Add(PC2B.Text);
   EventDate:=DateTimeToStr(Now);
   Slist.Add(EventDate);
   if SaveDialog1.Execute then
     SList.SaveToFile(SaveDialog1.FileName);
   SList.Free;
end;

procedure TForm3.Button1Click(Sender: TObject);
begin
  CreateTableParticipants;
end;

Procedure CreateTableCalibration;
Var SString : WideString;
Begin
  SString:='';
  SString:=SString+'CREATE TABLE IF NOT EXISTS `uctexp1`.`settings` (';
  SString:=SString+'`Timer1` int(10) unsigned NOT NULL,';
  SString:=SString+'`Timer2` int(10) unsigned NOT NULL,';
  SString:=SString+'`Timer3` int(10) unsigned NOT NULL,';
  SString:=SString+'`Mask` varchar(10) DEFAULT NULL,';
  SString:=SString+'`BGC1` varchar(10) DEFAULT NULL,';
  SString:=SString+'`BGC2` varchar(10) DEFAULT NULL,';
  SString:=SString+'`SizerInterval` int(10) unsigned NOT NULL,';
  SString:=SString+'`Overide` varchar(10) DEFAULT NULL,';
  SString:=SString+'`Random` varchar(10) DEFAULT NULL,';
  SString:=SString+'`PracticeLight` varchar(10) DEFAULT NULL,';
  SString:=SString+'`MainLight` varchar(10) DEFAULT NULL,';
  SString:=SString+'`Calibrate` varchar(11) DEFAULT NULL,';
  SString:=SString+'`Title` varchar(30) DEFAULT NULL,';
  SString:=SString+'`Graphics` varchar(10) DEFAULT NULL,';
  SString:=SString+'`Panel` varchar(10) DEFAULT NULL,';
  SString:=SString+'`CIterations` int(10) unsigned NOT NULL,';
  SString:=SString+'`TurnaroundM` int(10) unsigned NOT NULL,';
  SString:=SString+'`MaxRuns` int(10) unsigned NOT NULL,';
  SString:=SString+'`P1Time` int(10) unsigned NOT NULL,';
```

```
SString:=SString+'`P2Time` int(10) unsigned NOT NULL,';
SString:=SString+'`P3Time` int(10) unsigned NOT NULL,';
SString:=SString+'`P4Time` int(10) unsigned NOT NULL,';
SString:=SString+'`P5Time` int(10) unsigned NOT NULL,';
SString:=SString+'`FromBGR` int(10) unsigned NOT NULL,';
SString:=SString+'`FromBGG` int(10) unsigned NOT NULL,';
SString:=SString+'`FromBGB` int(10) unsigned NOT NULL,';
SString:=SString+'`ToBGR` int(10) unsigned NOT NULL,';
SString:=SString+'`ToBGG` int(10) unsigned NOT NULL,';
SString:=SString+'`ToBGB` int(10) unsigned NOT NULL,';
SString:=SString+'`FromFGR` int(10) unsigned NOT NULL,';
SString:=SString+'`FromFGG` int(10) unsigned NOT NULL,';
SString:=SString+'`FromFGB` int(10) unsigned NOT NULL,';
SString:=SString+'`ToFGR` int(10) unsigned NOT NULL,';
SString:=SString+'`ToFGG` int(10) unsigned NOT NULL,';
SString:=SString+'`ToFGB` int(10) unsigned NOT NULL,';
SString:=SString+'`Equiliminance` int(10) unsigned NOT NULL,';
SString:=SString+'`Sizer` TINYINT(1) unsigned NOT NULL,';
SString:=SString+'`SizerFactor` int(10) unsigned NOT NULL,';
SString:=SString+'`TargetWidth` int(10) unsigned NOT NULL,';
SString:=SString+'`TargetHeight` int(10) unsigned NOT NULL,';
SString:=SString+'`TargetLeft` int(10) unsigned NOT NULL,';
SString:=SString+'`TargetTop` int(10) unsigned NOT NULL,';
SString:=SString+'`S1C1R` int(10) unsigned NOT NULL,';
SString:=SString+'`S1C1G` int(10) unsigned NOT NULL,';
SString:=SString+'`S1C1B` int(10) unsigned NOT NULL,';
SString:=SString+'`S1C2R` int(10) unsigned NOT NULL,';
SString:=SString+'`S1C2G` int(10) unsigned NOT NULL,';
SString:=SString+'`S1C2B` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C1R` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C1G` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C1B` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C2R` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C2G` int(10) unsigned NOT NULL,';
SString:=SString+'`S2C2B` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C1R` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C1G` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C1B` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C2R` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C2G` int(10) unsigned NOT NULL,';
SString:=SString+'`S3C2B` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C1R` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C1G` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C1B` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C2R` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C2G` int(10) unsigned NOT NULL,';
SString:=SString+'`S4C2B` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C1R` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C1G` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C1B` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C2R` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C2G` int(10) unsigned NOT NULL,';
SString:=SString+'`S5C2B` int(10) unsigned NOT NULL,';
SString:=SString+'`S1T` int(10) unsigned NOT NULL,';
```

```
  SString:=SString+'`S2T` int(10) unsigned NOT NULL,';
  SString:=SString+'`S3T` int(10) unsigned NOT NULL,';
  SString:=SString+'`S4T` int(10) unsigned NOT NULL,';
  SString:=SString+'`S5T` int(10) unsigned NOT NULL,';
  SString:=SString+'`UVM` int(10) unsigned NOT NULL,';
  SString:=SString+'`LVM` int(10) unsigned NOT NULL,';
  SString:=SString+'`AVM` int(10) unsigned NOT NULL,';
  SString:=SString+'`FormR` int(10) unsigned NOT NULL,';
  SString:=SString+'`FormG` int(10) unsigned NOT NULL,';
  SString:=SString+'`FormB` int(10) unsigned NOT NULL,';
  SString:=SString+'`FieldR` int(10) unsigned NOT NULL,';
  SString:=SString+'`FieldG` int(10) unsigned NOT NULL,';
  SString:=SString+'`FieldB` int(10) unsigned NOT NULL,';
  SString:=SString+'`StimulusR` int(10) unsigned NOT NULL,';
  SString:=SString+'`StimulusG` int(10) unsigned NOT NULL,';
  SString:=SString+'`StimulusB` int(10) unsigned NOT NULL,';
  SString:=SString+'`User Calibration` TINYINT(1) unsigned NOT NULL,';
  SString:=SString+'`Cal1R` int(10) unsigned NOT NULL,';
  SString:=SString+'`Cal1G` int(10) unsigned NOT NULL,';
  SString:=SString+'`Cal1B` int(10) unsigned NOT NULL,';
  SString:=SString+'`Cal2R` int(10) unsigned NOT NULL,';
  SString:=SString+'`Cal2G` int(10) unsigned NOT NULL,';
  SString:=SString+'`Cal2B` int(10) unsigned NOT NULL,';
  SString:=SString+'`Prescribed Solution` TINYINT(1) unsigned NOT NULL,';
  SString:=SString+'`Bypass Calibration` TINYINT(1) unsigned NOT NULL,';
  SString:=SString+'`PC1R` int(10) unsigned NOT NULL,';
  SString:=SString+'`PC1G` int(10) unsigned NOT NULL,';
  SString:=SString+'`PC1B` int(10) unsigned NOT NULL,';
  SString:=SString+'`PC2R` int(10) unsigned NOT NULL,';
  SString:=SString+'`PC2G` int(10) unsigned NOT NULL,';
  SString:=SString+'`PC2B` int(10) unsigned NOT NULL,';
  SString:=SString+'`Date` datetime DEFAULT NULL,';
  SString:=SString+'PRIMARY KEY (`Timer1`) USING BTREE) ';
  SString:=SString+'ENGINE=InnoDB DEFAULT CHARSET=latin1  CHECKSUM=1;';
  Form3.ADOCommand1.CommandText:=SString;
  Form3.ADOCommand1.Execute;
End;

procedure TForm3.Button2Click(Sender: TObject);
begin
  CreateTableCalibration;
end;

procedure TForm3.Button3Click(Sender: TObject);
Var EventDate : ShortString;
begin
  ADOTable1.TableName:='settings';
  ADOTable1.Open;
  ADOTable1.Active:=True;
    With ADOTable1 Do
    Begin
      Append;
      Fields.FieldByName('Timer1').Value:=StrToInt(E1.Text);
      Fields.FieldByName('Timer2').Value:=StrToInt(E2.Text);
```

```
Fields.FieldByName('Timer3').Value:=StrToInt(E3.Text);
Fields.FieldByName('Mask').Value:=E4.Text;
Fields.FieldByName('BGC1').Value:=E5.Text;
Fields.FieldByName('BGC2').Value:=E6.Text;
Fields.FieldByName('SizerInterval').Value:=StrToInt(E7.Text);
Fields.FieldByName('Overide').Value:=E8.Text;
Fields.FieldByName('Random').Value:=E9.Text;
Fields.FieldByName('PracticeLight').Value:=E10.Text;
Fields.FieldByName('MainLight').Value:=E11.Text;
Fields.FieldByName('Calibrate').Value:=E12.Text;
Fields.FieldByName('Title').Value:=E13.Text;
Fields.FieldByName('Graphics').Value:=E14.Text;
Fields.FieldByName('Panel').Value:=E15.Text;
Fields.FieldByName('CIterations').Value:=StrToInt(E16.Text);
Fields.FieldByName('TurnaroundM').Value:=StrToInt(E17.Text);
Fields.FieldByName('MaxRuns').Value:=StrToInt(E18.Text);
Fields.FieldByName('P1Time').Value:=StrToInt(E19.Text);
Fields.FieldByName('P2Time').Value:=StrToInt(E20.Text);
Fields.FieldByName('P3Time').Value:=StrToInt(E21.Text);
Fields.FieldByName('P4Time').Value:=StrToInt(E22.Text);
Fields.FieldByName('P5Time').Value:=StrToInt(E23.Text);
Fields.FieldByName('FromBGR').Value:=StrToInt(E24.Text);
Fields.FieldByName('FromBGG').Value:=StrToInt(E25.Text);
Fields.FieldByName('FromBGB').Value:=StrToInt(E26.Text);
Fields.FieldByName('FromFGR').Value:=StrToInt(E27.Text);
Fields.FieldByName('FromFGG').Value:=StrToInt(E28.Text);
Fields.FieldByName('FromFGB').Value:=StrToInt(E29.Text);
Fields.FieldByName('Equiliminance').Value:=StrToInt(E30.Text);
Fields.FieldByName('ToBGR').Value:=StrToInt(E31.Text);
Fields.FieldByName('ToBGG').Value:=StrToInt(E32.Text);
Fields.FieldByName('ToBGB').Value:=StrToInt(E33.Text);
Fields.FieldByName('ToFGR').Value:=StrToInt(E34.Text);
Fields.FieldByName('ToFGG').Value:=StrToInt(E35.Text);
Fields.FieldByName('ToFGB').Value:=StrToInt(E36.Text);
Fields.FieldByName('SizerFactor').Value:=StrToInt(E37.Text);
if Form3.ChkSizer.Checked then
  Fields.FieldByName('Sizer').Value:=1
    Else
     Fields.FieldByName('Sizer').Value:=0;
Fields.FieldByName('TargetWidth').Value:=StrToInt(E38.Text);
Fields.FieldByName('TargetHeight').Value:=StrToInt(E39.Text);
Fields.FieldByName('TargetLeft').Value:=StrToInt(E40.Text);
Fields.FieldByName('TargetTop').Value:=StrToInt(E41.Text);
Fields.FieldByName('S1C1R').Value:=StrToInt(S1C1R.Text);
Fields.FieldByName('S1C1G').Value:=StrToInt(S1C1G.Text);
Fields.FieldByName('S1C1B').Value:=StrToInt(S1C1B.Text);
Fields.FieldByName('S1C2R').Value:=StrToInt(S1C2R.Text);
Fields.FieldByName('S1C2G').Value:=StrToInt(S1C2G.Text);
Fields.FieldByName('S1C2B').Value:=StrToInt(S1C2B.Text);
Fields.FieldByName('S2C1R').Value:=StrToInt(S2C1R.Text);
Fields.FieldByName('S2C1G').Value:=StrToInt(S2C1G.Text);
Fields.FieldByName('S2C1B').Value:=StrToInt(S2C1B.Text);
Fields.FieldByName('S2C2R').Value:=StrToInt(S2C2R.Text);
Fields.FieldByName('S2C2G').Value:=StrToInt(S2C2G.Text);
```

```
Fields.FieldByName('S2C2B').Value:=StrToInt(S2C2B.Text);
Fields.FieldByName('S3C1R').Value:=StrToInt(S3C1R.Text);
Fields.FieldByName('S3C1G').Value:=StrToInt(S3C1G.Text);
Fields.FieldByName('S3C1B').Value:=StrToInt(S3C1B.Text);
Fields.FieldByName('S3C2R').Value:=StrToInt(S3C2R.Text);
Fields.FieldByName('S3C2G').Value:=StrToInt(S3C2G.Text);
Fields.FieldByName('S3C2B').Value:=StrToInt(S3C2B.Text);
Fields.FieldByName('S4C1R').Value:=StrToInt(S4C1R.Text);
Fields.FieldByName('S4C1G').Value:=StrToInt(S4C1G.Text);
Fields.FieldByName('S4C1B').Value:=StrToInt(S4C1B.Text);
Fields.FieldByName('S4C2R').Value:=StrToInt(S4C2R.Text);
Fields.FieldByName('S4C2G').Value:=StrToInt(S4C2G.Text);
Fields.FieldByName('S4C2B').Value:=StrToInt(S4C2B.Text);
Fields.FieldByName('S5C1R').Value:=StrToInt(S5C1R.Text);
Fields.FieldByName('S5C1G').Value:=StrToInt(S5C1G.Text);
Fields.FieldByName('S5C1B').Value:=StrToInt(S5C1B.Text);
Fields.FieldByName('S5C2R').Value:=StrToInt(S5C2R.Text);
Fields.FieldByName('S5C2G').Value:=StrToInt(S5C2G.Text);
Fields.FieldByName('S5C2B').Value:=StrToInt(S5C2B.Text);
Fields.FieldByName('S1T').Value:=StrToInt(S1T.Text);
Fields.FieldByName('S2T').Value:=StrToInt(S2T.Text);
Fields.FieldByName('S3T').Value:=StrToInt(S3T.Text);
Fields.FieldByName('S4T') .Value:=StrToInt(S4T.Text);
Fields.FieldByName('S5T').Value:=StrToInt(S5T.Text);
Fields.FieldByName('UVM') .Value:=StrToInt(UVM.Text);
Fields.FieldByName('LVM') .Value:=StrToInt(LVM.Text);
Fields.FieldByName('AVM').Value:=StrToInt(AVM.Text);
Fields.FieldByName('FormR').Value:=StrToInt(FormR.Text);
Fields.FieldByName('FormG').Value:=StrToInt(FormG.Text);
Fields.FieldByName('FormB').Value:=StrToInt(FormB.Text);
Fields.FieldByName('FieldR').Value:=StrToInt(FieldR.Text);
Fields.FieldByName('FieldG').Value:=StrToInt(FieldG.Text);
Fields.FieldByName('FieldB').Value:=StrToInt(FieldB.Text);
Fields.FieldByName('StimulusR').Value:=StrToInt(StimulusR.Text);
Fields.FieldByName('StimulusG').Value:=StrToInt(StimulusG.Text);
Fields.FieldByName('StimulusB').Value:=StrToInt(StimulusB.Text);
if Form3.ChkUserCal.Checked then
  Fields.FieldByName('User Calibration').Value:=1
    Else
     Fields.FieldByName('User Calibration').Value:=0;
Fields.FieldByName('Cal1R').Value:=StrToInt(Cal1R.Text);
Fields.FieldByName('Cal1G').Value:=StrToInt(Cal1G.Text);
Fields.FieldByName('Cal1B').Value:=StrToInt(Cal1B.Text);
Fields.FieldByName('Cal2R').Value:=StrToInt(Cal2R.Text);
Fields.FieldByName('Cal2G').Value:=StrToInt(Cal2G.Text);
Fields.FieldByName('Cal2B').Value:=StrToInt(Cal2B.Text);
if Form3.ChkPrescribed.Checked then
  Fields.FieldByName('Prescribed Solution').Value:=1
    Else
     Fields.FieldByName('Prescribed Solution').Value:=0;
if Form3.ChkByPass.Checked then
  Fields.FieldByName('Bypass Calibration').Value:=1
    Else
     Fields.FieldByName('Bypass Calibration').Value:=0;
```

```
    Fields.FieldByName('PC1R').Value:=StrToInt(PC1R.Text);
    Fields.FieldByName('PC1G').Value:=StrToInt(PC1G.Text);
    Fields.FieldByName('PC1B').Value:=StrToInt(PC1B.Text);
    Fields.FieldByName('PC2R').Value:=StrToInt(PC2R.Text);
    Fields.FieldByName('PC2G').Value:=StrToInt(PC2G.Text);
    Fields.FieldByName('PC2B').Value:=StrToInt(PC2B.Text);
    EventDate:=DateTimeToStr(Now);
    Fields.FieldByName('Date').AsDateTime:=StrToDateTime(EventDate);
    UpDateRecord;
    Post;
   End;
end;

Procedure TForm3.UpdateFields;
Begin
  if ADOTable1.Active then
   With ADOTable1 Do
    Begin
      S1C1R.Text:=Fields.FieldByName('S1C1R').Value;
      S1C1G.Text:=Fields.FieldByName('S1C1G').Value;
      S1C1B.Text:=Fields.FieldByName('S1C1B').Value;
      S1C2R.Text:=Fields.FieldByName('S1C2R').Value;
      S1C2G.Text:=Fields.FieldByName('S1C2G').Value;
      S1C2B.Text:=Fields.FieldByName('S1C2B').Value;
      S2C1R.Text:=Fields.FieldByName('S2C1R').Value;
      S2C1G.Text:=Fields.FieldByName('S2C1G').Value;
      S2C1B.Text:=Fields.FieldByName('S2C1B').Value;
      S2C2R.Text:=Fields.FieldByName('S2C2R').Value;
      S2C2G.Text:=Fields.FieldByName('S2C2G').Value;
      S2C2B.Text:=Fields.FieldByName('S2C2B').Value;
      S3C1R.Text:=Fields.FieldByName('S3C1R').Value;
      S3C1G.Text:=Fields.FieldByName('S3C1G').Value;
      S3C1B.Text:=Fields.FieldByName('S3C1B').Value;
      S3C2R.Text:=Fields.FieldByName('S3C2R').Value;
      S3C2G.Text:=Fields.FieldByName('S3C2G').Value;
      S3C2B.Text:=Fields.FieldByName('S3C2B').Value;
      S4C1R.Text:=Fields.FieldByName('S4C1R').Value;
      S4C1G.Text:=Fields.FieldByName('S4C1G').Value;
      S4C1B.Text:=Fields.FieldByName('S4C1B').Value;
      S4C2R.Text:=Fields.FieldByName('S4C2R').Value;
      S4C2G.Text:=Fields.FieldByName('S4C2G').Value;
      S4C2B.Text:=Fields.FieldByName('S4C2B').Value;
      S5C1R.Text:=Fields.FieldByName('S5C1R').Value;
      S5C1G.Text:=Fields.FieldByName('S5C1G').Value;
      S5C1B.Text:=Fields.FieldByName('S5C1B').Value;
      S5C2R.Text:=Fields.FieldByName('S5C2R').Value;
      S5C2G.Text:=Fields.FieldByName('S5C2G').Value;
      S5C2B.Text:=Fields.FieldByName('S5C2B').Value;
      FormR.Text:=Fields.FieldByName('FormR').Value;
      FormG.Text:=Fields.FieldByName('FormG').Value;
      FormB.Text:=Fields.FieldByName('FormB').Value;
      FieldR.Text:=Fields.FieldByName('FieldR').Value;
      FieldG.Text:=Fields.FieldByName('FieldG').Value;
      FieldB.Text:=Fields.FieldByName('FieldB').Value;
```

```
      StimulusR.Text:=Fields.FieldByName('StimulusR').Value;
      StimulusG.Text:=Fields.FieldByName('StimulusG').Value;
      StimulusB.Text:=Fields.FieldByName('StimulusB').Value;
        if (Fields.FieldByName('Sizer').Value='1') then
           Form3.ChkSizer.Checked:=true
            Else
             Form3.ChkSizer.Checked:=False;
        if (Fields.FieldByName('User Calibration').Value='1') then
           Form3.ChkUserCal.Checked:=true
            Else
             Form3.ChkUserCal.Checked:=False;
      Cal1R.Text:=Fields.FieldByName('Cal1R').Value;
      Cal1G.Text:=Fields.FieldByName('Cal1G').Value;
      Cal1B.Text:=Fields.FieldByName('Cal1B').Value;
      Cal2R.Text:=Fields.FieldByName('Cal2R').Value;
      Cal2G.Text:=Fields.FieldByName('Cal2G').Value;
      Cal2B.Text:=Fields.FieldByName('Cal2B').Value;
      if (Fields.FieldByName('Prescribed Solution').Value='1') then
          Form3.ChkPrescribed.Checked:=true
           Else
            Form3.ChkPrescribed.Checked:=False;
      if (Fields.FieldByName('Bypass Calibration').Value='1') then
          Form3.ChkBypass.Checked:=True
           Else
             Form3.ChkBypass.Checked:=False;
      PC1R.Text:=Fields.FieldByName('PC1R').Value;
      PC1G.Text:=Fields.FieldByName('PC1G').Value;
      PC1B.Text:=Fields.FieldByName('PC1B').Value;
      PC2R.Text:=Fields.FieldByName('PC2R').Value;
      PC2G.Text:=Fields.FieldByName('PC2G').Value;
      PC2B.Text:=Fields.FieldByName('PC2B').Value;
     End;
//     UpdateColour;
End;

procedure TForm3.Button4Click(Sender: TObject);
begin
  ADOTable1.TableName:='settings';
  ADOTable1.Open;
  ADOTable1.Active:=True;
end;

procedure TForm3.Button5Click(Sender: TObject);
Var EventDate : ShortString;
begin
   If ADOTable1.Active Then
   Begin
     With ADOTable1 Do
      Begin
        Edit;
        Fields.FieldByName('Timer1').Value:=StrToInt(E1.Text);
        Fields.FieldByName('Timer2').Value:=StrToInt(E2.Text);
        Fields.FieldByName('Timer3').Value:=StrToInt(E3.Text);
        Fields.FieldByName('Mask').Value:=E4.Text;
```

```
Fields.FieldByName('BGC1').Value:=E5.Text;
Fields.FieldByName('BGC2').Value:=E6.Text;
Fields.FieldByName('SizerInterval').Value:=StrToInt(E7.Text);
Fields.FieldByName('Overide').Value:=E8.Text;
Fields.FieldByName('Random').Value:=E9.Text;
Fields.FieldByName('PracticeLight').Value:=E10.Text;
Fields.FieldByName('MainLight').Value:=E11.Text;
Fields.FieldByName('Calibrate').Value:=E12.Text;
Fields.FieldByName('Title').Value:=E13.Text;
Fields.FieldByName('Graphics').Value:=E14.Text;
Fields.FieldByName('Panel').Value:=E15.Text;
Fields.FieldByName('CIterations').Value:=StrToInt(E16.Text);
Fields.FieldByName('TurnaroundM').Value:=StrToInt(E17.Text);
Fields.FieldByName('MaxRuns').Value:=StrToInt(E18.Text);
Fields.FieldByName('P1Time').Value:=StrToInt(E19.Text);
Fields.FieldByName('P2Time').Value:=StrToInt(E20.Text);
Fields.FieldByName('P3Time').Value:=StrToInt(E21.Text);
Fields.FieldByName('P4Time').Value:=StrToInt(E22.Text);
Fields.FieldByName('P5Time').Value:=StrToInt(E23.Text);
Fields.FieldByName('FromBGR').Value:=StrToInt(E24.Text);
Fields.FieldByName('FromBGG').Value:=StrToInt(E25.Text);
Fields.FieldByName('FromBGB').Value:=StrToInt(E26.Text);
Fields.FieldByName('FromFGR').Value:=StrToInt(E27.Text);
Fields.FieldByName('FromFGG').Value:=StrToInt(E28.Text);
Fields.FieldByName('FromFGB').Value:=StrToInt(E29.Text);
Fields.FieldByName('Equiliminance').Value:=StrToInt(E30.Text);
Fields.FieldByName('ToBGR').Value:=StrToInt(E31.Text);
Fields.FieldByName('ToBGG').Value:=StrToInt(E32.Text);
Fields.FieldByName('ToBGB').Value:=StrToInt(E33.Text);
Fields.FieldByName('ToFGR').Value:=StrToInt(E34.Text);
Fields.FieldByName('ToFGG').Value:=StrToInt(E35.Text);
Fields.FieldByName('ToFGB').Value:=StrToInt(E36.Text);
Fields.FieldByName('SizerFactor').Value:=StrToInt(E37.Text);
if Form3.ChkSizer.Checked then
  Fields.FieldByName('Sizer').Value:=1
    Else
        Fields.FieldByName('Sizer').Value:=0;
Fields.FieldByName('TargetWidth').Value:=StrToInt(E38.Text);
Fields.FieldByName('TargetHeight').Value:=StrToInt(E39.Text);
Fields.FieldByName('TargetLeft').Value:=StrToInt(E40.Text);
Fields.FieldByName('TargetTop').Value:=StrToInt(E41.Text);
Fields.FieldByName('S1C1R').Value:=StrToInt(S1C1R.Text);
Fields.FieldByName('S1C1G').Value:=StrToInt(S1C1G.Text);
Fields.FieldByName('S1C1B').Value:=StrToInt(S1C1B.Text);
Fields.FieldByName('S1C2R').Value:=StrToInt(S1C2R.Text);
Fields.FieldByName('S1C2G').Value:=StrToInt(S1C2G.Text);
Fields.FieldByName('S1C2B').Value:=StrToInt(S1C2B.Text);
Fields.FieldByName('S2C1R').Value:=StrToInt(S2C1R.Text);
Fields.FieldByName('S2C1G').Value:=StrToInt(S2C1G.Text);
Fields.FieldByName('S2C1B').Value:=StrToInt(S2C1B.Text);
Fields.FieldByName('S2C2R').Value:=StrToInt(S2C2R.Text);
Fields.FieldByName('S2C2G').Value:=StrToInt(S2C2G.Text);
Fields.FieldByName('S2C2B').Value:=StrToInt(S2C2B.Text);
Fields.FieldByName('S3C1R').Value:=StrToInt(S3C1R.Text);
```

```
Fields.FieldByName('S3C1G').Value:=StrToInt(S3C1G.Text);
Fields.FieldByName('S3C1B').Value:=StrToInt(S3C1B.Text);
Fields.FieldByName('S3C2R').Value:=StrToInt(S3C2R.Text);
Fields.FieldByName('S3C2G').Value:=StrToInt(S3C2G.Text);
Fields.FieldByName('S3C2B').Value:=StrToInt(S3C2B.Text);
Fields.FieldByName('S4C1R').Value:=StrToInt(S4C1R.Text);
Fields.FieldByName('S4C1G').Value:=StrToInt(S4C1G.Text);
Fields.FieldByName('S4C1B').Value:=StrToInt(S4C1B.Text);
Fields.FieldByName('S4C2R').Value:=StrToInt(S4C2R.Text);
Fields.FieldByName('S4C2G').Value:=StrToInt(S4C2G.Text);
Fields.FieldByName('S4C2B').Value:=StrToInt(S4C2B.Text);
Fields.FieldByName('S5C1R').Value:=StrToInt(S5C1R.Text);
Fields.FieldByName('S5C1G').Value:=StrToInt(S5C1G.Text);
Fields.FieldByName('S5C1B').Value:=StrToInt(S5C1B.Text);
Fields.FieldByName('S5C2R').Value:=StrToInt(S5C2R.Text);
Fields.FieldByName('S5C2G').Value:=StrToInt(S5C2G.Text);
Fields.FieldByName('S5C2B').Value:=StrToInt(S5C2B.Text);
Fields.FieldByName('S1T').Value:=StrToInt(S1T.Text);
Fields.FieldByName('S2T').Value:=StrToInt(S2T.Text);
Fields.FieldByName('S3T').Value:=StrToInt(S3T.Text);
Fields.FieldByName('S4T') .Value:=StrToInt(S4T.Text);
Fields.FieldByName('S5T').Value:=StrToInt(S5T.Text);
Fields.FieldByName('UVM') .Value:=StrToInt(UVM.Text);
Fields.FieldByName('LVM') .Value:=StrToInt(LVM.Text);
Fields.FieldByName('AVM').Value:=StrToInt(AVM.Text);
Fields.FieldByName('FormR').Value:=StrToInt(FormR.Text);
Fields.FieldByName('FormG').Value:=StrToInt(FormG.Text);
Fields.FieldByName('FormB').Value:=StrToInt(FormB.Text);
Fields.FieldByName('FieldR').Value:=StrToInt(FieldR.Text);
Fields.FieldByName('FieldG').Value:=StrToInt(FieldG.Text);
Fields.FieldByName('FieldB').Value:=StrToInt(FieldB.Text);
Fields.FieldByName('StimulusR').Value:=StrToInt(StimulusR.Text);
Fields.FieldByName('StimulusG').Value:=StrToInt(StimulusG.Text);
Fields.FieldByName('StimulusB').Value:=StrToInt(StimulusB.Text);
if Form3.ChkUserCal.Checked then
  Fields.FieldByName('User Calibration').Value:=1
    Else
      Fields.FieldByName('User Calibration').Value:=0;
Fields.FieldByName('Cal1R').Value:=StrToInt(Cal1R.Text);
Fields.FieldByName('Cal1G').Value:=StrToInt(Cal1G.Text);
Fields.FieldByName('Cal1B').Value:=StrToInt(Cal1B.Text);
Fields.FieldByName('Cal2R').Value:=StrToInt(Cal2R.Text);
Fields.FieldByName('Cal2G').Value:=StrToInt(Cal2G.Text);
Fields.FieldByName('Cal2B').Value:=StrToInt(Cal2B.Text);
if Form3.ChkPrescribed.Checked then
  Fields.FieldByName('Prescribed Solution').Value:=1
    Else
      Fields.FieldByName('Prescribed Solution').Value:=0;
if Form3.ChkByPass.Checked then
   Fields.FieldByName('Bypass Calibration').Value:=1
    Else
      Fields.FieldByName('Bypass Calibration').Value:=0;
Fields.FieldByName('PC1R').Value:=StrToInt(PC1R.Text);
Fields.FieldByName('PC1G').Value:=StrToInt(PC1G.Text);
```

```
        Fields.FieldByName('PC1B').Value:=StrToInt(PC1B.Text);
        Fields.FieldByName('PC2R').Value:=StrToInt(PC2R.Text);
        Fields.FieldByName('PC2G').Value:=StrToInt(PC2G.Text);
        Fields.FieldByName('PC2B').Value:=StrToInt(PC2B.Text);
        EventDate:=DateTimeToStr(Now);
        Fields.FieldByName('Date').AsDateTime:=StrToDateTime(EventDate);
        UpDateRecord;
      Post;
    End;
  End;
end;

procedure TForm3.Button6Click(Sender: TObject);
begin
  ADOTable1.Active:=False;
  ADOTable1.Close;
end;

procedure TForm3.Button7Click(Sender: TObject);
begin
//  ChcT1.Checked:=False;
  ChcT1.Caption:='Not Connected';
  ChcT2.Caption:='Not Connected';
  ChcT3.Caption:='Not Connected';
  ChcT4.Caption:='Not Connected';
  ChcT5.Caption:='Not Connected';
  ChcT6.Caption:='Not Connected';
  ChcPSub1to3.Caption:='Not Connected';
  ChcPSub4to6.Caption:='Not Connected';
  ChcPSub7to10.Caption:='Not Connected';
  ChcSubU1to3.Caption:='Not Connected';
  ChcUSub4to6.Caption:='Not Connected';
  ChcUSub7to10.Caption:='Not Connected';
  ChcUSub7to10.Caption:='Not Connected';
  ChcPSub1to3B.Caption:='Not Connected';
  ChcPSub4to6B.Caption:='Not Connected';
  ChcPSub7to10B.Caption:='Not Connected';
  ChcSubU1to3B.Caption:='Not Connected';
  ChcUSub4to6B.Caption:='Not Connected';
  ChcUSub7to10B.Caption:='Not Connected';
  ChcS1to3.Caption:='Not Connected';
  ChcS4to6.Caption:='Not Connected';
  ChcS7to10.Caption:='Not Connected';
  ChcS1to3B.Caption:='Not Connected';
  ChcS4to6B.Caption:='Not Connected';
  ChcS7to10B.Caption:='Not Connected';
  Table2.Close;
  Table2.Active:=False;
  TableIn3to5.Close;
  TableIn3to5.Active:=False;
  Table6to10.Close;
  Table6to10.Active:=False;
  TableO1to3.Close;
  TableO1to3.Active:=False;
```

```
  TableO4to6.Close;
  TableO4to6.Active:=False;
  Table7to10.Close;
  Table7to10.Active:=False;
  TableTextFields.Close;
  TableTextFields.Active:=False;
  TablePSub1to3.Close;
  TablePSub1to3.Active:=False;
  TablePSub4to6.Close;
  TablePSub4to6.Active:=False;
  TablePSub7to10.Close;
  TablePSub7to10.Active:=false;
  TableUSub1to3.Close;
  TableUSub1to3.Active:=false;
  TableUSub4to6.Close;
  TableUSub4to6.Active:=False;
  TableUSub7to10.Close;
  TableUSub7to10.Active:=False;
  subp11to13.Close;
  subp11to13.Active:=False;
  subp14to16.Close;
  subp14to16.Active:=False;
  subp17to20.Close;
  subp17to20.Active:=False;
  subu10to13.Close;
  subu10to13.Active:=False;
  subu14to16.Close;
  subu14to16.active:=false;
  subu17to20.Close;
  subu17to20.Active:=False;
  ProgressBar1.Position:=0;
  Progressbar2.Position:=0;
  Progressbar3.Position:=0;
  Progressbar4.Position:=0;
  ADOConnection3.Close;
  ADOConnection3.Connected:=False;
end;

procedure TForm3.M3UpdateClick(Sender: TObject);
Var blob:  TBlobField;
begin
 if OpenDialog2.Execute then
    If Table2.Active Then
      Begin
        Table2.Edit;
        blob := Table2.fields.FieldByName('M3') as TBlobField;
        Blob.LoadFromFile(OpenDialog2.FileName);
        Table2.UpdateRecord;
      End;
end;

procedure TForm3.M4UpdateClick(Sender: TObject);
Var blob:  TBlobField;
begin
```

```
 if OpenDialog2.Execute then
    If Table2.Active Then
      Begin
        Table2.Edit;
        blob := Table2.fields.FieldByName('M4') as TBlobField;
        Blob.LoadFromFile(OpenDialog2.FileName);
        Table2.UpdateRecord;
      End;
end;

procedure TForm3.M5UpdateClick(Sender: TObject);
Var blob:  TBlobField;
begin
 if OpenDialog2.Execute then
    If Table2.Active Then
      Begin
        Table2.Edit;
        blob := Table2.fields.FieldByName('M5') as TBlobField;
        Blob.LoadFromFile(OpenDialog2.FileName);
        Table2.UpdateRecord;
      End;
end;
procedure TForm3.M6UpdateClick(Sender: TObject);
Var blob:  TBlobField;
begin
 if OpenDialog2.Execute then
    If Table2.Active Then
      Begin
        Table2.Edit;
        blob := Table2.fields.FieldByName('M6') as TBlobField;
        Blob.LoadFromFile(OpenDialog2.FileName);
        Table2.UpdateRecord;
      End;
end;

procedure TForm3.M7UpdateClick(Sender: TObject);
Var blob:  TBlobField;
begin
 if OpenDialog2.Execute then
    If Table2.Active Then
      Begin
        Table2.Edit;
        blob := Table2.fields.FieldByName('M7') as TBlobField;
        Blob.LoadFromFile(OpenDialog2.FileName);
        Table2.UpdateRecord;
      End;
end;

procedure TForm3.MAUpdateClick(Sender: TObject);
Var blob:  TBlobField;
begin
 if OpenDialog2.Execute then
    If Table2.Active Then
      Begin
```

```
      Table2.Edit;
      blob := Table2.fields.FieldByName('MA') as TBlobField;
      Blob.LoadFromFile(OpenDialog2.FileName);
      Table2.UpdateRecord;
     End;
end;

procedure TForm3.M2UpdateClick(Sender: TObject);
Var blob:  TBlobField;
begin
 if OpenDialog2.Execute then
   If Table2.Active Then
     Begin
      Table2.Edit;
      blob := Table2.fields.FieldByName('M2') as TBlobField;
      Blob.LoadFromFile(OpenDialog2.FileName);
      Table2.UpdateRecord;
     End;
end;

procedure TForm3.ChcPSub1to3BClick(Sender: TObject);
begin
  if ADOConnection3.Connected then
  Begin
    ProgressBar3.StepBy(3);
    subp11to13.Open;
    subp11to13.Active:=true;
    ChcPSub1to3B.Checked:=True;
    ChcPSub1to3B.Caption:='Connected';
    if ChcAutoSub2.Checked then
      Form3.ChcPSub4to6BClick(Sender);
  End;
end;

procedure TForm3.ChcPSub1to3Click(Sender: TObject);
begin
  if ADOConnection3.Connected then
  Begin
    ProgressBar2.StepBy(3);
    TablePSub1to3.Open;
    TablePSub1to3.Active:=true;
    ChcPSub1to3.Checked:=True;
    ChcPSub1to3.Caption:='Connected';
    if ChcAutoSub.Checked then
      Form3.ChcPSub4to6Click(Sender);
  End;
end;

procedure TForm3.ChcPSub4to6BClick(Sender: TObject);
begin
  if ADOConnection3.Connected then
  Begin
    ProgressBar3.StepBy(3);
    subp14to16.Open;
```

```
    subp14to16.Active:=true;
    ChcPSub4to6B.Checked:=True;
    ChcPSub4to6B.Caption:='Connected';
    if ChcAutoSub2.Checked then
      Form3.ChcPSub7to10BClick(Sender);
  End;
end;

procedure TForm3.ChcPSub4to6Click(Sender: TObject);
Begin
 if ADOConnection3.Connected then
  Begin
    ProgressBar2.StepBy(3);
    TablePSub4to6.Open;
    TablePSub4to6.Active:=true;
    ChcPSub4to6.Checked:=True;
    ChcPSub4to6.Caption:='Connected';
  End;
    if ChcAutoSub.Checked then
      Form3.ChcPSub7to10Click(Sender);
End;

procedure TForm3.ChcPSub7to10BClick(Sender: TObject);
begin
  if ADOConnection3.Connected then
  Begin
    ProgressBar3.StepBy(4);
    subp17to20.Open;
    subp17to20.Active:=true;
    ChcPSub7to10B.Checked:=True;
    ChcPSub7to10B.Caption:='Connected';
    if ChcAutoSub2.Checked then
      Form3.ChcSubU1to3BClick(Sender);
  End;
end;

procedure TForm3.ChcPSub7to10Click(Sender: TObject);
Begin
 if ADOConnection3.Connected then
  Begin
    ProgressBar2.StepBy(4);
    TablePSub7to10.Open;
    TablePSub7to10.Active:=true;
    ChcPSub7to10.Checked:=True;
    ChcPSub7to10.Caption:='Connected';
  End;
    if ChcAutoSub.Checked then
      Form3.ChcSubU1to3Click(Sender);
End;

procedure TForm3.ChcS1to3BClick(Sender: TObject);
begin
  if ADOConnection3.Connected then
  Begin
```

```
    ProgressBar4.StepBy(3);
    S1TO3B.Open;
    S1TO3B.Active:=true;
    ChcS1to3B.Checked:=True;
    ChcS1to3B.Caption:='Connected';
    if ChcAutoS.Checked then
      Form3.ChcS4to6BClick(Sender);
  End;
end;

procedure TForm3.ChcS1to3Click(Sender: TObject);
begin
  if (ADOConnection3.Connected) then
  begin
    ProgressBar4.StepBy(3);
    S1to3.Open;
    S1to3.Active:=True;
    ChcS1to3.Checked:=True;
    ChcS1to3.Caption:='Connected';
    if ChcAutoS.Checked then
      Form3.ChcS4to6Click(Sender);
  end;
end;

procedure TForm3.ChcS4to6BClick(Sender: TObject);
begin
  if ADOConnection3.Connected then
  Begin
    ProgressBar4.StepBy(3);
    S4TO6B.Open;
    S4TO6B.Active:=true;
    ChcS4to6B.Checked:=True;
    ChcS4to6B.Caption:='Connected';
    if ChcAutoS.Checked then
      Form3.ChcS7to10BClick(Sender);
  End;
end;

procedure TForm3.ChcS4to6Click(Sender: TObject);
begin
  if (ADOConnection3.Connected) Then
  begin
    ProgressBar4.StepBy(3);
    S4To6.Open;
    S4To6.Active:=True;
    ChcS4to6.Checked:=True;
    ChcS4to6.Caption:='Connected';
    if ChcAutoS.Checked then
      Form3.ChcS7to10Click(Sender);
  end;
end;

procedure TForm3.ChcS7to10BClick(Sender: TObject);
begin
```

```
  if ADOConnection3.Connected then
  Begin
    ProgressBar4.StepBy(4);
    S7TO10B.Open;
    S7TO10B.Active:=true;
    ChcS7to10B.Checked:=True;
    ChcS7to10B.Caption:='Connected';
    LEDConnect.Appearance.Fill.Color:=ClLime;
  End;
end;

procedure TForm3.ChcS7to10Click(Sender: TObject);
begin
  if ADOConnection3.Connected then
  Begin
    ProgressBar4.StepBy(4);
    S7To10.Open;
    S7To10.Active:=true;
    ChcS7to10.Checked:=True;
    ChcS7to10.Caption:='Connected';
    if ChcAutoS.Checked then
      Form3.ChcS1to3BClick(Sender);
  End;
end;

procedure TForm3.ChcSubU1to3BClick(Sender: TObject);
begin
  if ADOConnection3.Connected then
  Begin
    ProgressBar3.StepBy(3);
    subu10to13.Open;
    subu10to13.Active:=true;
    ChcSubU1to3B.Checked:=True;
    ChcSubU1to3B.Caption:='Connected';
    if ChcAutoSub2.Checked then
      Form3.ChcUSub4to6BClick(Sender);
  End;
end;

procedure TForm3.ChcSubU1to3Click(Sender: TObject);
Begin
 if ADOConnection3.Connected then
  Begin
    ProgressBar2.StepBy(3);
    TableUSub1to3.Open;
    TableUSub1to3.Active:=true;
    ChcSubU1to3.Checked:=True;
    ChcSubU1to3.Caption:='Connected';
  End;
    if ChcAutoSub.Checked then
      Form3.ChcUSub4to6Click(Sender);
End;

procedure TForm3.ChcT1Click(Sender: TObject);
```

```
begin
  if ADOConnection3.Connected then
  Begin
    ProgressBar1.StepBy(5);
    TableIn3to5.Open;
    TableIn3to5.Active:=true;
    ChcT1.Checked:=True;
    ChcT1.Caption:='Connected';
    if (ADOConnection3.Connected) and (ChcAuto.Checked) then
      Form3.ChcT2Click(Sender);
  End;
end;

procedure TForm3.ChcT2Click(Sender: TObject);
begin
  if (ADOConnection3.Connected) then
  Begin
    ProgressBar1.StepBy(5);
    Table6to10.Open;
    Table6to10.Active:=true;
    ChcT2.Checked:=True;
    ChcT2.Caption:='Connected';
    IF (ADOConnection3.Connected) and (ChcAuto.Checked) Then
      Form3.ChcT3Click(Sender);
  End;
end;

procedure TForm3.ChcT3Click(Sender: TObject);
begin
  if (ADOConnection3.Connected) then
  Begin
    ProgressBar1.StepBy(3);
    TableO1to3.Open;
    TableO1to3.Active:=true;
    ChcT3.Checked:=True;
    ChcT3.Caption:='Connected';
    IF (ADOConnection3.Connected) and (ChcAuto.Checked) Then
      Form3.ChcT4Click(Sender);
  End;
end;

procedure TForm3.ChcT4Click(Sender: TObject);
begin
  if (ADOConnection3.Connected) then
  Begin
    ProgressBar1.StepBy(3);
    TableO4to6.Open;
    TableO4to6.Active:=true;
    ChcT4.Checked:=True;
    ChcT4.Caption:='Connected';
    IF (ADOConnection3.Connected) and (ChcAuto.Checked) Then
      Form3.ChcT5Click(Sender);
  End;
end;
```

```
procedure TForm3.ChcT5Click(Sender: TObject);
begin
  if (ADOConnection3.Connected) then
  Begin
    ProgressBar1.StepBy(4);
    Table7to10.Open;
    Table7to10.Active:=true;
    ChcT5.Checked:=True;
    ChcT5.Caption:='Connected';
    IF (ADOConnection3.Connected) and (ChcAuto.Checked) Then
      Form3.ChcT6Click(Sender);
  End;
end;

procedure TForm3.ChcT6Click(Sender: TObject);
begin
  if (ADOConnection3.Connected) then
  begin
    TableTextFields.Open;
    TableTextFields.Active:=True;
    ChcT6.Checked:=True;
    ChcT6.Caption:='Connected';
    Form3.Refresh;
    if ChcAutoSub.Checked then
      Begin
       Form3.Refresh;
       Form3.ChcPSub1to3Click(Sender);
      End
        Else
          LEDConnect.Appearance.Fill.Color:=ClLime;
  end;
end;

procedure TForm3.ChcUSub4to6BClick(Sender: TObject);
begin
  if ADOConnection3.Connected then
  Begin
    ProgressBar3.StepBy(3);
    subu14to16.Open;
    subu14to16.Active:=true;
    ChcUSub4to6B.Checked:=True;
    ChcUSub4to6B.Caption:='Connected';
    if ChcAutoSub2.Checked then
      Form3.ChcUSub7to10BClick(Sender);
  End;
end;

procedure TForm3.ChcUSub4to6Click(Sender: TObject);
Begin
 if ADOConnection3.Connected then
  Begin
    ProgressBar2.StepBy(3);
    TableUSub4to6.Open;
```

```
    TableUSub4to6.Active:=true;
    ChcUSub4to6.Checked:=True;
    ChcUSub4to6.Caption:='Connected';    //TableUSub7to10
  End;
    if ChcAutoSub.Checked then
      Form3.ChcUSub7to10Click(Sender);
End;

procedure TForm3.ChcUSub7to10BClick(Sender: TObject);
begin
  if ADOConnection3.Connected then
  Begin
    ProgressBar3.StepBy(4);
    subu17to20.Open;
    subu17to20.Active:=true;
    ChcUSub7to10B.Checked:=True;
    ChcUSub7to10B.Caption:='Connected';
    if ChcAutoS.Checked then
      Begin
//       Form3.AdvSmoothTabPager1.ActivePage:=AdvSmoothTabPage11;
        Form3.ChcS1to3Click(Sender);
      End
        Else
          LEDConnect.Appearance.Fill.Color:=ClLime;
  End;
end;

procedure TForm3.ChcUSub7to10Click(Sender: TObject);
Begin
 if ADOConnection3.Connected then
  Begin
    ProgressBar2.StepBy(4);
    TableUSub7to10.Open;
    TableUSub7to10.Active:=true;
    ChcUSub7to10.Checked:=True;
    ChcUSub7to10.Caption:='Connected';      //TableUSub7to10
     if ChcAutoSub2.Checked then
     Begin
//      Form3.AdvSmoothTabPager1.ActivePage:=AdvSmoothTabPage10;
      Form3.ChcPSub1to3BClick(Sender);
     End
        Else
          LEDConnect.Appearance.Fill.Color:=ClLime;
  End;
End;

procedure TForm3.ChkBypassClick(Sender: TObject);
begin
   If ADOTable1.Active Then
    Begin
      With ADOTable1 Do
        Begin
          Edit;
          if ChkByPass.Checked then
```

```
          Fields.FieldByName('Bypass Calibration').Value:=1
            Else
              Fields.FieldByName('Bypass Calibration').Value:=0;
        End;
    End;
end;

procedure TForm3.ChkPrescribedClick(Sender: TObject);
begin
   If ADOTable1.Active Then
    Begin
      With ADOTable1 Do
        Begin
          Edit;
          if ChkPrescribed.Checked then
            Fields.FieldByName('Prescribed Solution').Value:=1
              Else
                Fields.FieldByName('Prescribed Solution').Value:=0;
        End;
    End;
end;

procedure TForm3.ChkSizerClick(Sender: TObject);
begin
   If ADOTable1.Active Then
    Begin
      With ADOTable1 Do
        Begin
          Edit;
          if ChkSizer.Checked then
            Fields.FieldByName('Sizer').Value:=1
              Else
                Fields.FieldByName('Sizer').Value:=0;
        End;
    End;
end;

procedure TForm3.ChkUserCalClick(Sender: TObject);
begin
   If ADOTable1.Active Then
    Begin
      With ADOTable1 Do
        Begin
          Edit;
          if ChkUserCal.Checked then
            Fields.FieldByName('User Calibration').Value:=1
              Else
                Fields.FieldByName('User Calibration').Value:=0;
        End;
    End;
end;

procedure TForm3.ComboBox1Change(Sender: TObject);
begin
```

```
 If ADOConnection1.Connected Then
 Begin
   ADOConnection1.Close;
   ADOConnection1.Connected:=False;
 End;
  if ComboBox1.Text='Local Vista'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE2;Initial Catalog=uctexp1';
  if ComboBox1.Text='Mansfield'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE1;Initial Catalog=uctexp1';
  if ComboBox1.Text='Work PC'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE3;Initial Catalog=uctexp1';
  if ComboBox1.Text='UCT Server'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE5;Initial Catalog=uctexp1';
  if ComboBox1.Text='Windows 7 PC'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=DOUGAS-HP;Initial Catalog=uctexp1';
  ADOCOmmand1.ConnectionString:=CString;
  Try
    ADOConnection1.ConnectionString:=CString;
    ADOConnection1.Connected:=True;
  Except
    ShowMessage('Cannot connect');
  End;
  Button1.Enabled:=ADOConnection1.Connected;
  Button4.Enabled:=ADOConnection1.Connected;
  Button6.Enabled:=ADOConnection1.Connected;
  BtnOpenFile.Enabled:=ADOConnection1.Connected;
  BtnSaveFile.Enabled:=ADOConnection1.Connected;
end;

procedure TForm3.ComboBox2Change(Sender: TObject);
begin
 If ADOConnection1.Connected Then
 Begin
   ADOConnection1.Close;
   ADOConnection1.Connected:=False;
 End;
  if ComboBox1.Text='Local Vista'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE2;Initial Catalog=uctexp1';
  if ComboBox2.Text='Mansfield'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE1;Initial Catalog=uctexp1';
  if ComboBox2.Text='Work PC'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE3;Initial Catalog=uctexp1';
  if ComboBox2.Text='UCT Server'
   then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=UCTMSQLE5;Initial Catalog=uctexp1';
  if ComboBox2.Text='Windows 7 PC'
```

```
  then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=DOUGAS-HP;Initial Catalog=uctexp1';
 ADOCOmmand1.ConnectionString:=CString;
 Try
   ADOConnection1.ConnectionString:=CString;
   ADOConnection1.Connected:=True;
 Except
   ShowMessage('Cannot connect');
 End;
 Button2.Enabled:=ADOConnection1.Connected;
 Button3.Enabled:=ADOConnection1.Connected;
 Button5.Enabled:=ADOConnection1.Connected;
 Button4.Enabled:=ADOConnection1.Connected;
 Button6.Enabled:=ADOConnection1.Connected;
end;

procedure TForm3.ComboBox3Change(Sender: TObject);
begin
 If ADOConnection3.Connected Then
 Begin
   ADOConnection3.Close;
   ADOConnection3.Connected:=False;
 End;
 if ComboBox3.Text='Local Vista'
  then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT DELL;Mode=ReadWrite;Initial Catalog=iat2';
 if ComboBox3.Text='Mansfield'
  then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT Mansfield;Mode=ReadWrite;Initial Catalog=iat2';
 if ComboBox3.Text='Work PC'
  then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT2 WORK HP;Mode=ReadWrite;Initial Catalog=iat2';
 if ComboBox3.Text='UCT Server'
  then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT CHOMSKY;Mode=ReadWrite;Initial Catalog=iat2';
 if ComboBox3.Text='Windows 7 PC'
  then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT W7HP;Mode=ReadWrite;Initial Catalog=iat2';
 if ComboBox3.Text='W7 Siemens'
  then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT DOUGLAS-PC;Mode=ReadWrite;Initial Catalog=iat2';
 if ComboBox3.Text='MANSFIELD-PC'
  then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT Mansfield-PC;Mode=ReadWrite;Initial Catalog=iat2';
 if ComboBox3.Text='Mansfield64'
  then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT2 Mansfield64;Mode=ReadWrite;Initial Catalog=iat2';
 if ComboBox3.Text='DouglasW8'
  then CString:='Provider=MSDASQL.1;Password=douglasjohn;Persist Security Info=True;User ID=Douglas;Data
Source=IAT2 DouglasW8B;Mode=ReadWrite;Initial Catalog=iat2';
 Try
   ADOConnection3.ConnectionString:=CString;
   ADOConnection3.Connected:=True;
 Except
```

```
    ShowMessage('Cannot connect');
  End;
end;

procedure TForm3.DataSource1DataChange(Sender: TObject; Field: TField);
begin
  if DBSizer.Field.Value='0' then
    ChkSizer.Checked:=false;
  if DBSizer.Field.Value='1' then
    ChkSizer.Checked:=true;
  if DBCal.Field.Value='0' then
    ChkUserCal.Checked:=false;
  if DBCal.Field.Value='1' then
    ChkUserCal.Checked:=True;
  if DBPrescribed.Field.Value='0' then
    ChkPrescribed.Checked:=false;
  if DBPrescribed.Field.Value='1' then
    ChkPrescribed.Checked:=True;
  if DbBypass.Field.Value='1' then
    ChkByPass.Checked:=True;
  if DbBypass.Field.Value='0' then
    ChkByPass.Checked:=False;
    UpdateColour;
//    UpdateFields;
end;


Procedure TForm3.DBGrid1MouseLeave(Sender: TObject);
begin
//  UpdateFields;
end;

procedure TForm3.DBImageM10DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableUSub7to10.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourceUSub7to10;
         TableUSub7to10.Edit;
         blob := TableUSub7to10.fields.FieldByName('subu10') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableUSub7to10.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageM11DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu10to13.Active Then
```

```
      Begin
        DbNavigator2.DataSource:=DataSourcesubu10to13;
        subu10to13.Edit;
        blob := subu10to13.fields.FieldByName('subu11') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        subu10to13.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageM12DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu10to13.Active Then
      Begin
        DbNavigator2.DataSource:=DataSourcesubu10to13;
        subu10to13.Edit;
        blob := subu10to13.fields.FieldByName('subu12') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        subu10to13.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageM13Click(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu10to13.Active Then
      Begin
        DbNavigator5.DataSource:=DataSourcesubu10to13;
        subu10to13.Edit;
        blob := subu10to13.fields.FieldByName('subu13') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        subu10to13.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageM13DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu10to13.Active Then
      Begin
        DbNavigator2.DataSource:=DataSourcesubu10to13;
        subu10to13.Edit;
        blob := subu10to13.fields.FieldByName('subu13') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        subu10to13.UpdateRecord;
```

```
      End;
   End;
end;

procedure TForm3.DBImageM14DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu14to16.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubu14to16;
         subu14to16.Edit;
         blob := subu14to16.fields.FieldByName('subu14') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subu14to16.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBImageM15Click(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu14to16.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourcesubu14to16;
         subu14to16.Edit;
         blob := subu14to16.fields.FieldByName('subu15') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subu14to16.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBImageM15DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu14to16.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubu14to16;
         subu14to16.Edit;
         blob := subu14to16.fields.FieldByName('subu15') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subu14to16.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBImageM16DblClick(Sender: TObject);
Var blob: TBlobField;
```

```
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu14to16.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubu14to16;
         subu14to16.Edit;
         blob := subu14to16.fields.FieldByName('subu16') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subu14to16.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageM17DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu17to20.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubu17to20;
         subu17to20.Edit;
         blob := subu17to20.fields.FieldByName('subu17') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subu17to20.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageM18DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu17to20.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubu17to20;
         subu17to20.Edit;
         blob := subu17to20.fields.FieldByName('subu18') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subu17to20.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageM19DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu17to20.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubu17to20;
```

```
        subu17to20.Edit;
        blob := subu17to20.fields.FieldByName('subu19') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        subu17to20.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageM1DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableUSub1to3.Active Then
      Begin
        DbNavigator5.DataSource:=DataSourceUSub1to3;
        TableUSub1to3.Edit;
        blob := TableUSub1to3.fields.FieldByName('subu1') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        TableUSub1to3.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageM20DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subu17to20.Active Then
      Begin
        DbNavigator2.DataSource:=DataSourcesubu17to20;
        subu17to20.Edit;
        blob := subu17to20.fields.FieldByName('subu20') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        subu17to20.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageM2DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableUSub1to3.Active Then
      Begin
        DbNavigator5.DataSource:=DataSourceUSub1to3;
        TableUSub1to3.Edit;
        blob := TableUSub1to3.fields.FieldByName('subu2') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        TableUSub1to3.UpdateRecord;
      End;
   End;
```

```
end;

procedure TForm3.DBImageM3DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableUSub1to3.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourceUSub1to3;
         TableUSub1to3.Edit;
         blob := TableUSub1to3.fields.FieldByName('subu3') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableUSub1to3.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageM4DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableUSub4to6.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourceUSub4to6;
         TableUSub4to6.Edit;
         blob := TableUSub4to6.fields.FieldByName('subu4') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableUSub4to6.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageM5DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableUSub4to6.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourceUSub4to6;
         TableUSub4to6.Edit;
         blob := TableUSub4to6.fields.FieldByName('subu5') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableUSub4to6.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageM6DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
```

```
  Begin
    If TableUSub4to6.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourceUSub4to6;
         TableUSub4to6.Edit;
         blob := TableUSub4to6.fields.FieldByName('subu6') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableUSub4to6.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageM7DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableUSub7to10.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourceUSub7to10;
         TableUSub7to10.Edit;
         blob := TableUSub7to10.fields.FieldByName('subu7') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableUSub7to10.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageM8DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableUSub7to10.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourceUSub7to10;
         TableUSub7to10.Edit;
         blob := TableUSub7to10.fields.FieldByName('subu8') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableUSub7to10.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageM9DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableUSub7to10.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourceUSub7to10;
         TableUSub7to10.Edit;
         blob := TableUSub7to10.fields.FieldByName('subu9') as TBlobField;
```

```
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        TableUSub7to10.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageP10DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TablePSub7to10.Active Then
      Begin
        DbNavigator5.DataSource:=DataSourcePSub7to10;
        TablePSub7to10.Edit;
        blob := TablePSub7to10.fields.FieldByName('subp10') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        TablePSub7to10.UpdateRecord;
      End;
   End;
end;


procedure TForm3.DBImageP11DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subp11to13.Active Then
      Begin
        DBNavigator2.DataSource:=DataSourcesubp11to13;
        subp11to13.Edit;
        blob := subp11to13.fields.FieldByName('subp11') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        subp11to13.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageP12DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subp11to13.Active Then
      Begin
        DBNavigator2.DataSource:=DataSourcesubp11to13;
        subp11to13.Edit;
        blob := subp11to13.fields.FieldByName('subp12') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        subp11to13.UpdateRecord;
      End;
   End;
end;
```

```
procedure TForm3.DBImageP13DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subp11to13.Active Then
       Begin
         DBNavigator2.DataSource:=DataSourcesubp11to13;
         subp11to13.Edit;
         blob := subp11to13.fields.FieldByName('subp13') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subp11to13.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageP14DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subp14to16.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubp14to16;
         subp14to16.Edit;
         blob := subp14to16.fields.FieldByName('subp14') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subp14to16.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageP15DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subp14to16.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubp14to16;
         subp14to16.Edit;
         blob := subp14to16.fields.FieldByName('subp15') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subp14to16.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageP16DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
```

```
   If subp14to16.Active Then
      Begin
        DbNavigator2.DataSource:=DataSourcesubp14to16;
        subp14to16.Edit;
        blob := subp14to16.fields.FieldByName('subp16') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        subp14to16.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageP17DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subp17to20.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubp17to20;
         subp17to20.Edit;
         blob := subp17to20.fields.FieldByName('subp17') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subp17to20.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageP18DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subp17to20.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubp17to20;
         subp17to20.Edit;
         blob := subp17to20.fields.FieldByName('subp18') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subp17to20.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageP19DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subp17to20.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubp17to20;
         subp17to20.Edit;
         blob := subp17to20.fields.FieldByName('subp19') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
```

```
        subp17to20.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageP1DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TablePSub1to3.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourcePSub1to3;
         TablePSub1to3.Edit;
         blob := TablePSub1to3.fields.FieldByName('subp1') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TablePSub1to3.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBImageP20DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If subp17to20.Active Then
       Begin
         DbNavigator2.DataSource:=DataSourcesubp17to20;
         subp17to20.Edit;
         blob := subp17to20.fields.FieldByName('subp20') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         subp17to20.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBImageP2DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TablePSub1to3.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourcePSub1to3;
         TablePSub1to3.Edit;
         blob := TablePSub1to3.fields.FieldByName('subp2') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TablePSub1to3.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBImageP3DblClick(Sender: TObject);
```

```
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TablePSub1to3.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourcePSub1to3;
         TablePSub1to3.Edit;
         blob := TablePSub1to3.fields.FieldByName('subp3') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TablePSub1to3.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageP4DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TablePSub4to6.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourcePSub4to6;
         TablePSub4to6.Edit;
         blob := TablePSub4to6.fields.FieldByName('subp4') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TablePSub4to6.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageP5DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TablePSub4to6.Active Then
       Begin
         DbNavigator5.DataSource:=DataSourcePSub4to6;
         TablePSub4to6.Edit;
         blob := TablePSub4to6.fields.FieldByName('subp5') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TablePSub4to6.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageP6DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TablePSub4to6.Active Then
       Begin
```

```
        DbNavigator5.DataSource:=DataSourcePSub4to6;
        TablePSub4to6.Edit;
        blob := TablePSub4to6.fields.FieldByName('subp6') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        TablePSub4to6.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageP7DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TablePSub7to10.Active Then
      Begin
        DbNavigator5.DataSource:=DataSourcePSub7to10;
        TablePSub7to10.Edit;
        blob := TablePSub7to10.fields.FieldByName('subp7') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        TablePSub7to10.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageP8DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TablePSub7to10.Active Then
      Begin
        DbNavigator5.DataSource:=DataSourcePSub7to10;
        TablePSub7to10.Edit;
        blob := TablePSub7to10.fields.FieldByName('subp8') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        TablePSub7to10.UpdateRecord;
      End;
   End;
end;


procedure TForm3.DBImageP9DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TablePSub7to10.Active Then
      Begin
        DbNavigator5.DataSource:=DataSourcePSub7to10;
        TablePSub7to10.Edit;
        blob := TablePSub7to10.fields.FieldByName('subp9') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        TablePSub7to10.UpdateRecord;
```

```
      End;
   End;
end;


procedure TForm3.DBImageS10BDblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S7TO10B.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS7TO10B;
         S7TO10B.Edit;
         blob := S7TO10B.fields.FieldByName('subu20') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S7TO10B.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBImageS10DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S7TO10.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS7TO10;
         S7TO10.Edit;
         blob := S7TO10.fields.FieldByName('subp20') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S7TO10.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBImageS1BDblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S1TO3B.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS1TO3B;
         S1TO3B.Edit;
         blob := S1TO3B.fields.FieldByName('subu11') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S1TO3B.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBImageS1DblClick(Sender: TObject);
```

```
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S1to3.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS1TO3;
         S1to3.Edit;
         blob := S1to3.fields.FieldByName('subp11') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S1to3.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageS2BDblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S1TO3B.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS1TO3B;
         S1TO3B.Edit;
         blob := S1TO3B.fields.FieldByName('subu12') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S1TO3B.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageS2DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S1to3.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS1TO3;
         S1to3.Edit;
         blob := S1to3.fields.FieldByName('subp12') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S1to3.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageS3BDblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S1TO3B.Active Then
       Begin
```

```
        DbNavigator6.DataSource:=DataSourceS1TO3B;
        S1TO3B.Edit;
        blob := S1TO3B.fields.FieldByName('subu13') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        S1TO3B.UpdateRecord;
      End;
   End;
end;
procedure TForm3.DBImageS3DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S1to3.Active Then
      Begin
        DbNavigator6.DataSource:=DataSourceS1TO3;
        S1to3.Edit;
        blob := S1to3.fields.FieldByName('subp13') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        S1to3.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageS4BDblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S4TO6B.Active Then
      Begin
        DbNavigator6.DataSource:=DataSourceS4TO6B;
        S4TO6B.Edit;
        blob := S4TO6B.fields.FieldByName('subu14') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        S4TO6B.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageS4DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S4To6.Active Then
      Begin
        DbNavigator6.DataSource:=DataSourceS4To6;
        S4To6.Edit;
        blob := S4To6.fields.FieldByName('subp14') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        S4To6.UpdateRecord;
      End;
   End;
```

```
end;

procedure TForm3.DBImageS5BDblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S4TO6B.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS4TO6B;
         S4TO6B.Edit;
         blob := S4TO6B.fields.FieldByName('subu15') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S4TO6B.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageS5DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S4To6.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS4To6;
         S4To6.Edit;
         blob := S4To6.fields.FieldByName('subp15') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S4To6.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageS6BDblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S4TO6B.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS4TO6B;
         S4TO6B.Edit;
         blob := S4TO6B.fields.FieldByName('subu16') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S4TO6B.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageS6DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
```

```
  Begin
    If S4To6.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS4To6;
         S4To6.Edit;
         blob := S4To6.fields.FieldByName('subp16') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S4To6.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageS7BDblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S7TO10B.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS7TO10B;
         S7TO10B.Edit;
         blob := S7TO10B.fields.FieldByName('subu17') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S7TO10B.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageS7DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S7TO10.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS7TO10;
         S7TO10.Edit;
         blob := S7TO10.fields.FieldByName('subp17') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         S7TO10.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBImageS8BDblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S7TO10B.Active Then
       Begin
         DbNavigator6.DataSource:=DataSourceS7TO10B;
         S7TO10B.Edit;
         blob := S7TO10B.fields.FieldByName('subu18') as TBlobField;
```

```
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        S7TO10B.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBImageS8DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S7TO10.Active Then
       Begin
        DbNavigator6.DataSource:=DataSourceS7TO10;
        S7TO10.Edit;
        blob := S7TO10.fields.FieldByName('subp18') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        S7TO10.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBImageS9BDblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S7TO10B.Active Then
       Begin
        DbNavigator6.DataSource:=DataSourceS7TO10B;
        S7TO10B.Edit;
        blob := S7TO10B.fields.FieldByName('subu19') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        S7TO10B.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBImageS9DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If S7TO10.Active Then
       Begin
        DbNavigator6.DataSource:=DataSourceS7TO10;
        S7TO10.Edit;
        blob := S7TO10.fields.FieldByName('subp19') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        S7TO10.UpdateRecord;
       End;
   End;
end;
```

```
procedure TForm3.DBIn10DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If Table6to10.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceI6to10;
         Table6to10.Edit;
         blob := Table6to10.fields.FieldByName('In10') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         Table6to10.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBIn1DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If Table2.Active Then
       Begin
         Table2.Edit;
         blob := Table2.fields.FieldByName('In1') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         Table2.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBIn2DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If Table2.Active Then
       Begin
         Table2.Edit;
         blob := Table2.fields.FieldByName('In2') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         Table2.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBIn3DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableIn3to5.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceI3to5;
```

```
        TableIn3to5.Edit;
        blob := TableIn3to5.fields.FieldByName('In3') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        TableIn3to5.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBIn4DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableIn3to5.Active Then
      Begin
        DbNavigator3.DataSource:=DataSourceI3to5;
        TableIn3to5.Edit;
        blob := TableIn3to5.fields.FieldByName('In4') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        TableIn3to5.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBIn5DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableIn3to5.Active Then
      Begin
        DbNavigator3.DataSource:=DataSourceI3to5;
        TableIn3to5.Edit;
        blob := TableIn3to5.fields.FieldByName('In5') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        TableIn3to5.UpdateRecord;
      End;
   End;
end;
procedure TForm3.DBIn6DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If Table2.Active Then
      Begin
        DbNavigator3.DataSource:=DataSourceI6to10;
        Table6to10.Edit;
        blob := Table6to10.fields.FieldByName('In6') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        Table6to10.UpdateRecord;
      End;
   End;
end;
```

```
procedure TForm3.DBIn7DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If Table6to10.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceI6to10;
         Table6to10.Edit;
         blob := Table6to10.fields.FieldByName('In7') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         Table6to10.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBIn8DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If Table6to10.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceI6to10;
         Table6to10.Edit;
         blob := Table6to10.fields.FieldByName('In8') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         Table6to10.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBIn9DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If Table6to10.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceI6to10;
         Table6to10.Edit;
         blob := Table6to10.fields.FieldByName('In9') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         Table6to10.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBOut10DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
```

```
   If Table7to10.Active Then
      Begin
        DbNavigator3.DataSource:=DataSourceO7to10;
        Table7to10.Edit;
        blob := Table7to10.fields.FieldByName('Out10') as TBlobField;
        Blob.LoadFromFile(OpenPictureDialog1.FileName);
        Table7to10.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBOut1DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableO1to3.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceO1to3;
         TableO1to3.Edit;
         blob := TableO1to3.fields.FieldByName('Out1') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableO1to3.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBOut2DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableO1to3.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceO1to3;
         TableO1to3.Edit;
         blob := TableO1to3.fields.FieldByName('Out2') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableO1to3.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBOut3DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableO1to3.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceO1to3;
         TableO1to3.Edit;
         blob := TableO1to3.fields.FieldByName('Out3') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
```

```
        TableO1to3.UpdateRecord;
      End;
   End;
end;

procedure TForm3.DBOut4DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableO4to6.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceO4to6;
         TableO4to6.Edit;
         blob := TableO4to6.fields.FieldByName('Out4') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableO4to6.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBOut5DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableO4to6.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceO4to6;
         TableO4to6.Edit;
         blob := TableO4to6.fields.FieldByName('Out5') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableO4to6.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBOut6DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If TableO4to6.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceO4to6;
         TableO4to6.Edit;
         blob := TableO4to6.fields.FieldByName('Out6') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         TableO4to6.UpdateRecord;
       End;
   End;
end;

procedure TForm3.DBOut7DblClick(Sender: TObject);
```

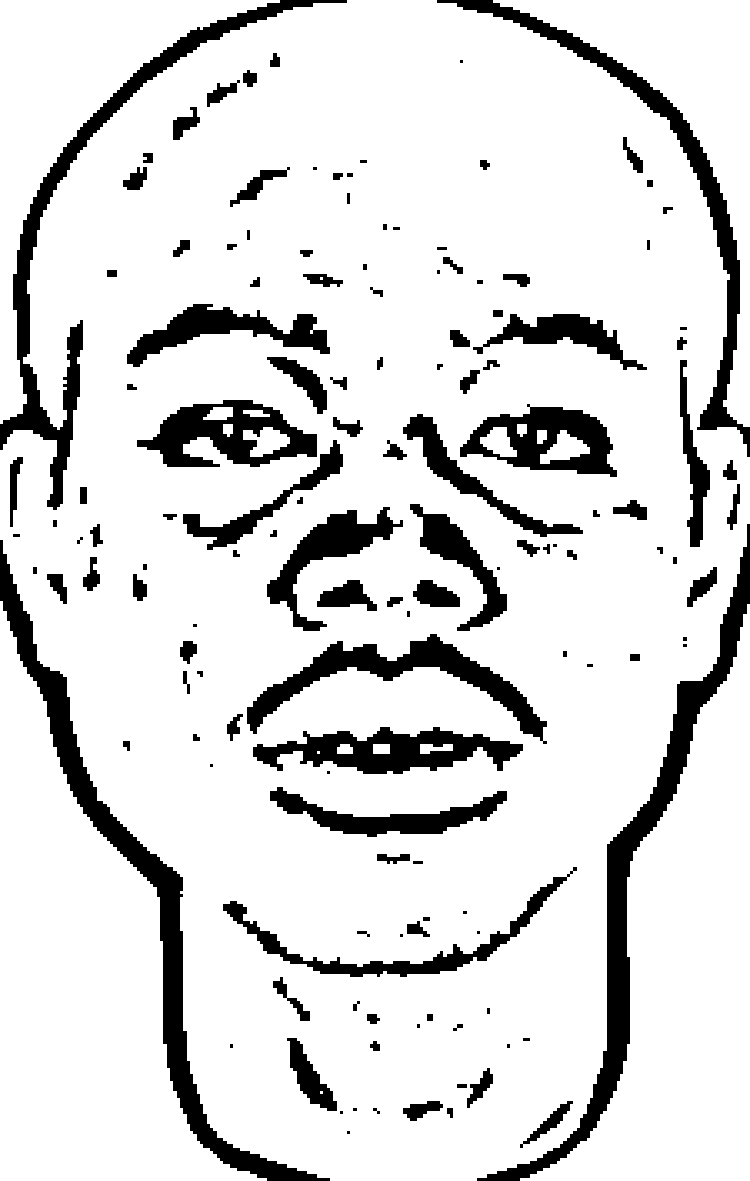
```
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If Table7to10.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceO7to10;
         Table7to10.Edit;
         blob := Table7to10.fields.FieldByName('Out7') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         Table7to10.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBOut8DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If Table7to10.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceO7to10;
         Table7to10.Edit;
         blob := Table7to10.fields.FieldByName('Out8') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         Table7to10.UpdateRecord;
       End;
    End;
end;

procedure TForm3.DBOut9DblClick(Sender: TObject);
Var blob: TBlobField;
begin
  if OpenPictureDialog1.Execute  then
  Begin
    If Table7to10.Active Then
       Begin
         DbNavigator3.DataSource:=DataSourceO7to10;
         Table7to10.Edit;
         blob := Table7to10.fields.FieldByName('Out9') as TBlobField;
         Blob.LoadFromFile(OpenPictureDialog1.FileName);
         Table7to10.UpdateRecord;
       End;
    End;
end;

procedure TForm3.FormClose(Sender: TObject; var Action: TCloseAction);
begin
 ADOTable1.Active:=false;
 ADoTable1.Close;
 ADOCommand1.Free;
 ADOCommand2.Free;
 ADOConnection1.Close;
```

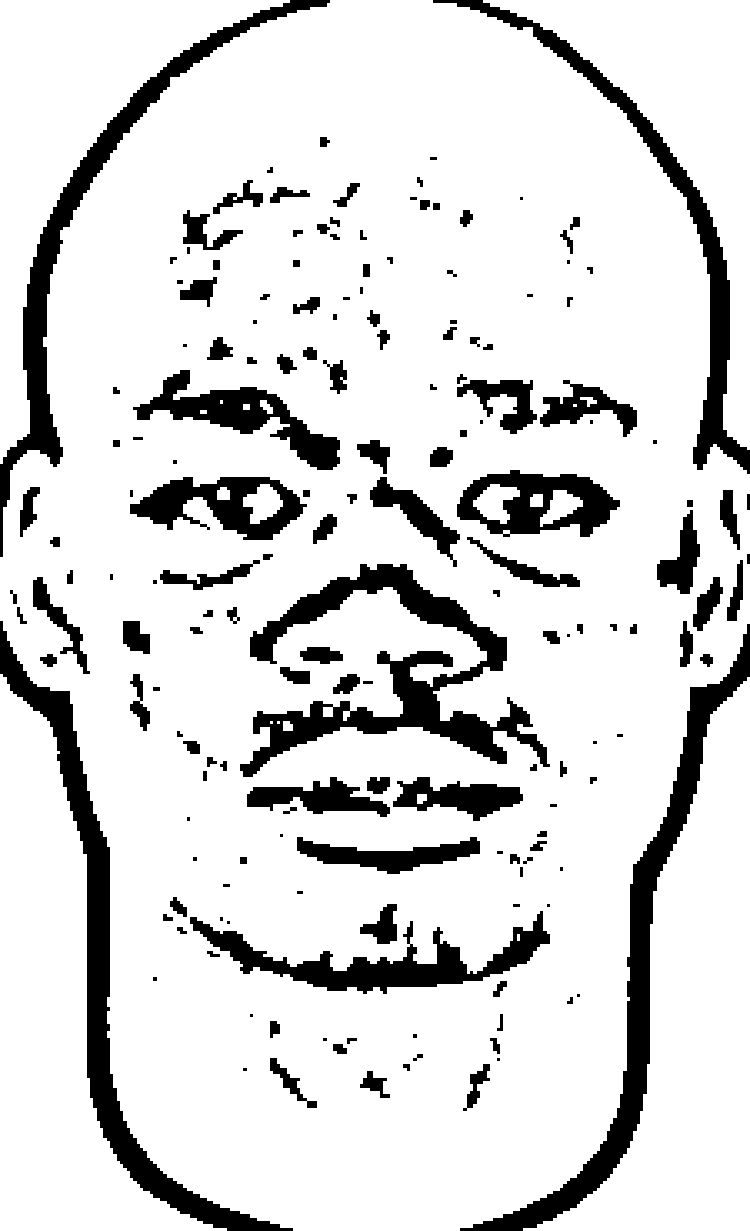
```
 ADOConnection2.Close;
 Table2.Active:=false;
 ADOConnection3.Close;
end;

procedure TForm3.FormShow(Sender: TObject);
begin
//  UpdateFields;
end;

procedure TForm3.M1OpenClick(Sender: TObject);
Var blob:  TBlobField;
begin
  if OpenDialog2.Execute then
       If Table2.Active Then
       Begin
         Table2.Edit;
         blob := Table2.fields.FieldByName('M1') as TBlobField;
         Blob.LoadFromFile(OpenDialog2.FileName);
       End;
end;

Procedure TForm3.UpdateColour;
Var RR,GG,BB : Byte;
Begin
  RR:=StrToInt(S1C1R.Field.Value);
  GG:=StrToInt(S1C1G.Field.Value);
  BB:=StrToInt(S1C1B.Field.Value);
  S1C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S1C2R.Field.Value);
  GG:=StrToInt(S1C2G.Field.Value);
  BB:=StrToInt(S1C2B.Field.Value);
  S1C2.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S2C1R.Field.Value);
  GG:=StrToInt(S2C1G.Field.Value);
  BB:=StrToInt(S2C1B.Field.Value);
  S2C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S2C2R.Field.Value);
  GG:=StrToInt(S2C2G.Field.Value);
  BB:=StrToInt(S2C2B.Field.Value);
  S2C2.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S3C1R.Field.Value);
  GG:=StrToInt(S3C1G.Field.Value);
  BB:=StrToInt(S3C1B.Field.Value);
  S3C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S3C2R.Field.Value);
  GG:=StrToInt(S3C2G.Field.Value);
  BB:=StrToInt(S3C2B.Field.Value);
  S3C2.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S4C1R.Field.Value);
  GG:=StrToInt(S4C1G.Field.Value);
  BB:=StrToInt(S4C1B.Field.Value);
  S4C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S4C2R.Field.Value);
```

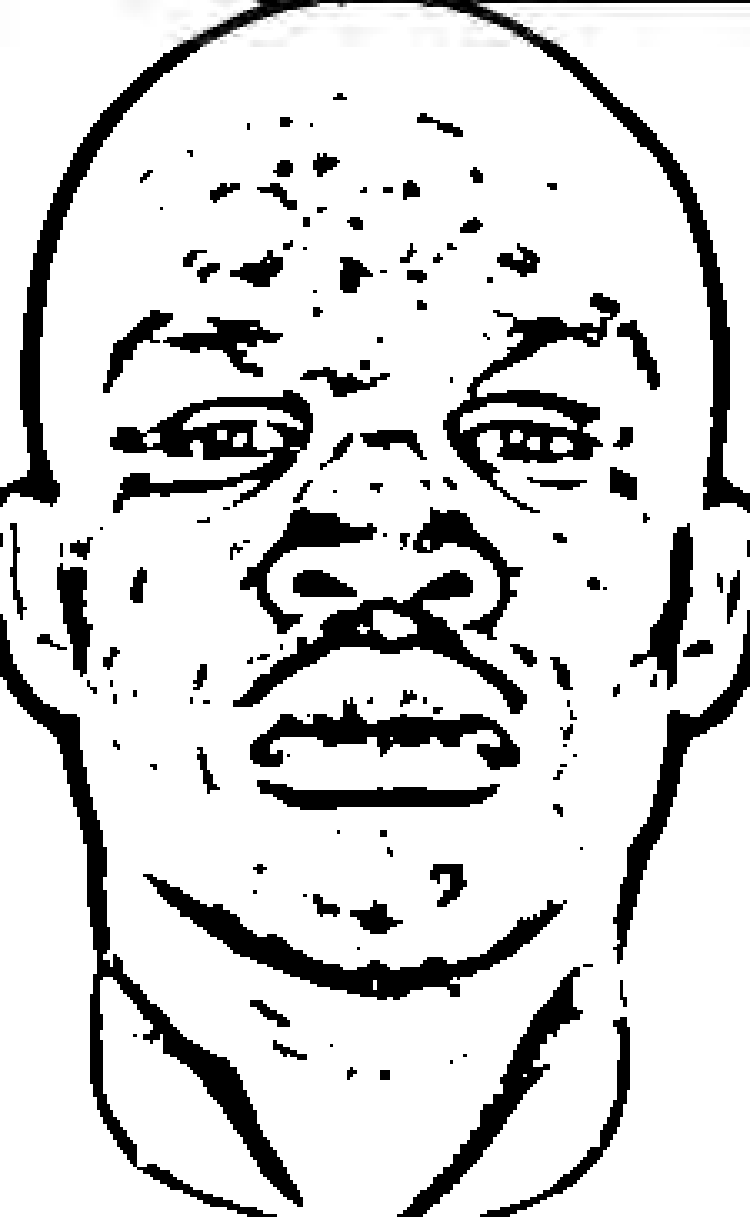
```
  GG:=StrToInt(S4C2G.Field.Value);
  BB:=StrToInt(S4C2B.Field.Value);
  S4C2.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S5C1R.Field.Value);
  GG:=StrToInt(S5C1G.Field.Value);
  BB:=StrToInt(S5C1B.Field.Value);
  S5C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S5C2R.Field.Value);
  GG:=StrToInt(S5C2G.Field.Value);
  BB:=StrToInt(S5C2B.Field.Value);
  S5C2.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToINt(FormR.Field.Value);
  GG:=StrToINt(FormG.Field.Value);
  BB:=StrToINt(FormB.Field.Value);
  FormR.Color:=RGB(RR,GG,BB);
  FormG.Color:=RGB(RR,GG,BB);
  FormB.Color:=RGB(RR,GG,BB);
  RR:=StrToINt(FieldR.Field.Value);
  GG:=StrToINt(FieldG.Field.Value);
  BB:=StrToINt(FieldB.Field.Value);
  FieldR.Color:=RGB(RR,GG,BB);
  FieldG.Color:=RGB(RR,GG,BB);
  FieldB.Color:=RGB(RR,GG,BB);
  RR:=StrToINt(StimulusR.Field.Value);
  GG:=StrToINt(StimulusG.Field.Value);
  BB:=StrToINt(StimulusB.Field.Value);
  StimulusR.Color:=RGB(RR,GG,BB);
  StimulusG.Color:=RGB(RR,GG,BB);
  StimulusB.Color:=RGB(RR,GG,BB);
  RR:=StrToINt(PC1R.Field.Value);
  GG:=StrToINt(PC1G.Field.Value);
  BB:=StrToINt(PC1B.Field.Value);
  PC1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToINt(PC2R.Field.Value);
  GG:=StrToINt(PC2G.Field.Value);
  BB:=StrToINt(PC2B.Field.Value);
  PC2.Brush.Color:=RGB(RR,GG,BB);
End;

procedure TForm3.T20S1Click(Sender: TObject);
Var RR,GG,BB : Byte;
begin
  RR:=StrToInt(S1C1R.Text);
  GG:=StrToInt(S1C1G.Text);
  BB:=StrToInt(S1C1B.Text);
  S1C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S1C2R.Text);
  GG:=StrToInt(S1C2G.Text);
  BB:=StrToInt(S1C2B.Text);
  S1C2.Brush.Color:=RGB(RR,GG,BB);
end;

procedure TForm3.T20S2Click(Sender: TObject);
Var RR,GG,BB : Byte;
```

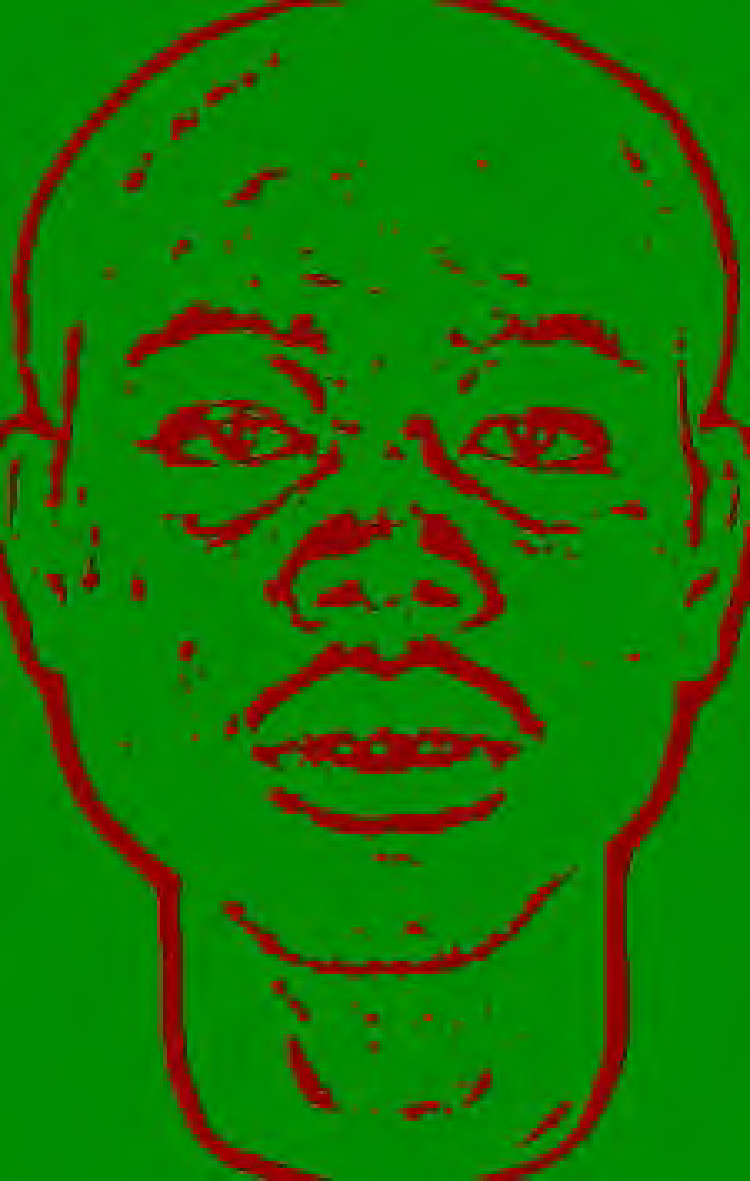
```
begin
  RR:=StrToInt(S2C1R.Text);
  GG:=StrToInt(S2C1G.Text);
  BB:=StrToInt(S2C1B.Text);
  S2C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S2C2R.Text);
  GG:=StrToInt(S2C2G.Text);
  BB:=StrToInt(S2C2B.Text);
  S2C2.Brush.Color:=RGB(RR,GG,BB);
end;

procedure TForm3.T20S3Click(Sender: TObject);
Var RR,GG,BB : Byte;
begin
  RR:=StrToInt(S3C1R.Text);
  GG:=StrToInt(S3C1G.Text);
  BB:=StrToInt(S3C1B.Text);
  S3C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S3C2R.Text);
  GG:=StrToInt(S3C2G.Text);
  BB:=StrToInt(S3C2B.Text);
  S3C2.Brush.Color:=RGB(RR,GG,BB);
end;

procedure TForm3.T20S4Click(Sender: TObject);
Var RR,GG,BB : Byte;
begin
  RR:=StrToInt(S4C1R.Text);
  GG:=StrToInt(S4C1G.Text);
  BB:=StrToInt(S4C1B.Text);
  S4C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S4C2R.Text);
  GG:=StrToInt(S4C2G.Text);
  BB:=StrToInt(S4C2B.Text);
  S4C2.Brush.Color:=RGB(RR,GG,BB);
end;

procedure TForm3.T20S5Click(Sender: TObject);
Var RR,GG,BB : Byte;
begin
  RR:=StrToInt(S5C1R.Text);
  GG:=StrToInt(S5C1G.Text);
  BB:=StrToInt(S5C1B.Text);
  S5C1.Brush.Color:=RGB(RR,GG,BB);
  RR:=StrToInt(S5C2R.Text);
  GG:=StrToInt(S5C2G.Text);
  BB:=StrToInt(S5C2B.Text);
  S5C2.Brush.Color:=RGB(RR,GG,BB);
end;

procedure TForm3.Table2AfterOpen(DataSet: TDataSet);
begin
 // LEDConnect.Appearance.Fill.Color:=ClLime;
end;
```

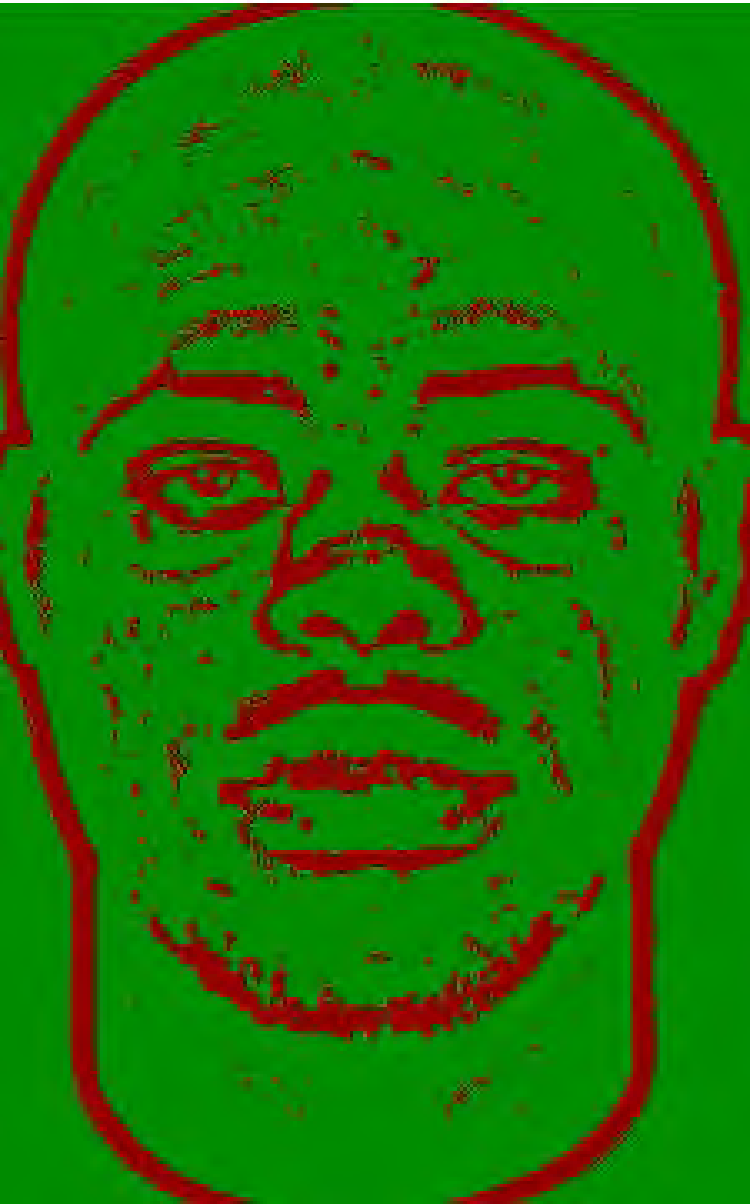
end.

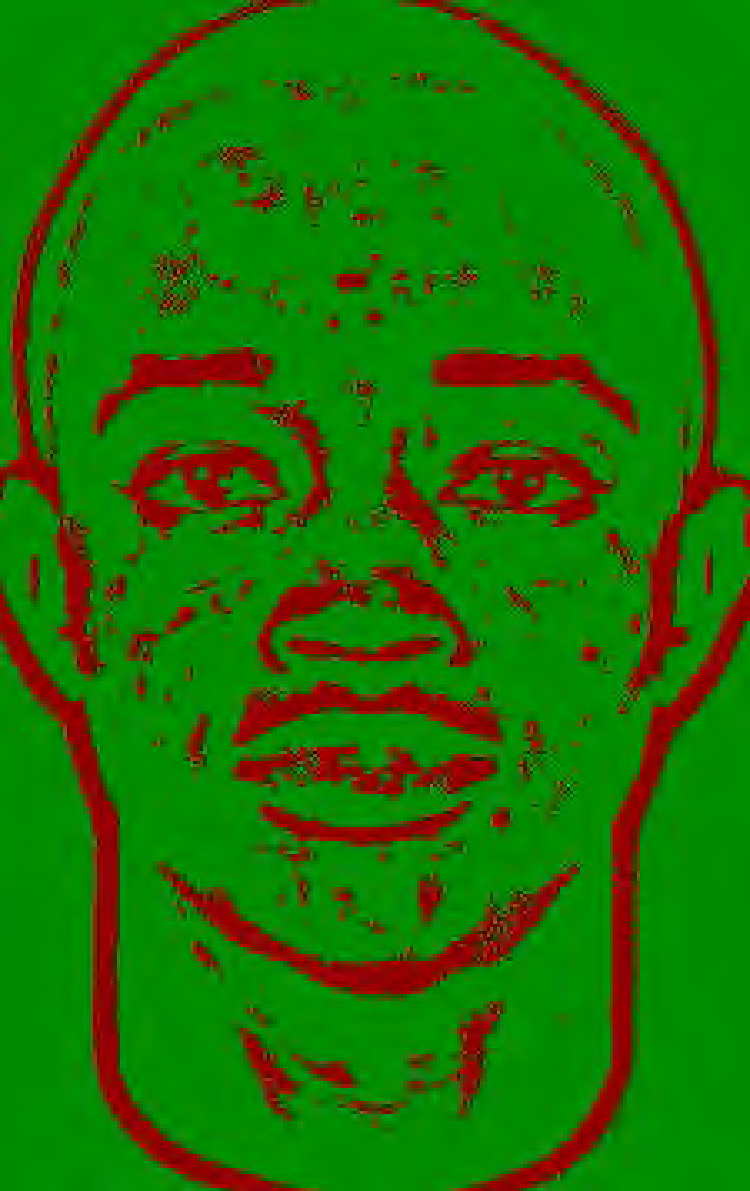


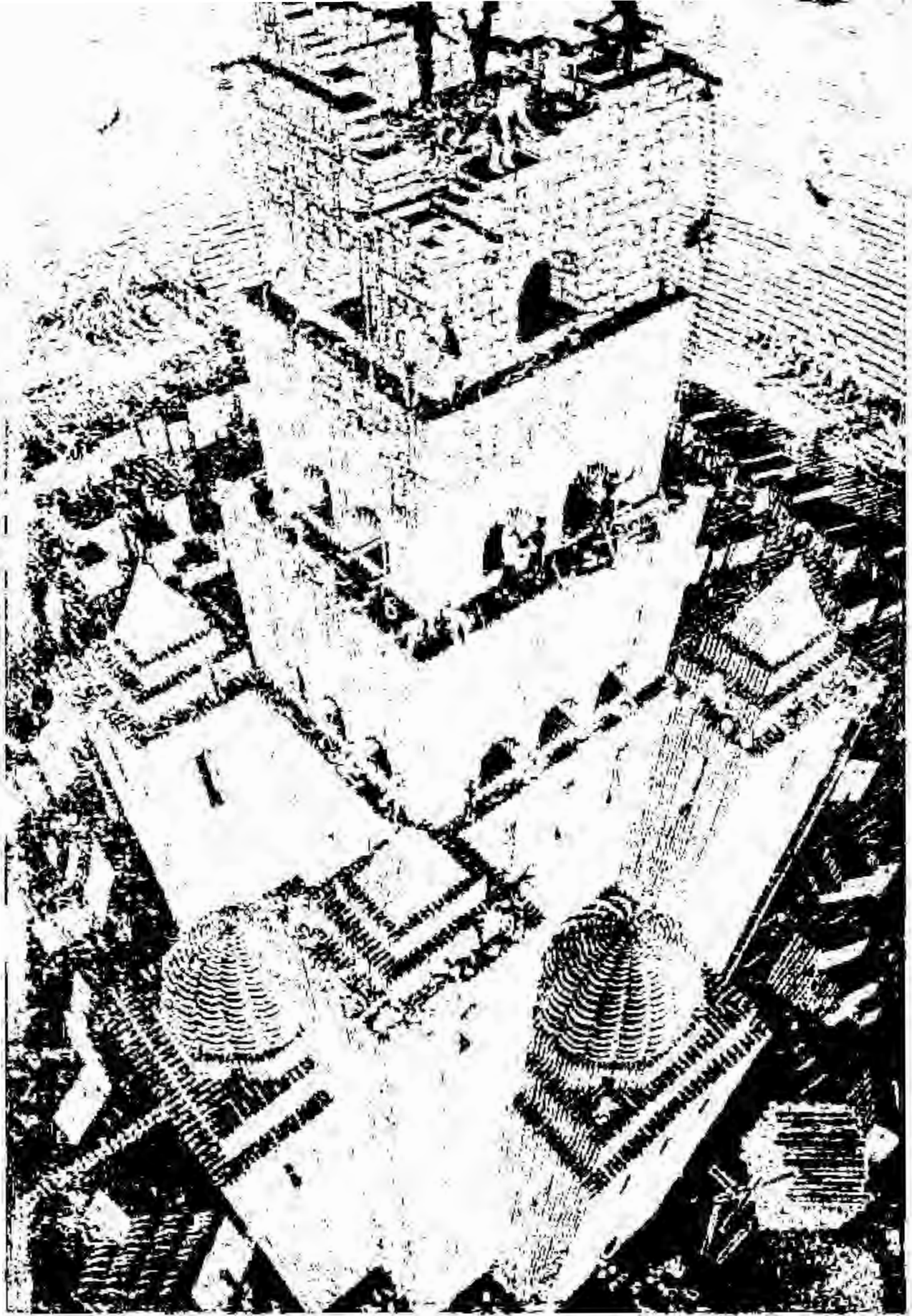


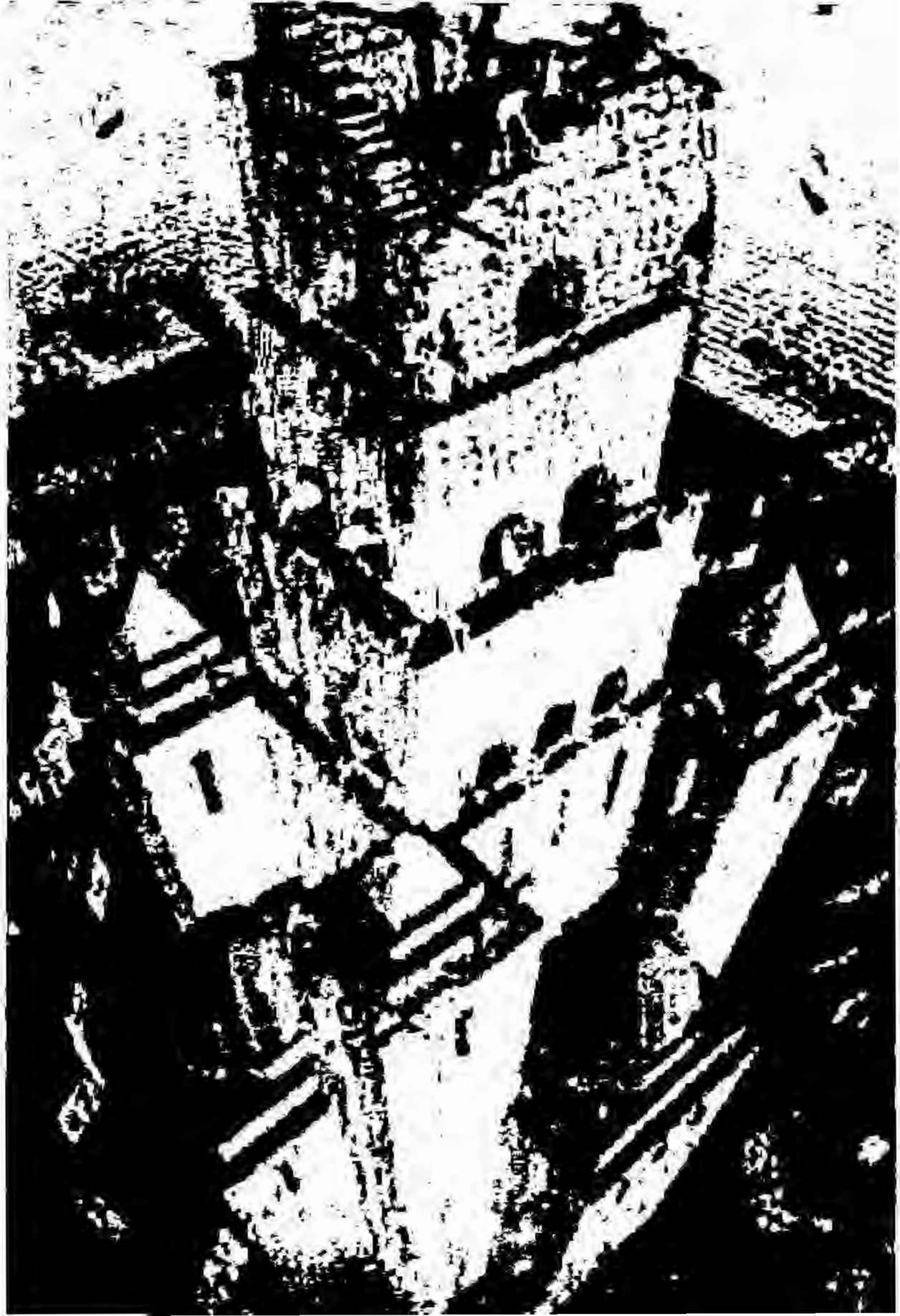


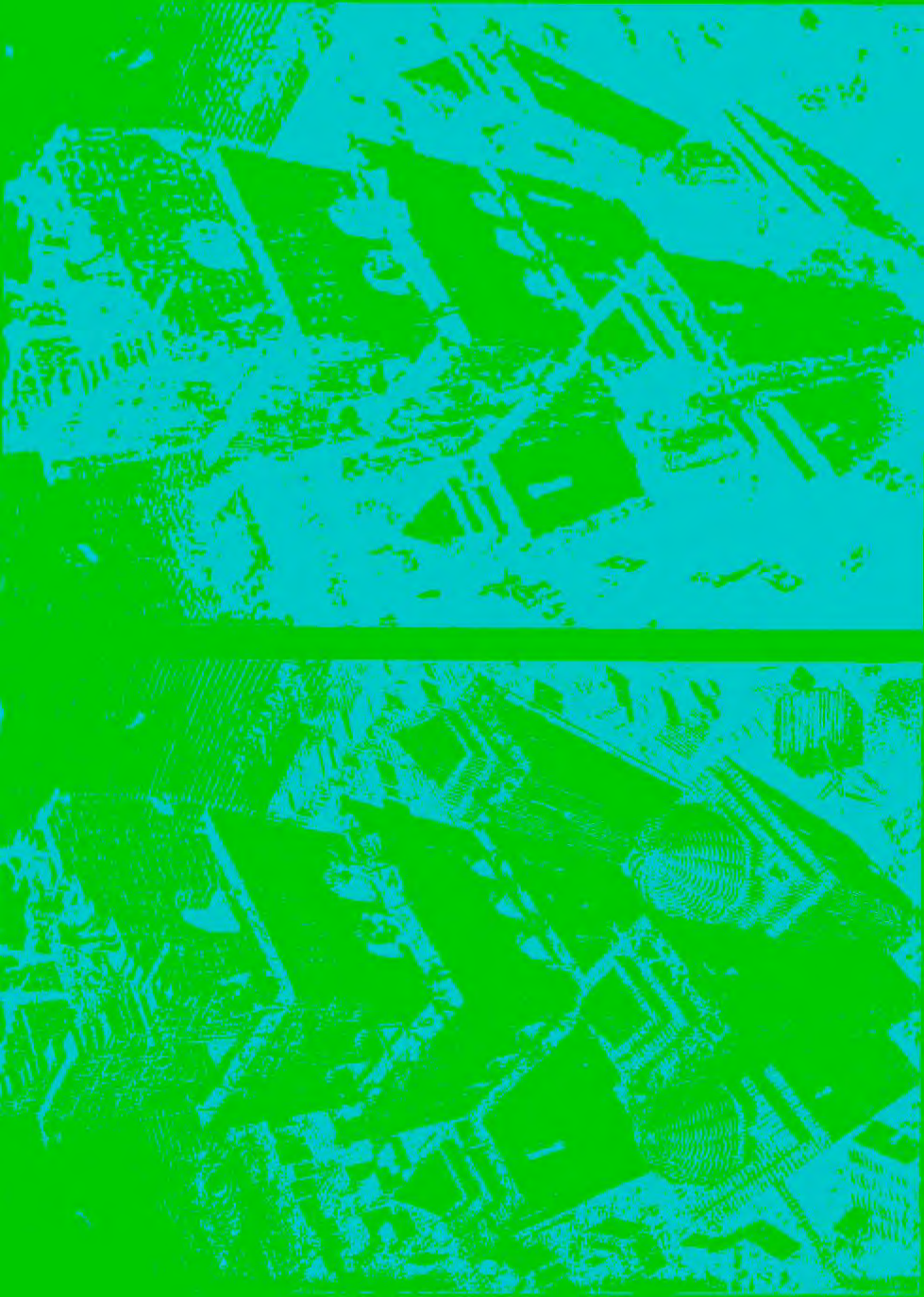


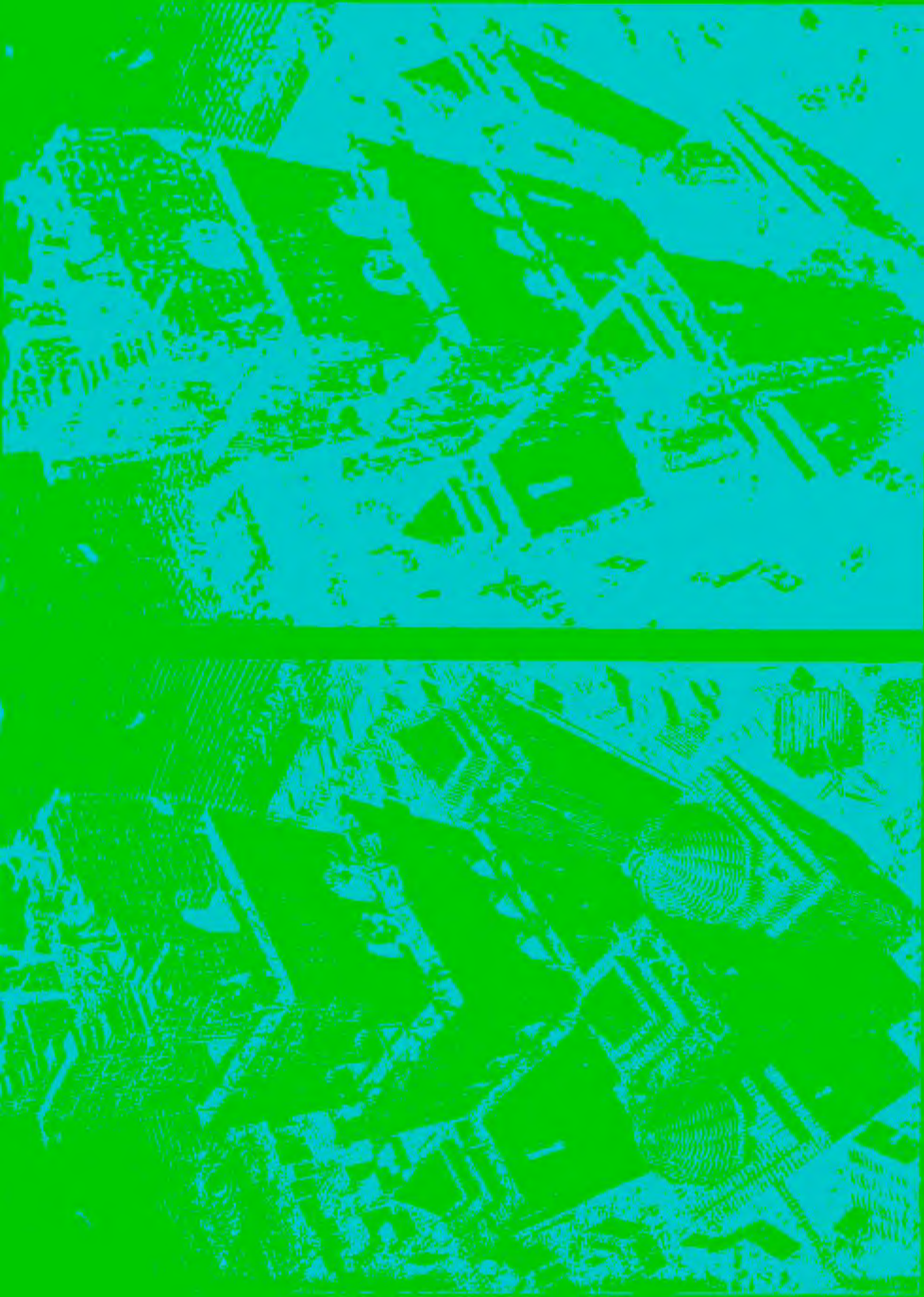






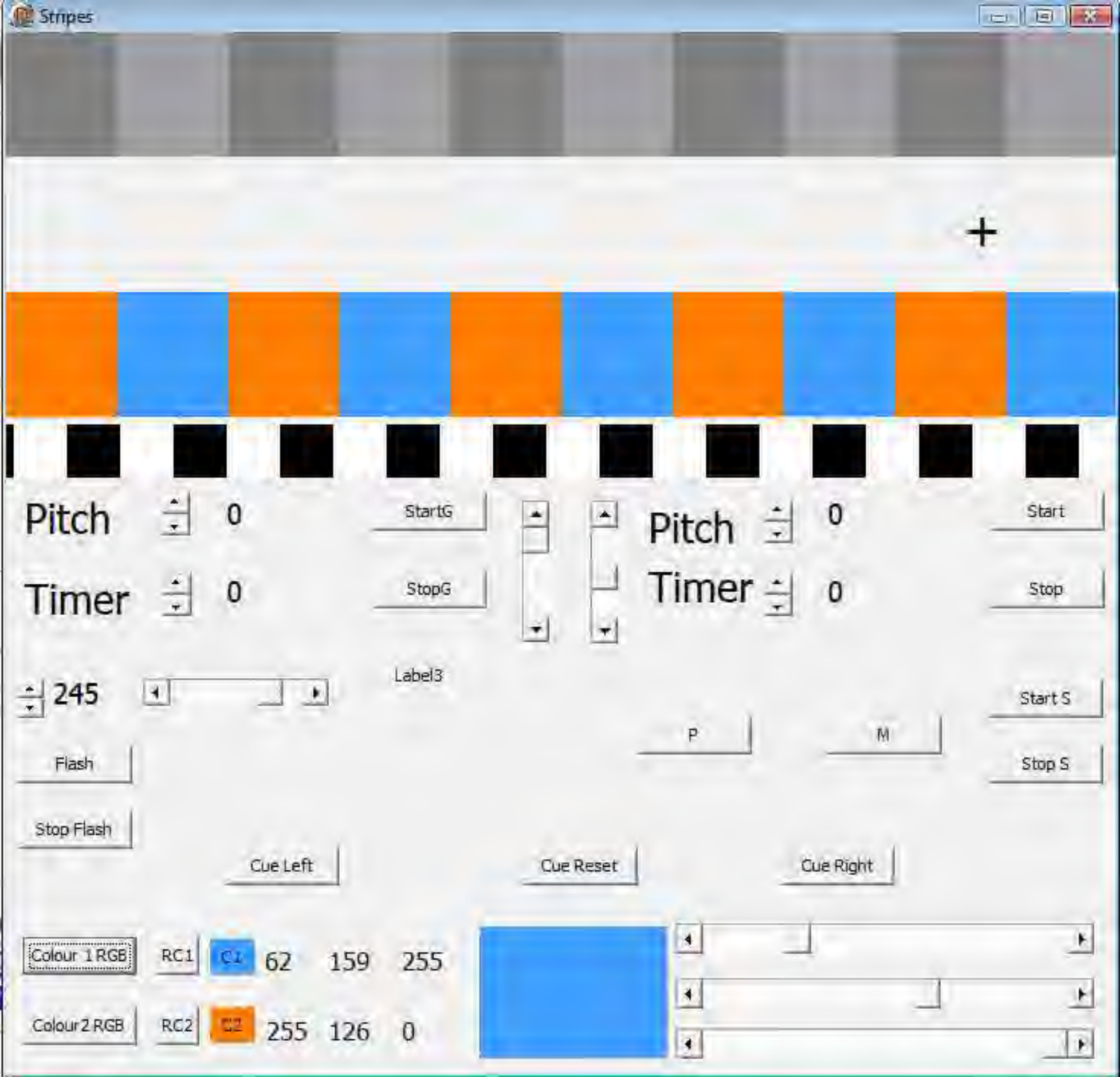




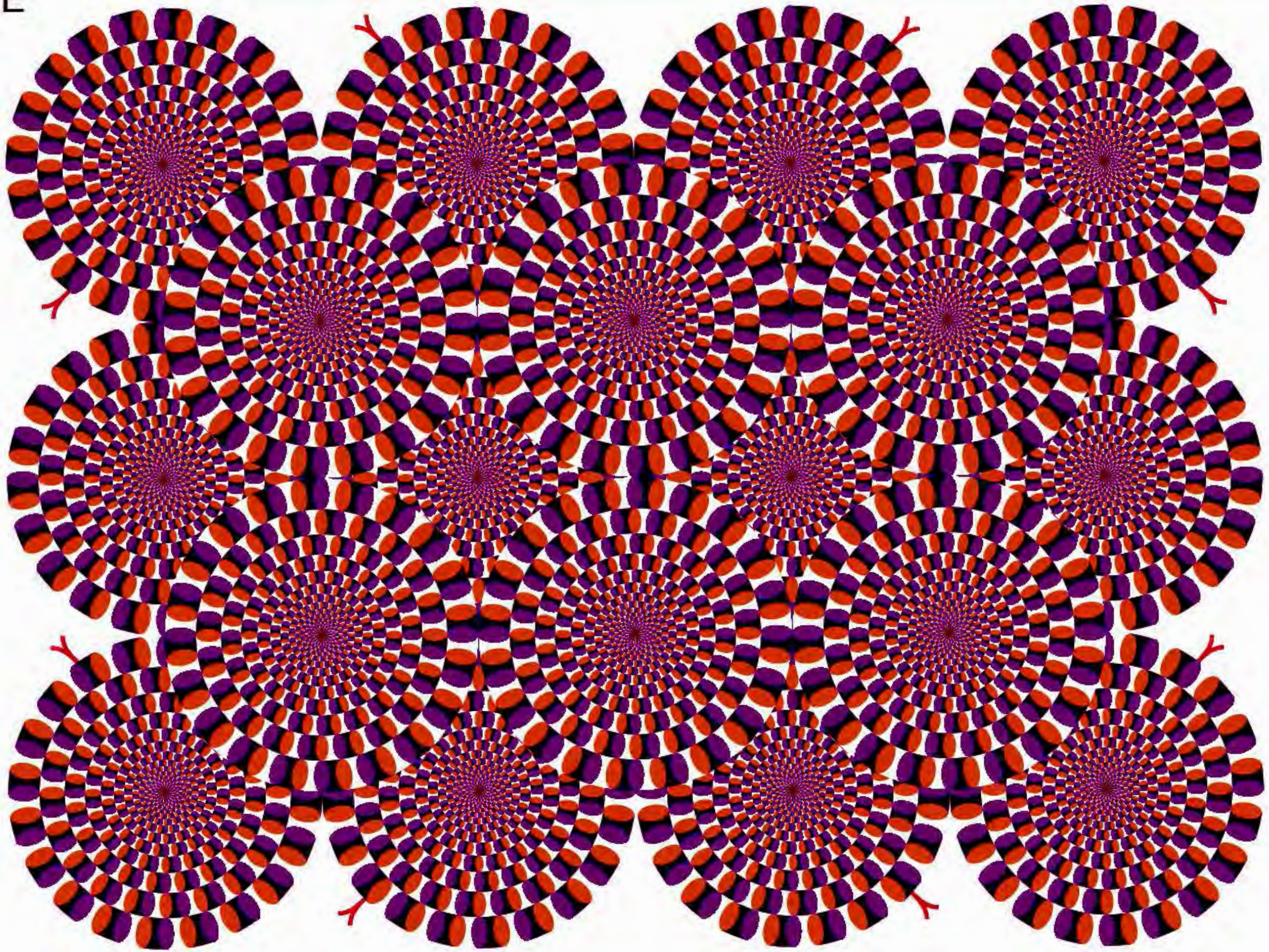


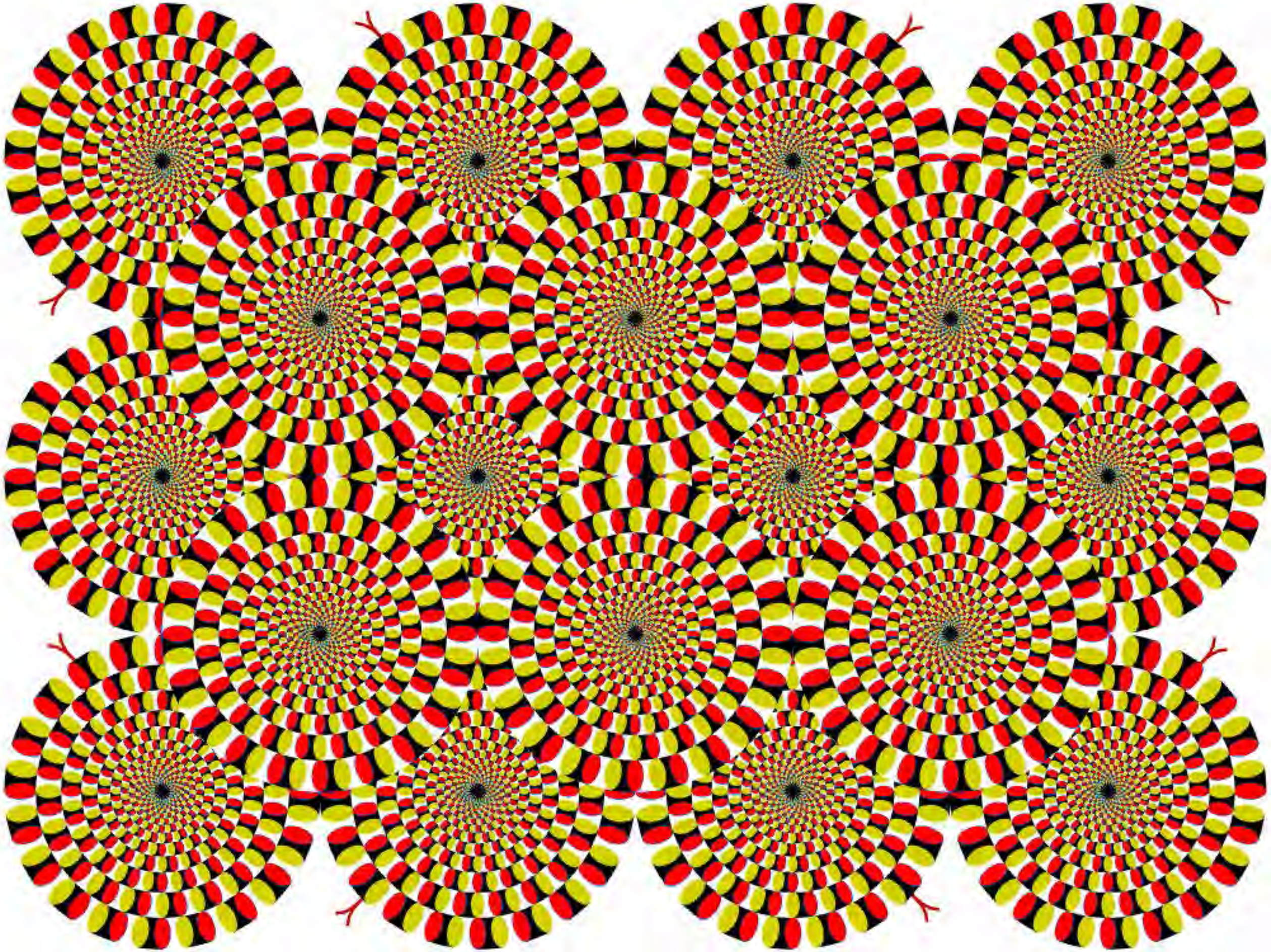
```
library NORMSINV;

{ Important note about DLL memory management: ShareMem must be the
  first unit in your library's USES clause AND your project's (select
  Project-View Source) USES clause if your DLL exports any procedures or
  functions that pass strings as parameters or function results. This
  applies to all strings passed to and from your DLL--even those that
  are nested in records and classes. ShareMem is the interface unit to
  the BORLNDMM.DLL shared memory manager, which must be deployed along
  with your DLL. To avoid using BORLNDMM.DLL, pass string information
  using PChar or ShortString parameters. }

uses
  Math,
  SysUtils,
  Classes;

{$R *.res}

const a: array[1..6] of double=(

-3.969683028665376e+01,
  2.209460984245205e+02,
 -2.759285104469687e+02,
  1.383577518672690e+02,
-3.066479806614716e+01,
 2.506628277459239e+00 );

const b: array[1..5] of double =(

 -5.447609879822406e+01,
 1.615858368580409e+02,
 -1.556989798598866e+02,
  6.680131188771972e+01,
 -1.328068155288572e+01  );

const c: array[1..6] of double=(

 -7.784894002430293e-03,
 -3.223964580411365e-01,
 -2.400758277161838e+00,
 -2.549732539343734e+00,
  4.374664141464968e+00,
  2.938163982698783e+00  );

const d: array[1..4] of double=(
 7.784695709041462e-03,
 3.224671290700398e-01,
 2.445134137142996e+00,
 3.754408661907416e+00 );

//Define break-points.
```

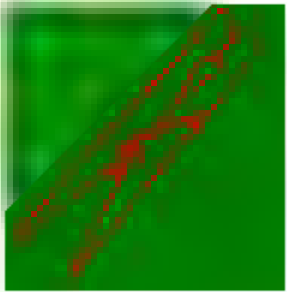
```
const p_low = 0.02425;
const p_high = 1 - p_low ;
//const infinity=1.7E308; // maximum size of double

Procedure NormZ(Var p,ZScore: double) Export;
var q,r: Double;

  begin
  if ((p<=0) or (p>=1)) then raise Einvalidargument.create('Inverse Cum Norm argument must be in range 0<p<1');

  //Rational approximation for lower region.
  if ((0 < p) and  (p< p_low)) then begin
    q := sqrt(-2*ln(p));
    ZScore := (((((c[1]*q+c[2])*q+c[3])*q+c[4])*q+c[5])*q+c[6]) /
      ((((d[1]*q+d[2])*q+d[3])*q+d[4])*q+1);
  end;

 //Rational approximation for central region.
 if (p_low <= p) and (p <= p_high ) then begin
    q := p - 0.5 ;
    r := q*q ;
    ZScore := (((((a[1]*r+a[2])*r+a[3])*r+a[4])*r+a[5])*r+a[6])*q /
     (((((b[1]*r+b[2])*r+b[3])*r+b[4])*r+b[5])*r+1) ;
  end;

  //Rational approximation for upper region.
  if ((p_high < p) and (p < 1))  then begin
    q := sqrt(-2*ln(1-p));
    ZScore := -(((((c[1]*q+c[2])*q+c[3])*q+c[4])*q+c[5])*q+c[6]) /
      ((((d[1]*q+d[2])*q+d[3])*q+d[4])*q+1)  ;
  end;
end;

Exports NormZ;

begin
end.
```

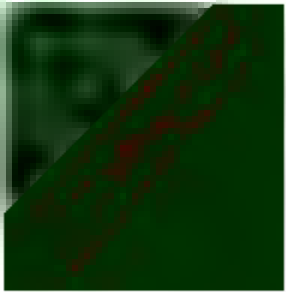

Stripes
Pitch
0
StartG
Pitch
0
Start
Timer
0
StopG
Timer
0
Stop
245
Label3
Start S
P
M
Flash
Stop S
Stop Flash
Cue Left
Cue Reset
Cue Right
Colour 1 RGB
RC1
C1
62
159
255
Colour 2 RGB
RC2
C2
255
126
0

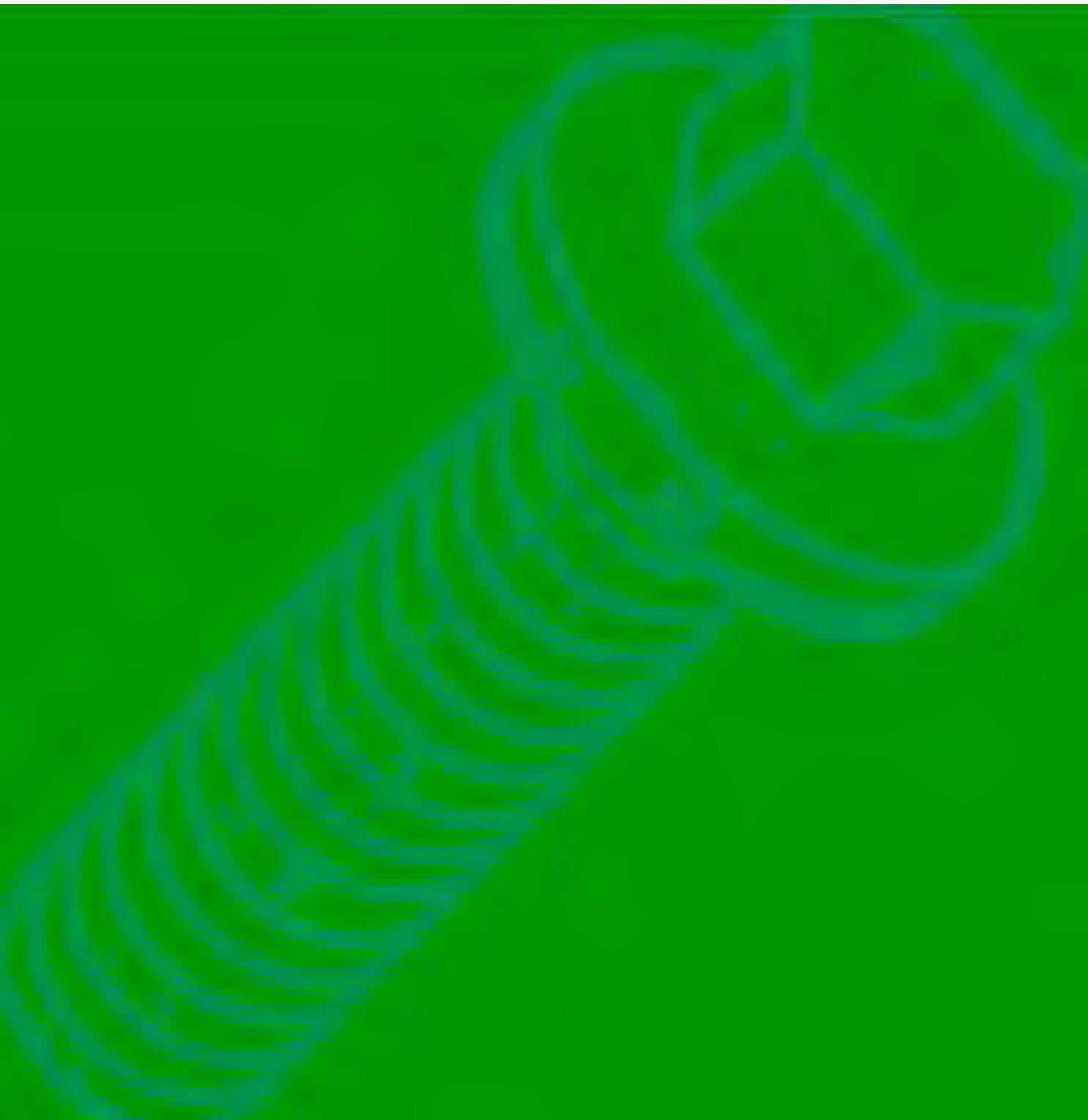
E

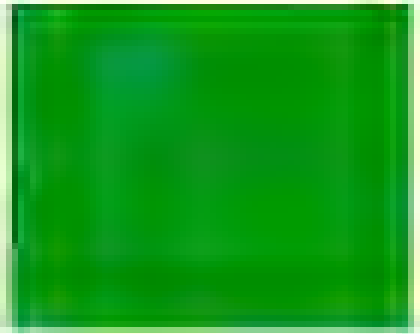












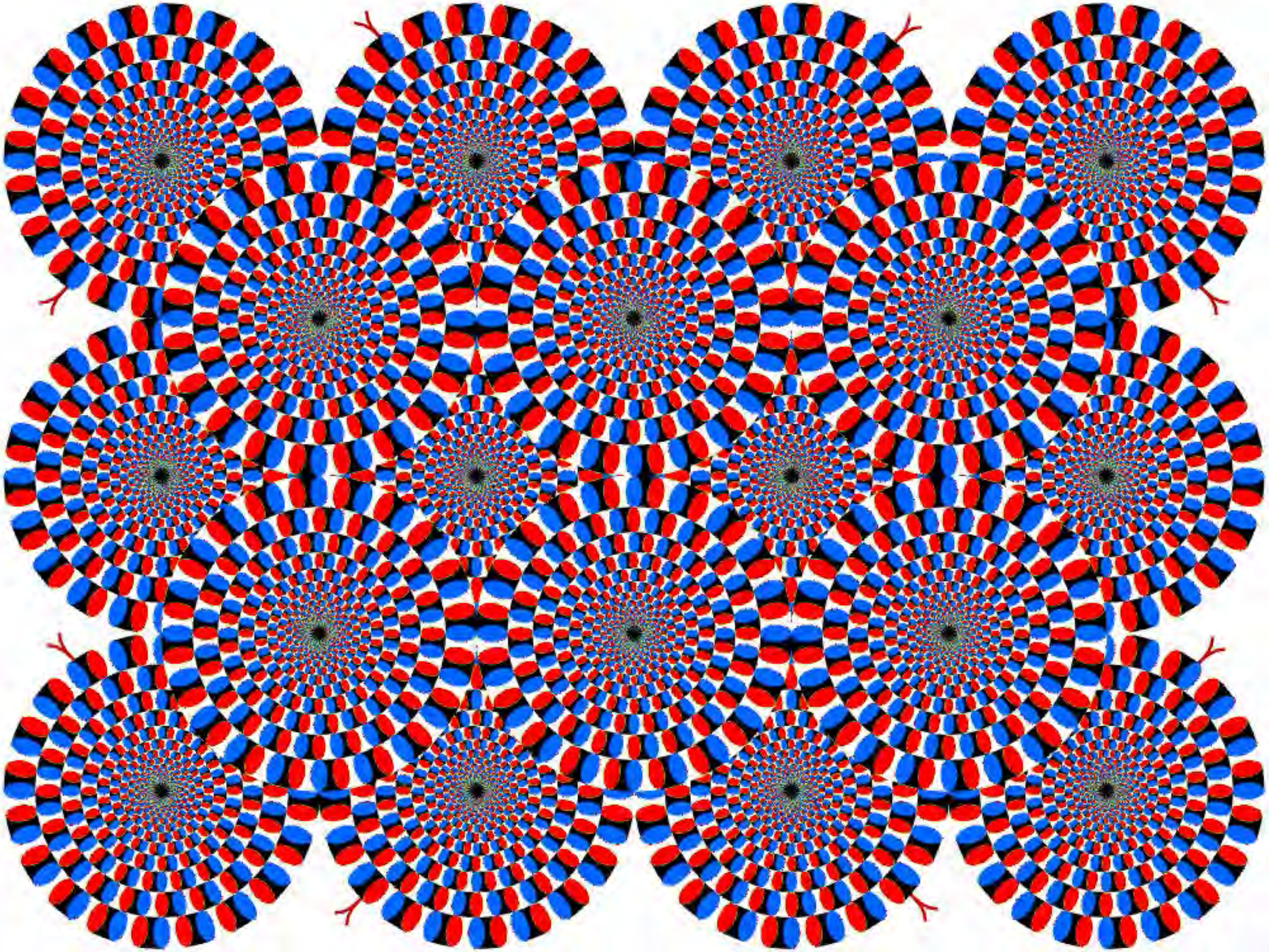


Image Processor

BackGround

50 120 50

200 0 0

Enabled

ForeGround

180 180 180

0 200 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

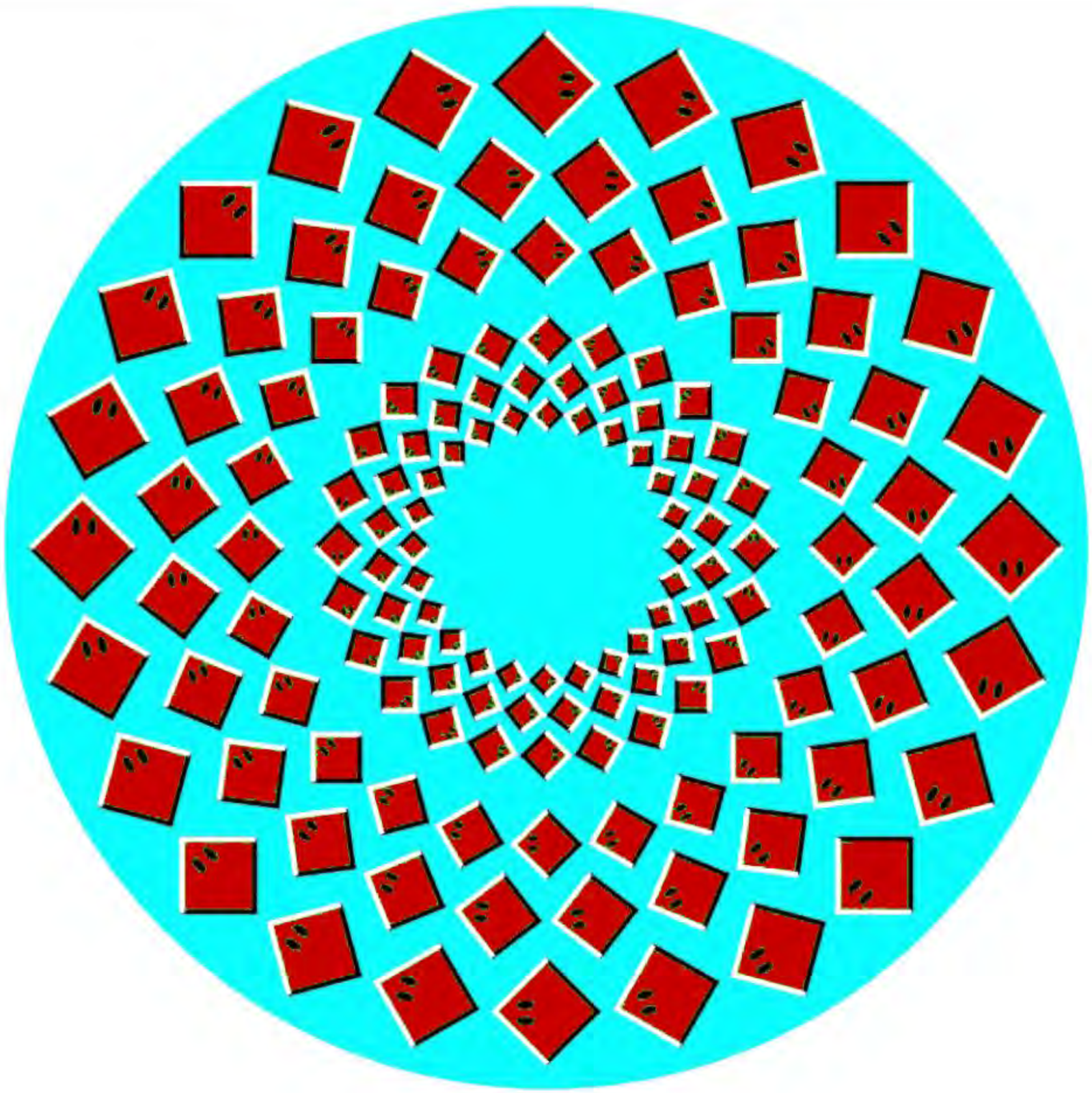


Image Processor

BackGround

50 120 50

0 120 0

Enabled

ForeGround

180 180 180

120 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

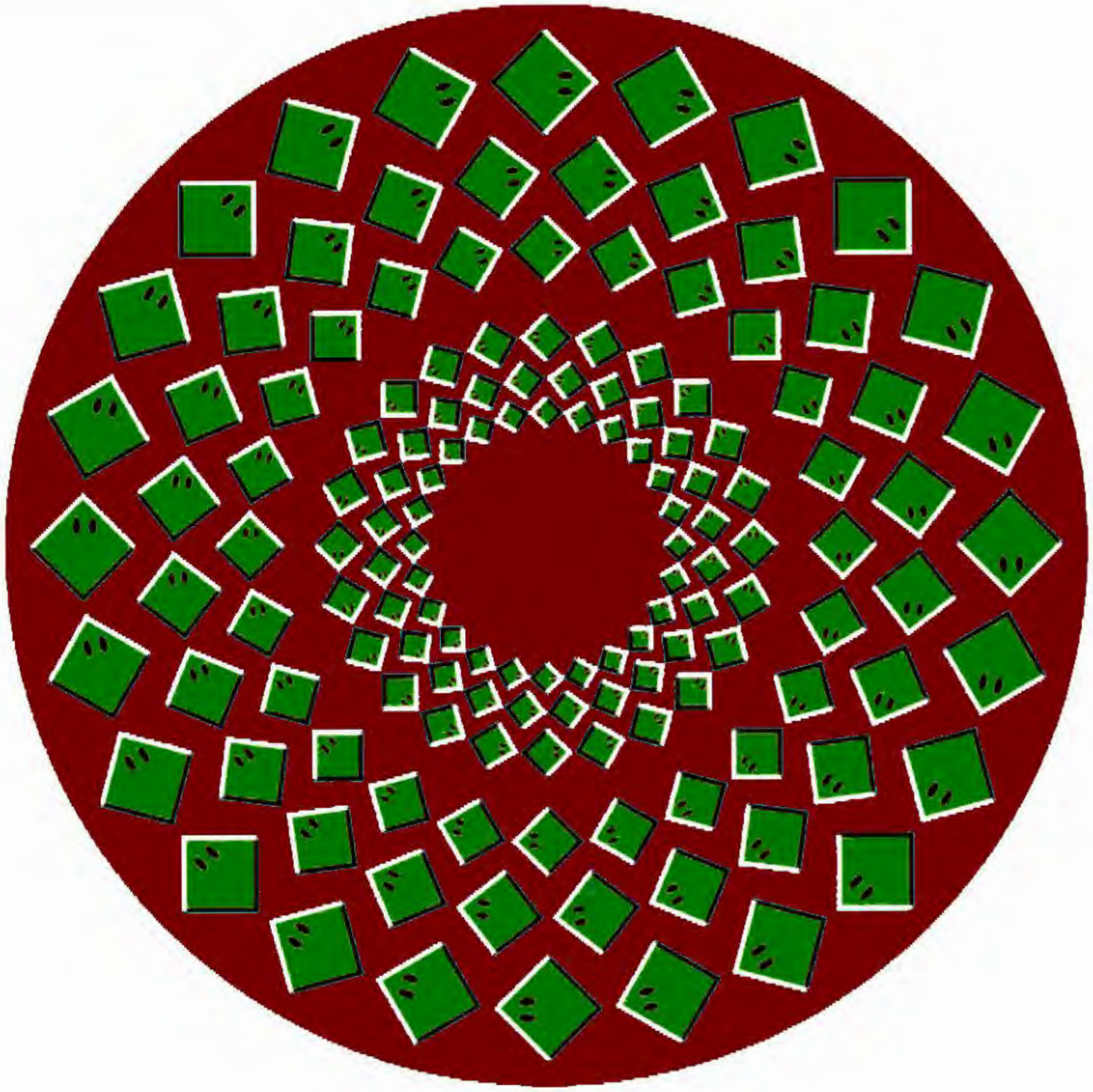


Image Processor

BackGround

50 | 120 | 50

0 | 140 | 0

Enabled

ForeGround

180 | 180 | 180

140 | 0 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

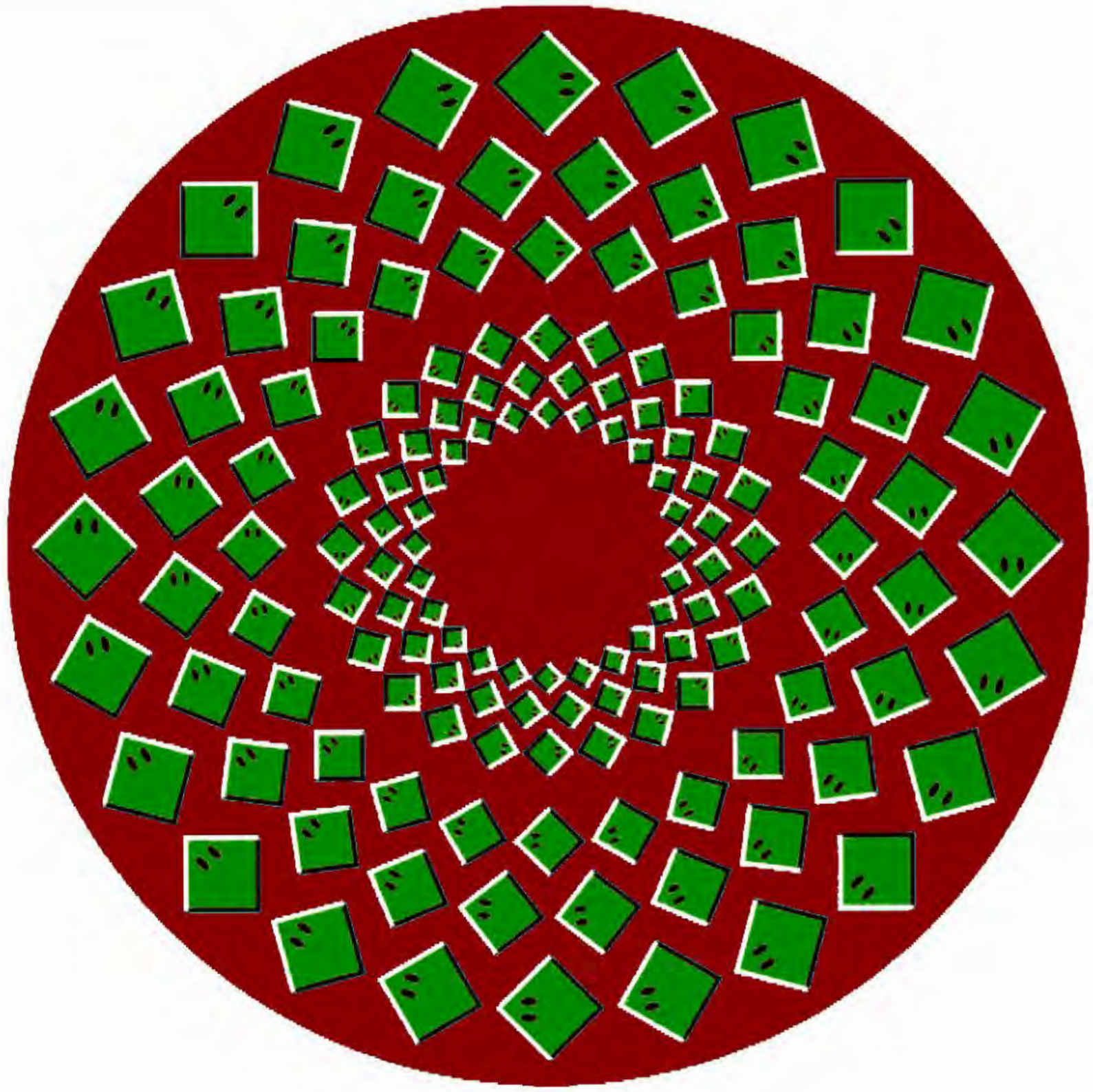


Image Processor

BackGround

50 120 50

0 150 0

Enabled

ForeGround

180 180 180

150 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

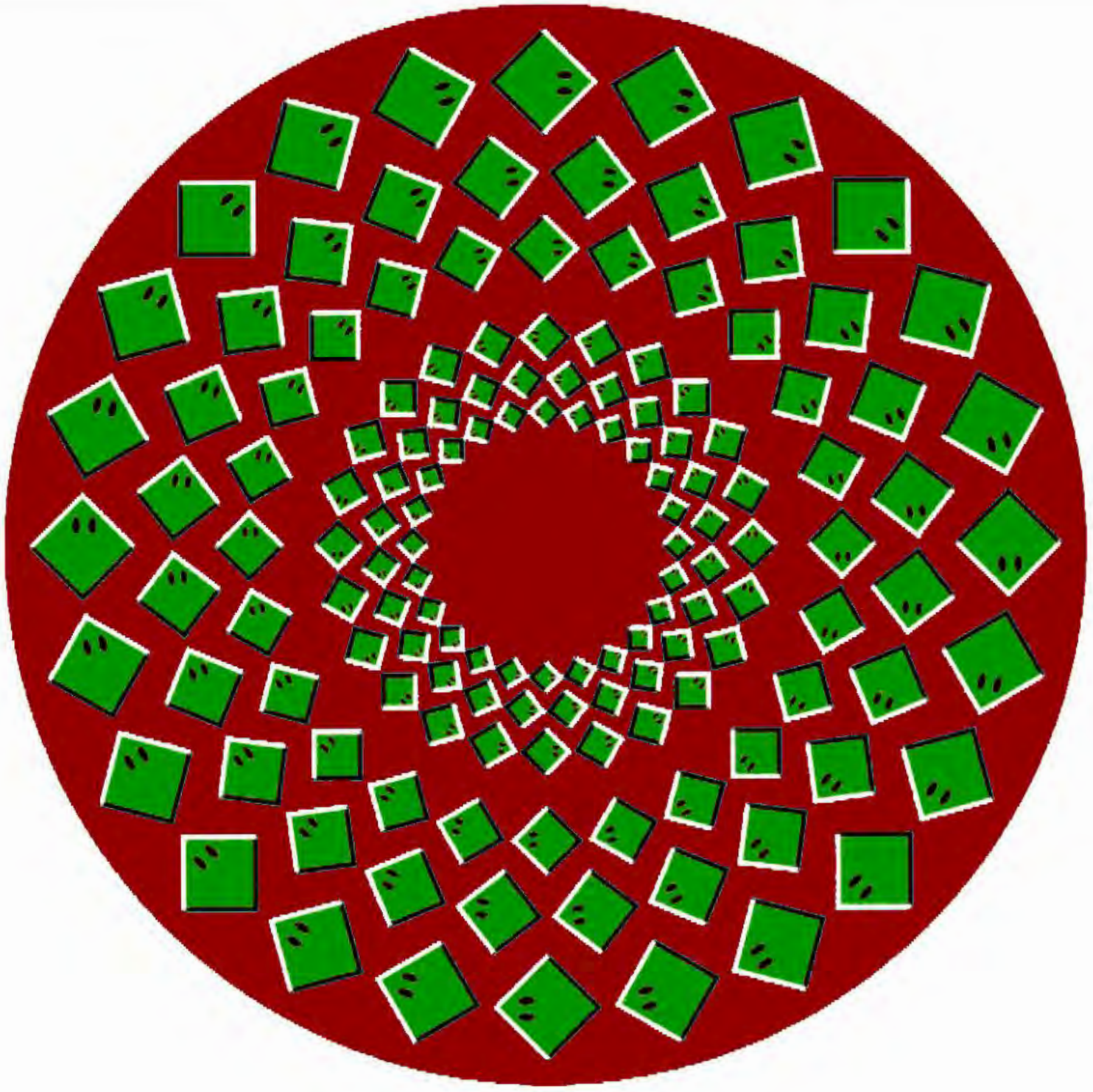


Image Processor

BackGround

50 120 50

0 170 0

Enabled

ForeGround

180 180 180

170 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

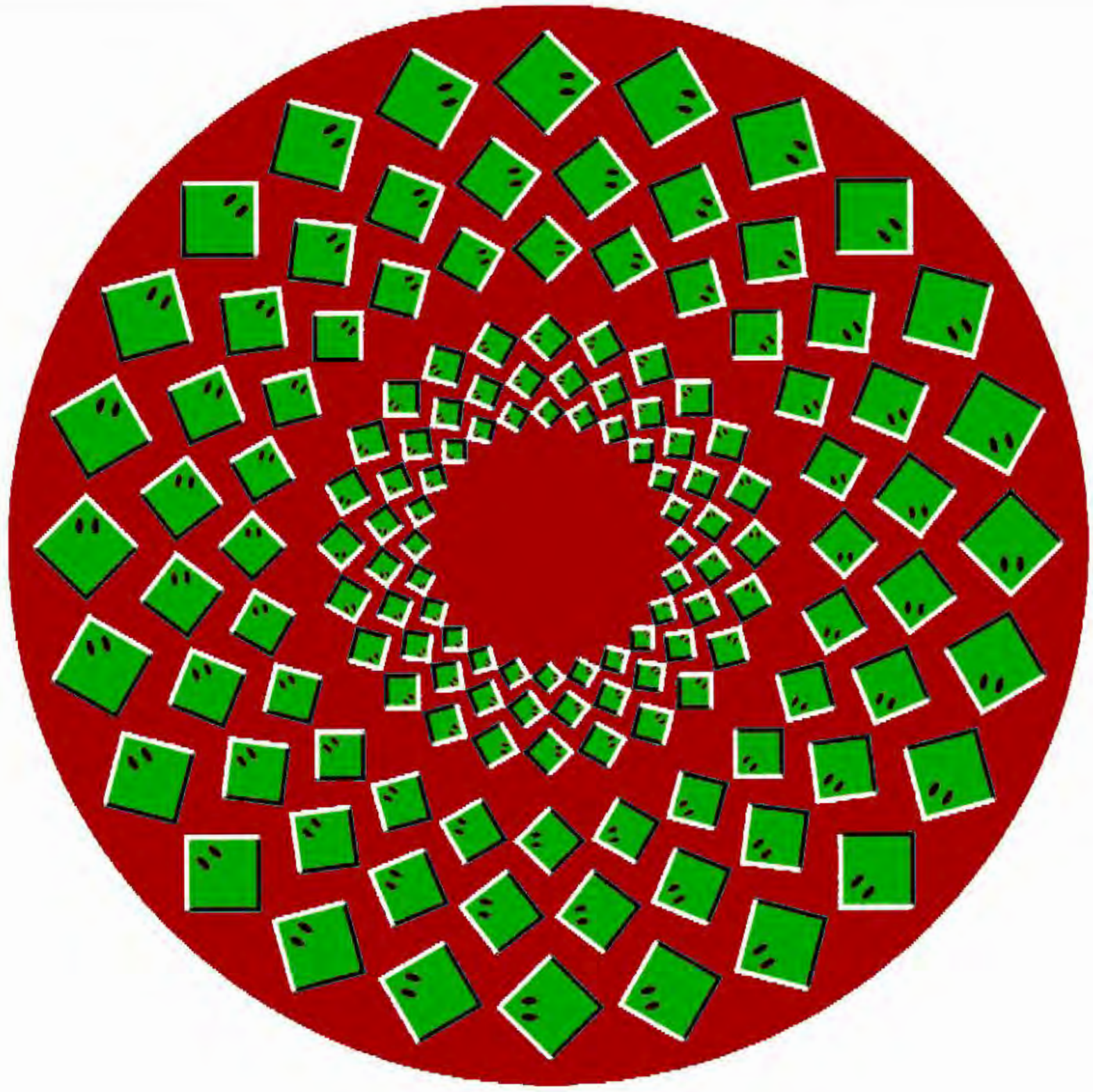


Image Processor

BackGround

50 120 50

0 110 0

Enabled

ForeGround

180 180 180

200 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

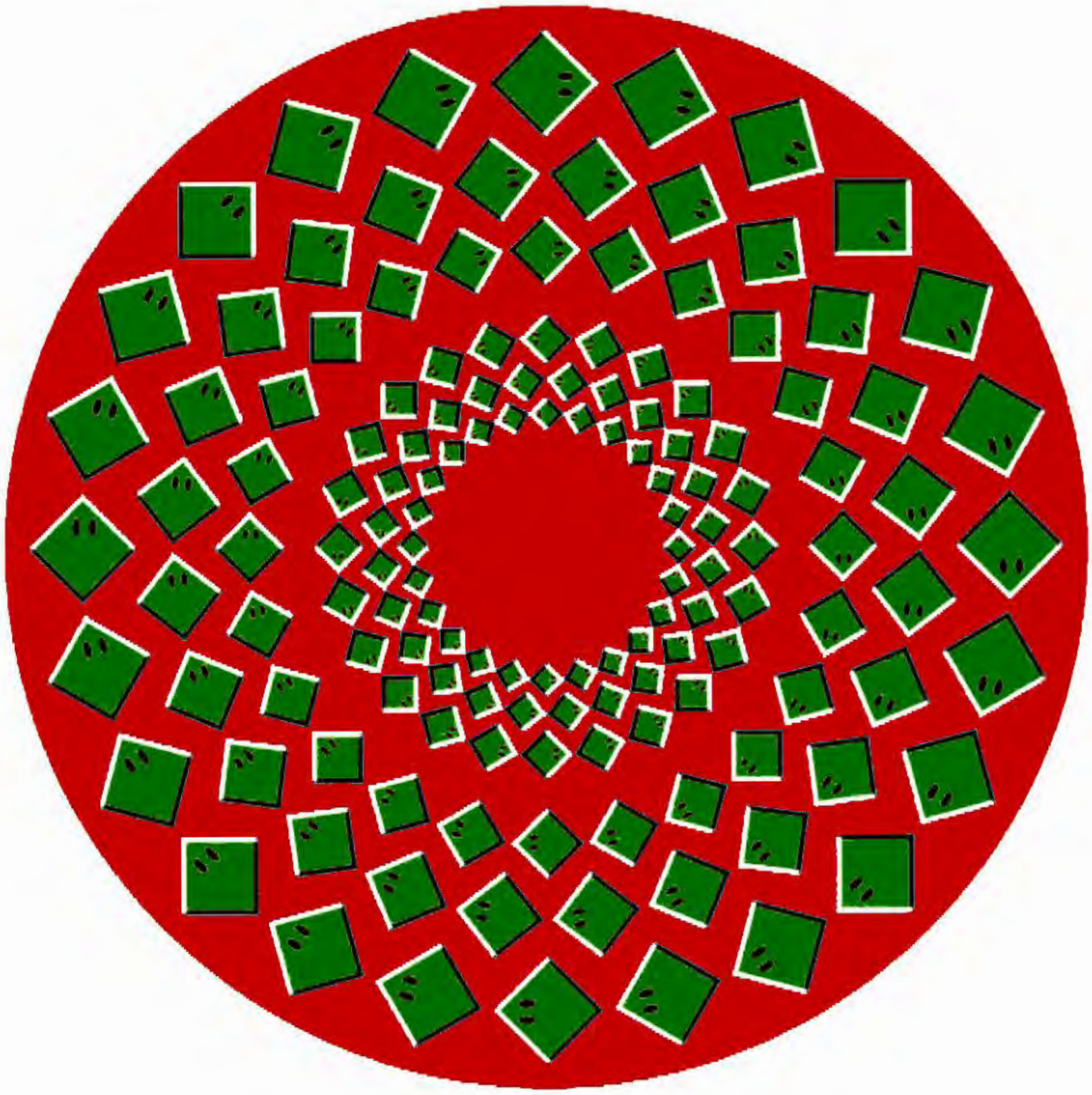


Image Processor

BackGround

50 120 50

0 120 0

Enabled

ForeGround

180 180 180

200 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

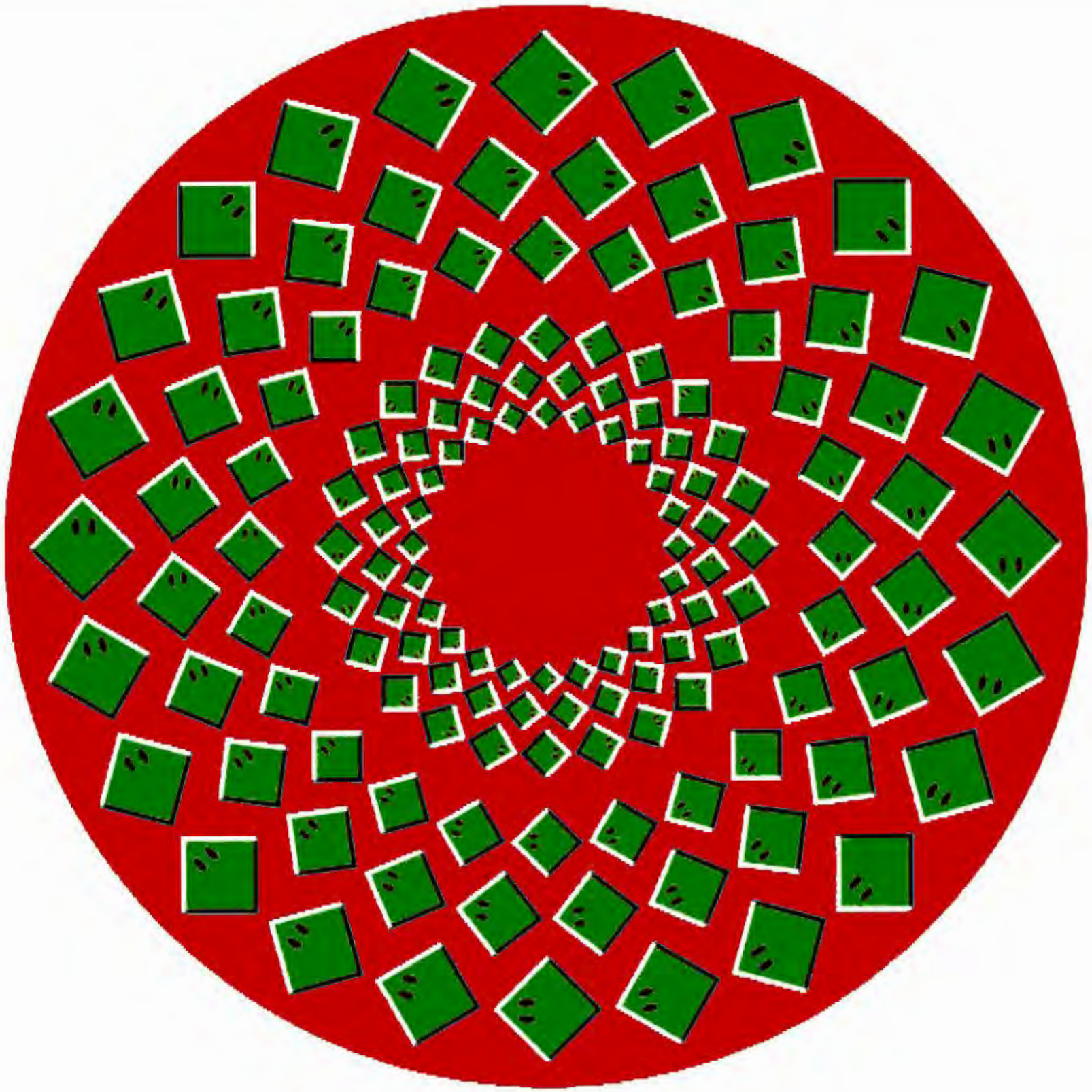


Image Processor

BackGround

50 120 50

0 140 0

Enabled

ForeGround

180 180 180

200 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

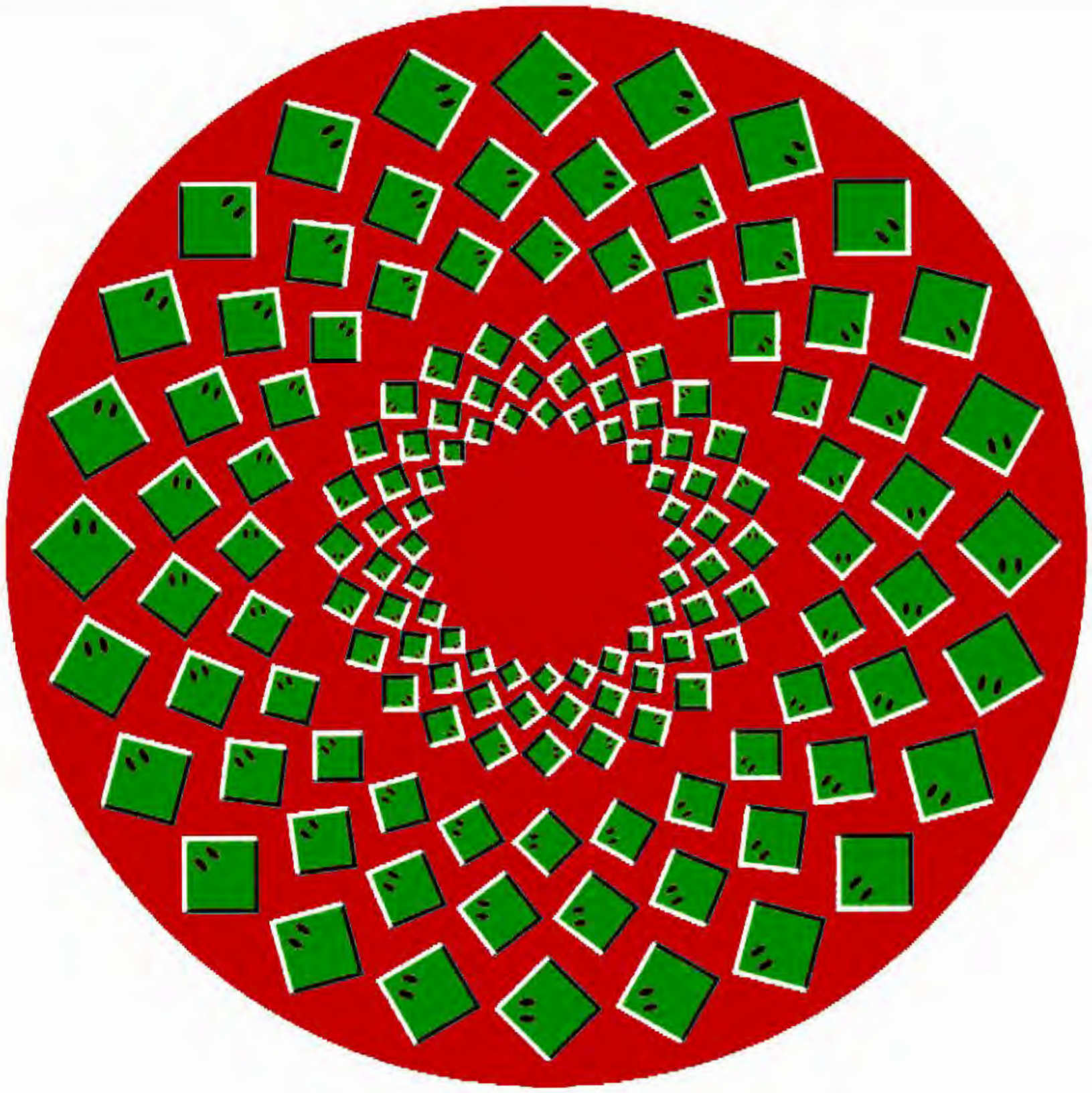


Image Processor

BackGround

50 120 50

0 160 0

Enabled

ForeGround

180 180 180

200 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

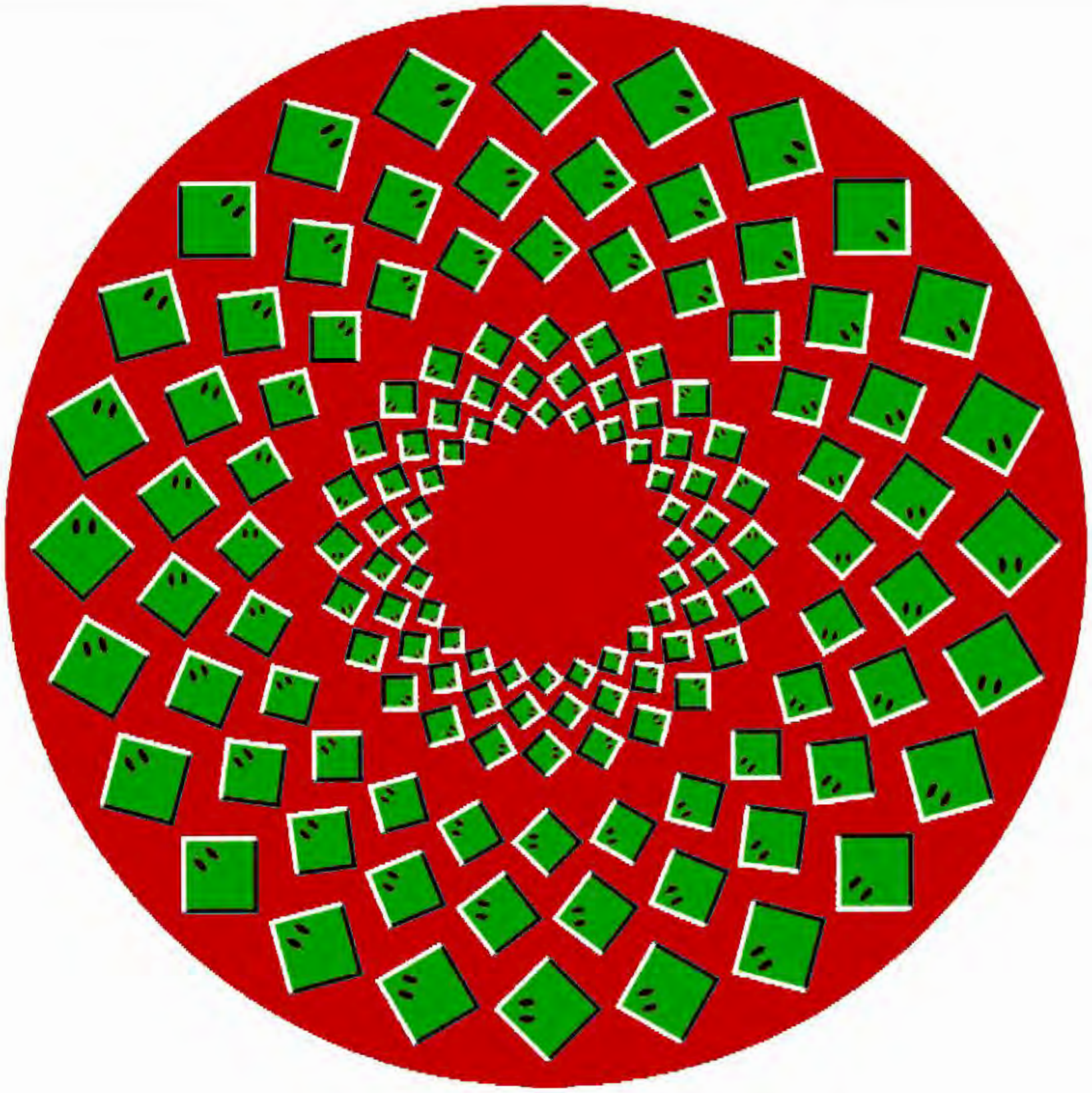


Image Processor

BackGround

50 | 120 | 50

0 | 180 | 0

Enabled

ForeGround

180 | 180 | 180

200 | 0 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW | Rotate CCW

0°

Save Image

Load Image

Reload Image

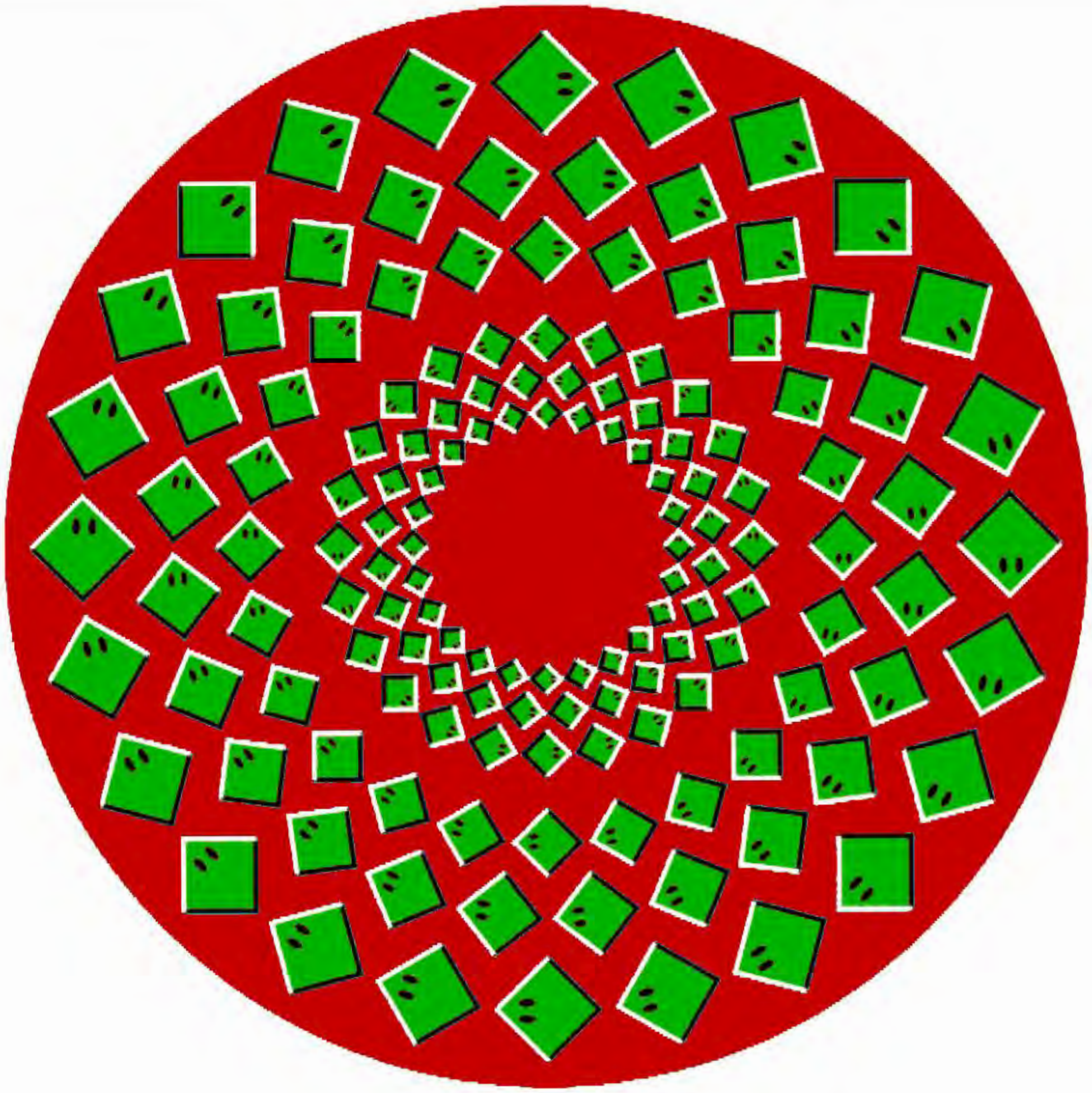


Image Processor

BackGround

50 120 50

0 200 0

Enabled

ForeGround

180 180 180

200 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

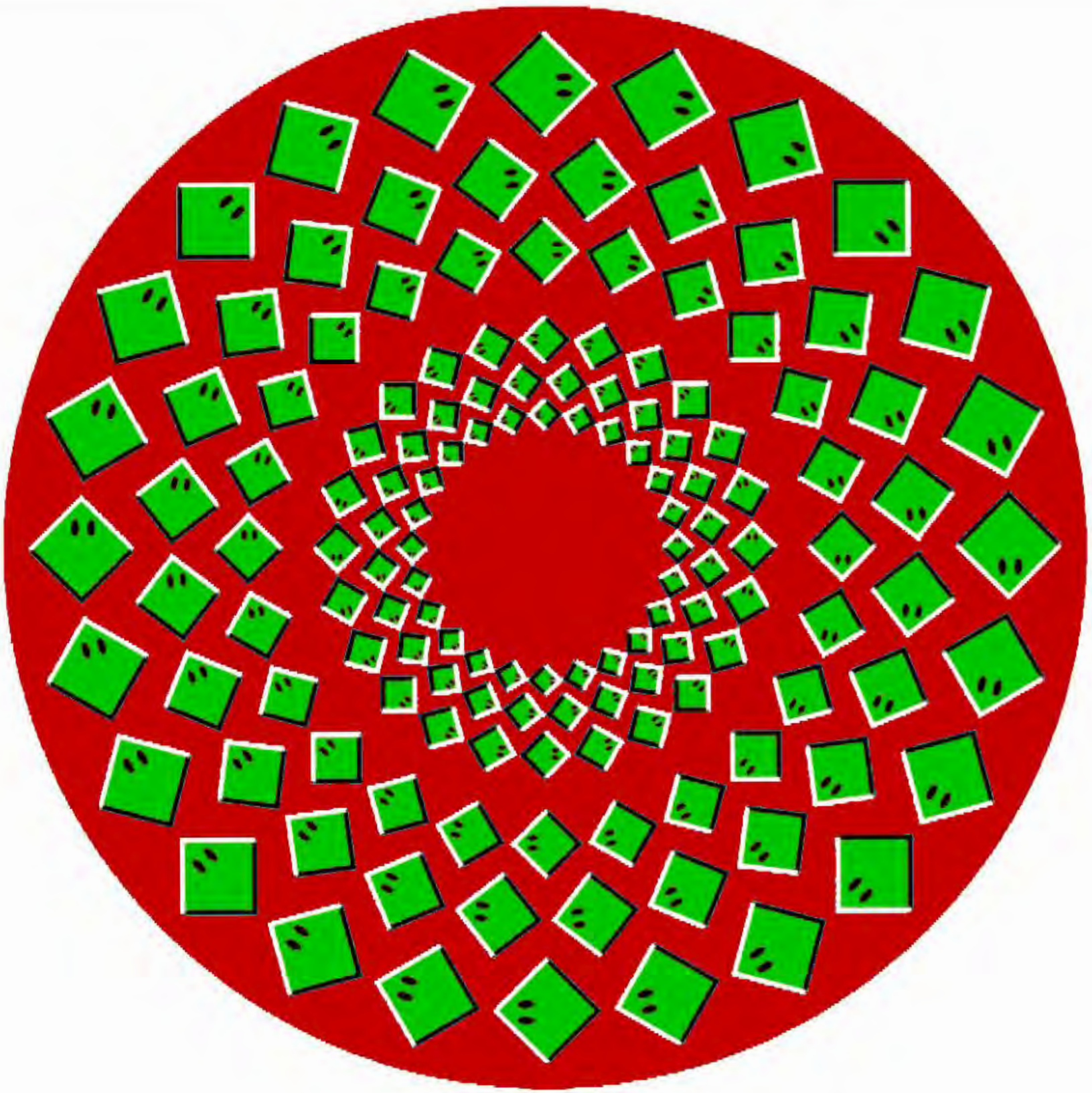


Image Processor

BackGround

50 120 50

0 200 0

Enabled

ForeGround

180 180 180

200 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

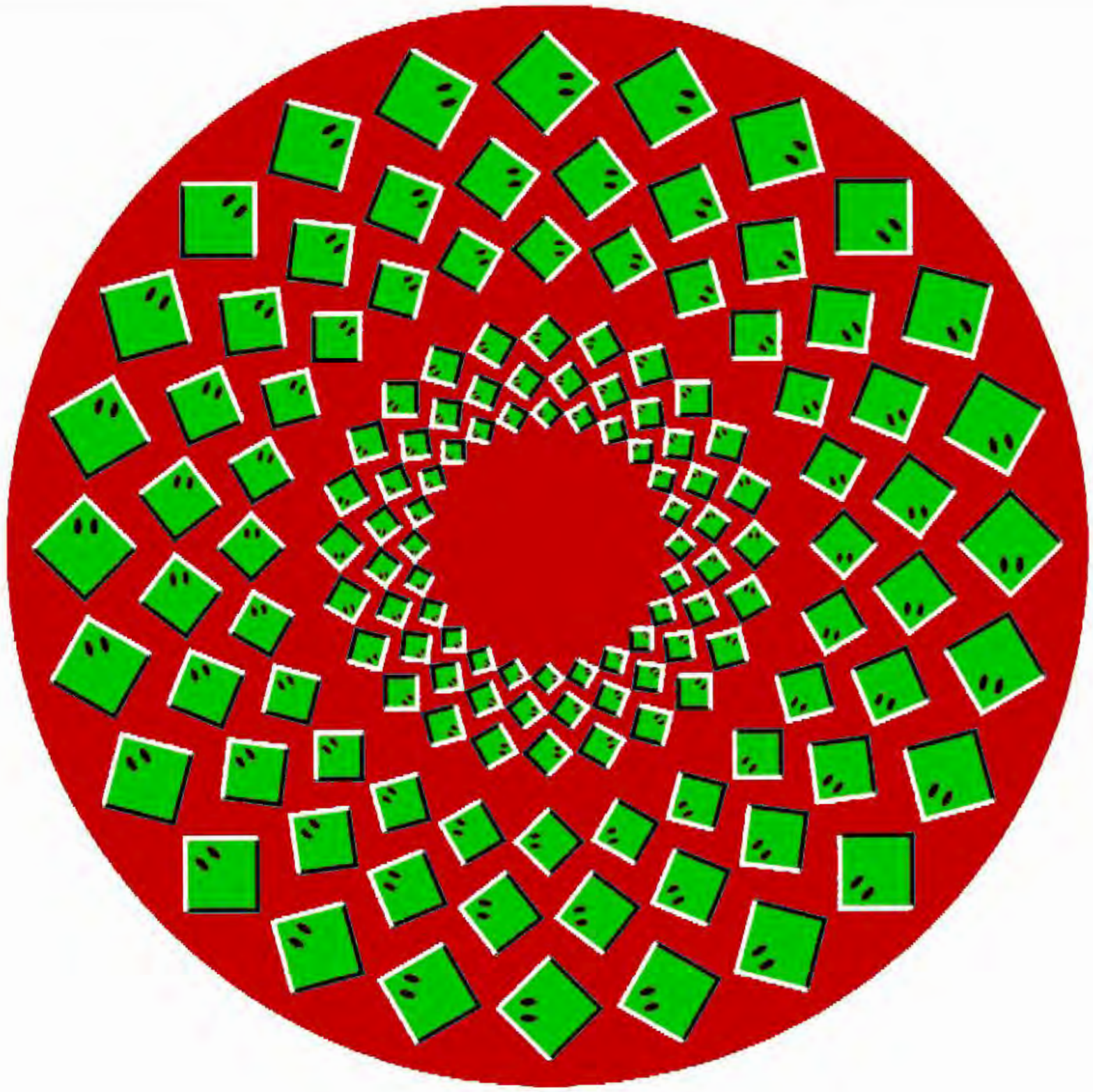


Image Processor

BackGround

50 | 120 | 50

0 | 160 | 0

Enabled

ForeGround

180 | 180 | 180

220 | 0 | 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

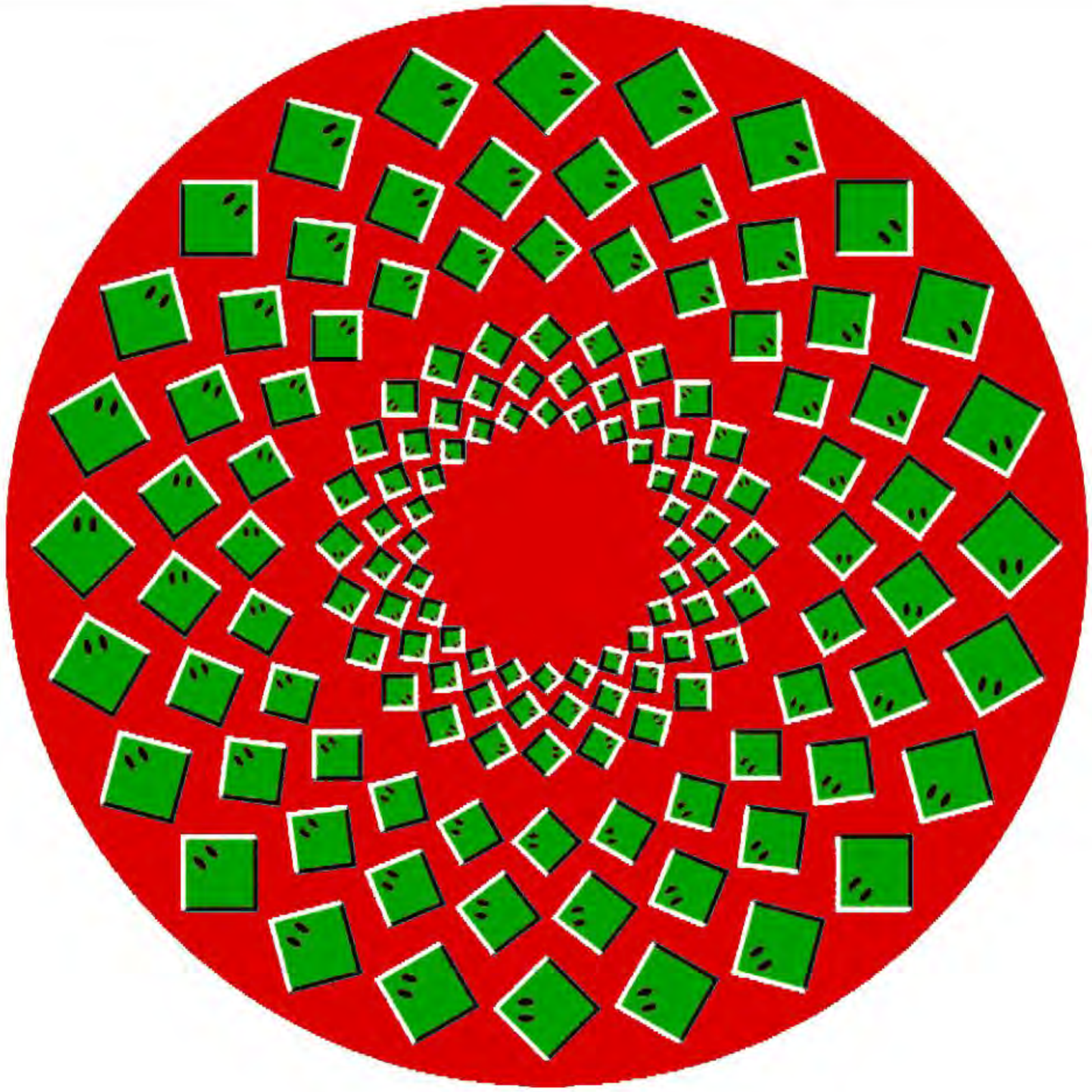


Image Processor

BackGround

50 120 50

0 180 0

Enabled

ForeGround

180 180 180

220 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

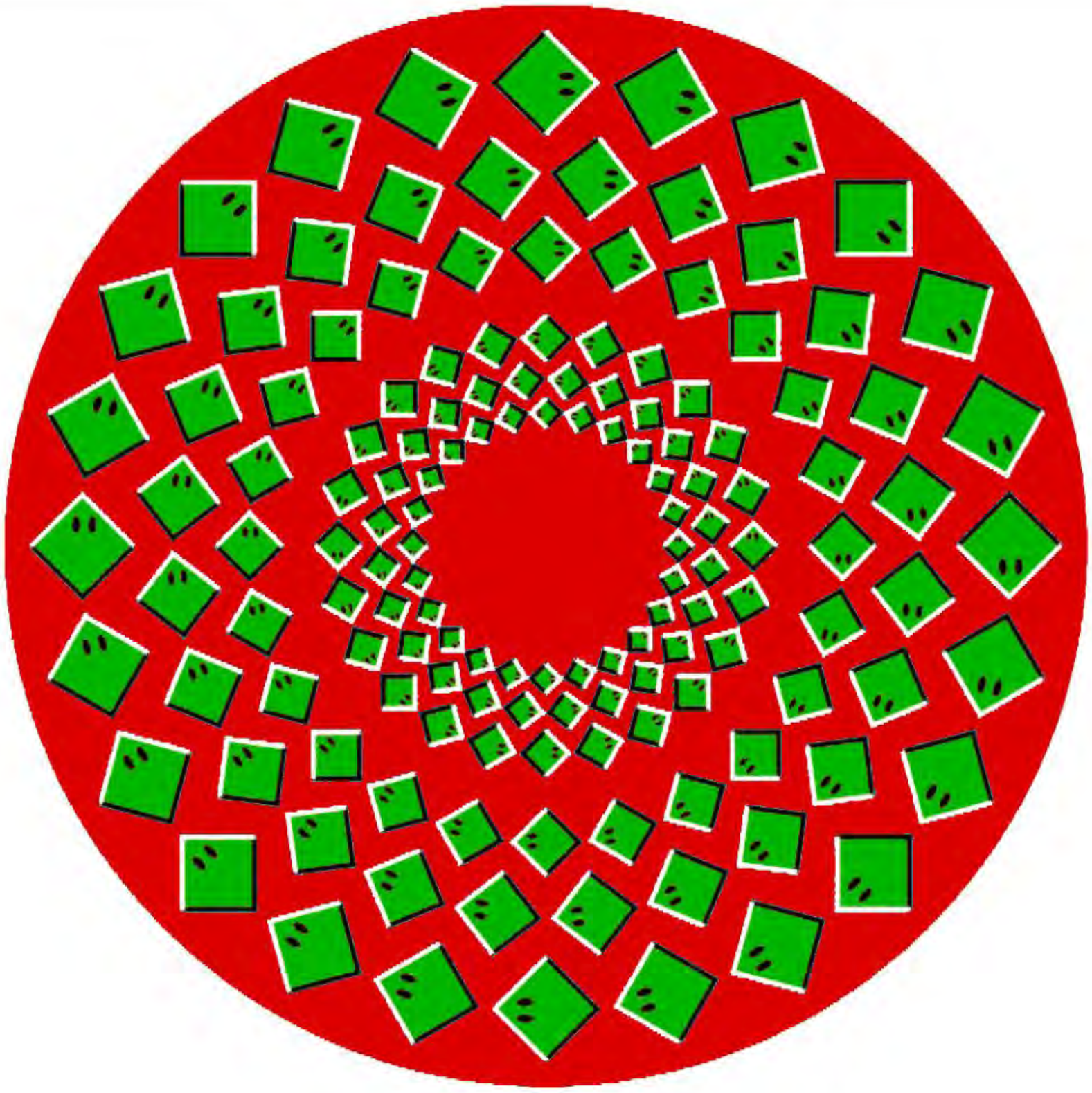


Image Processor

BackGround

50 120 50

0 200 0

Enabled

ForeGround

180 180 180

220 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

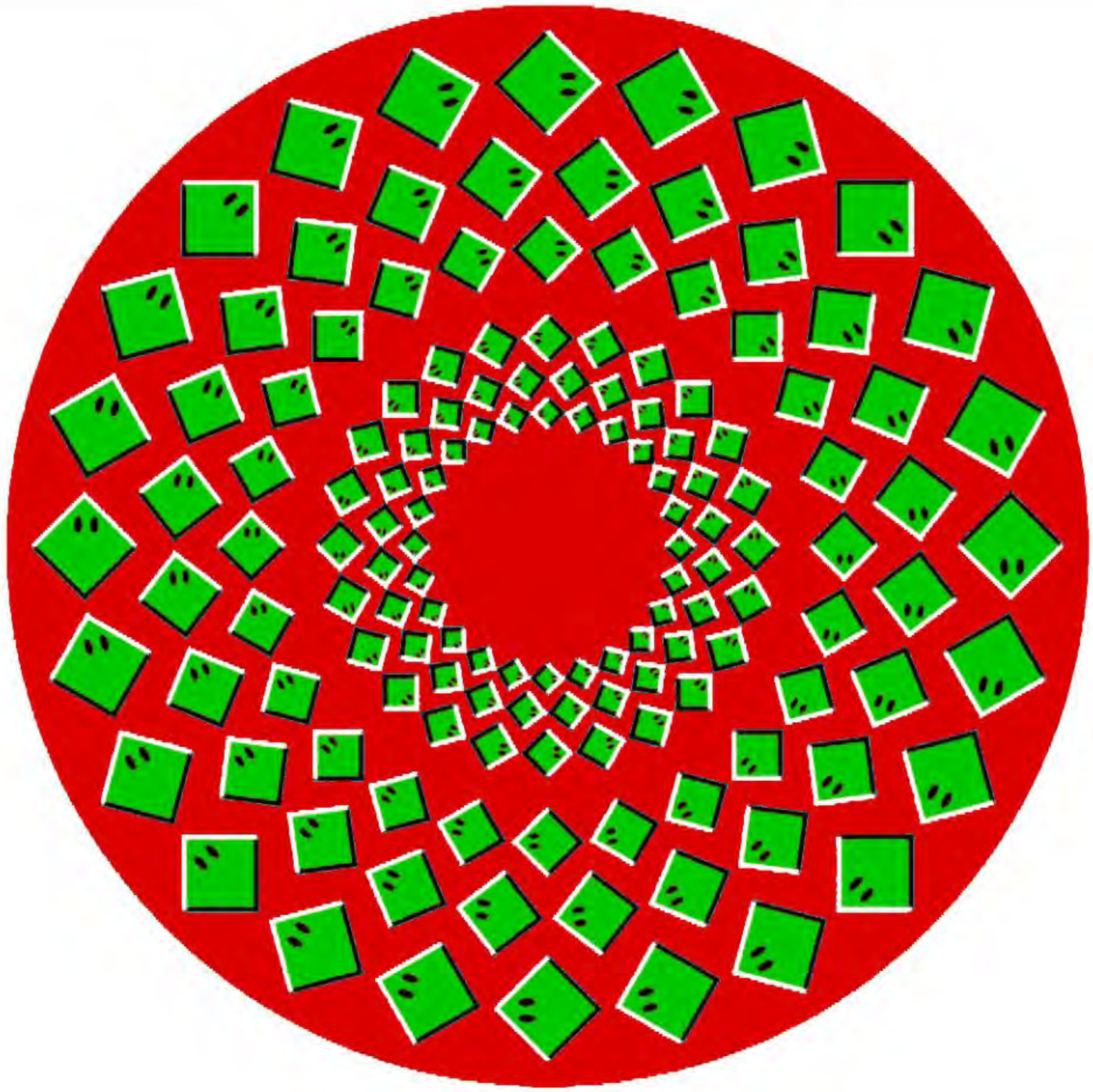


Image Processor

BackGround

50 120 50

0 220 0

Enabled

ForeGround

180 180 180

220 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

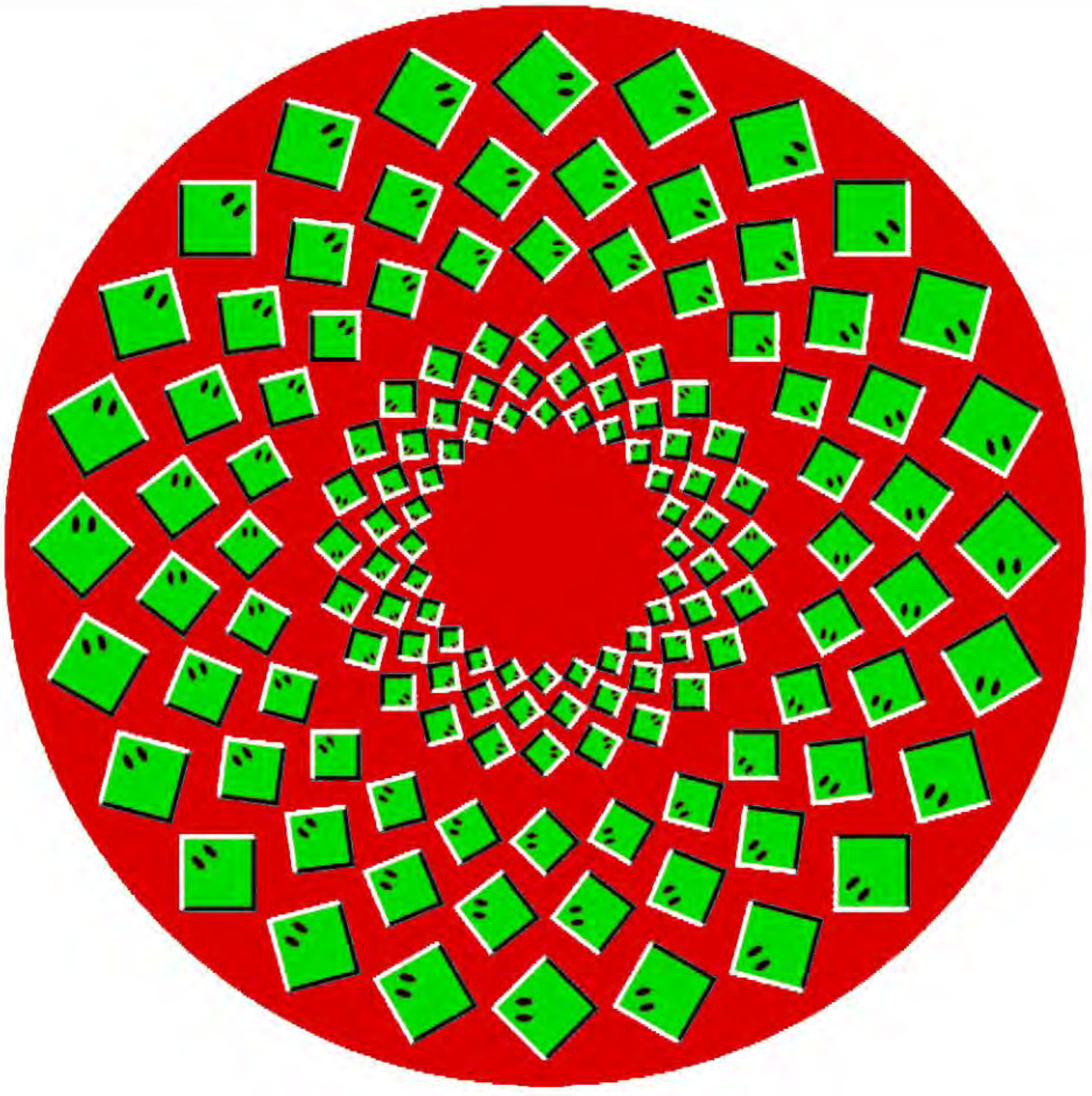


Image Processor

BackGround

50 120 50

0 240 0

Enabled

ForeGround

180 180 180

220 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

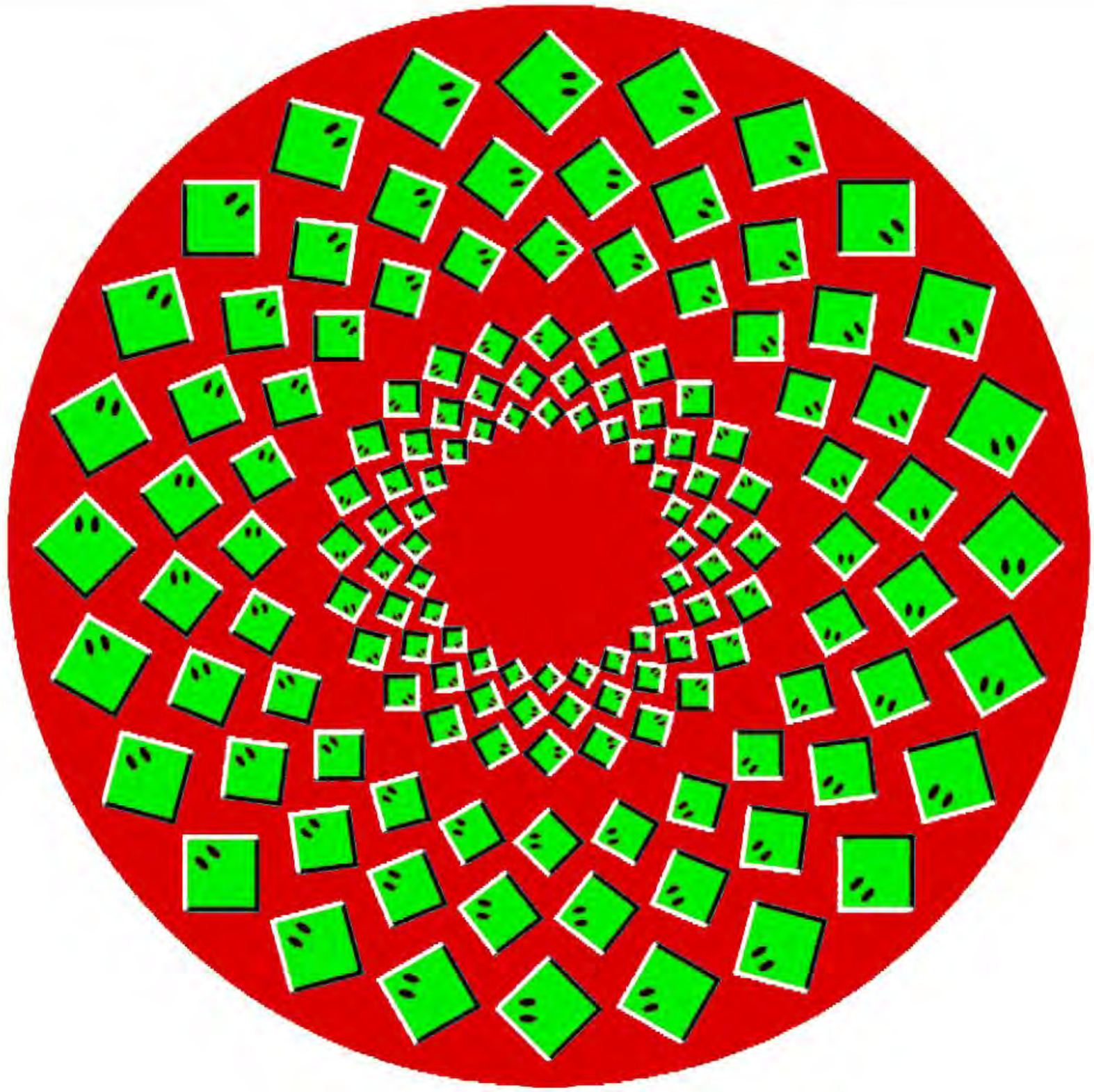


Image Processor

BackGround

50 120 50

0 230 0

Enabled

ForeGround

180 180 180

230 0 0

Enabled

Expander1

Apply

Reload and Apply

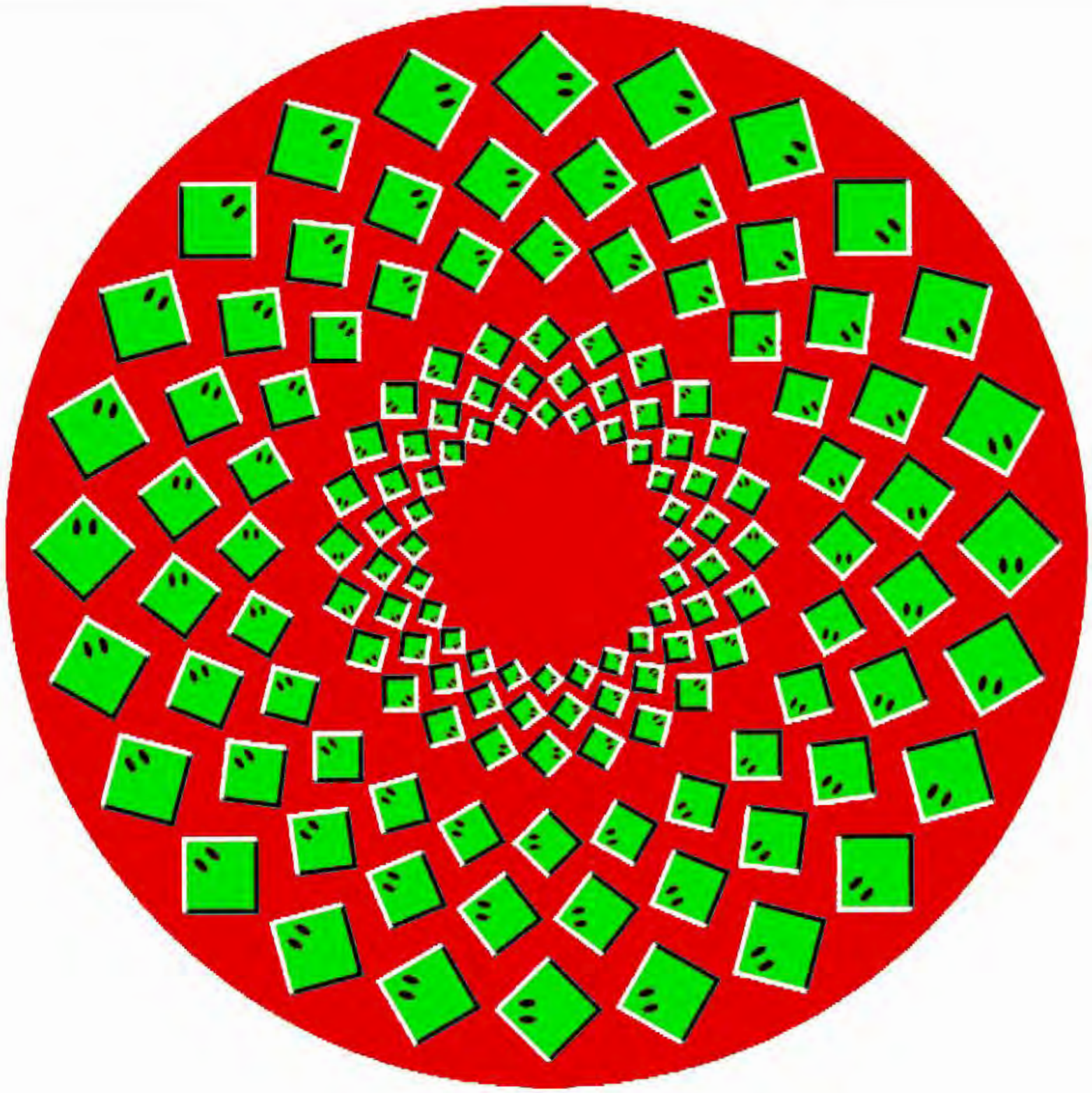
Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image



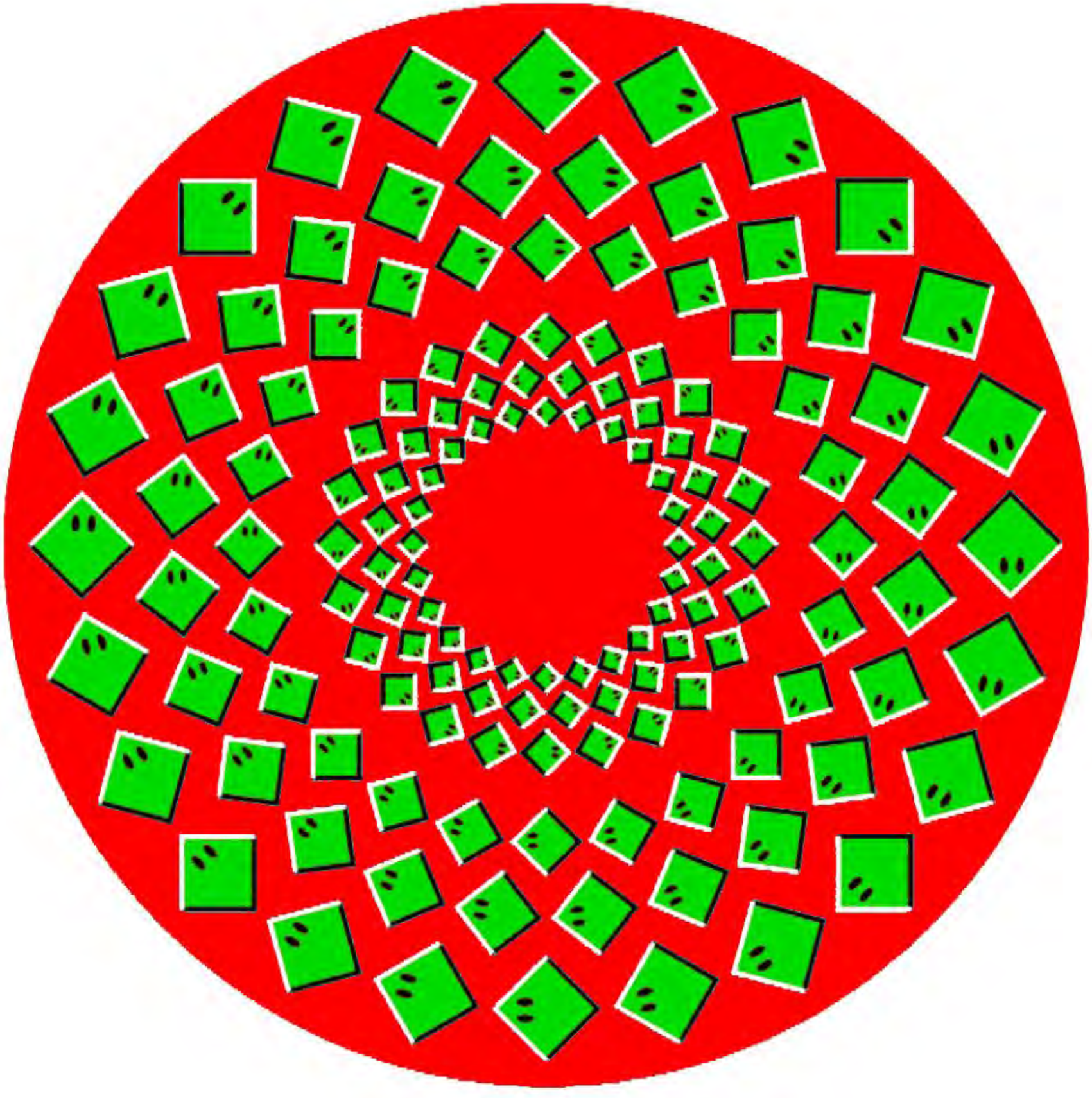


Image Processor

BackGround

50 120 50

0 140 0

Enabled

ForeGround

180 180 180

220 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

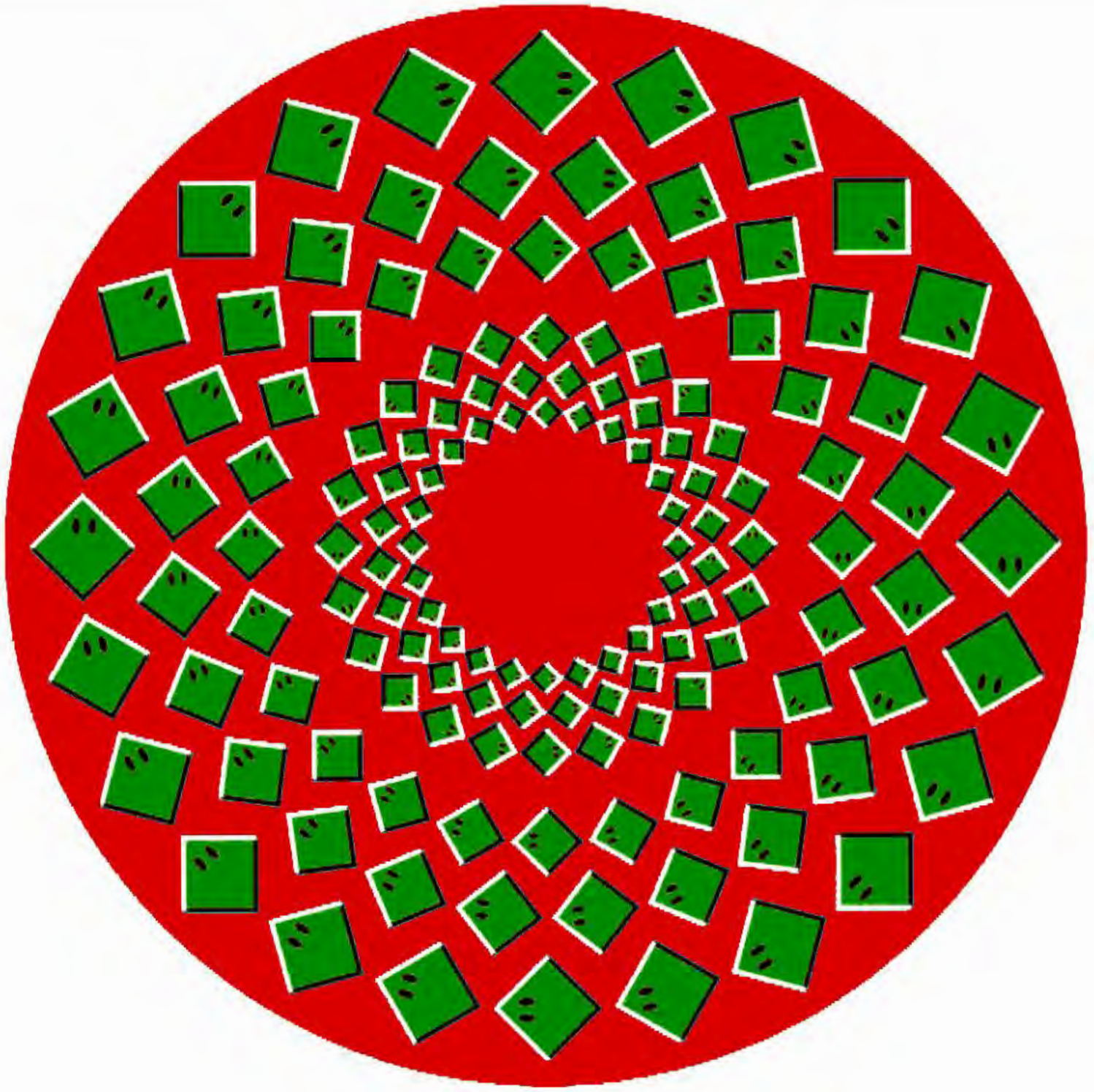


Image Processor

BackGround

50 120 50

0 150 0

Enabled

ForeGround

180 180 180

150 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

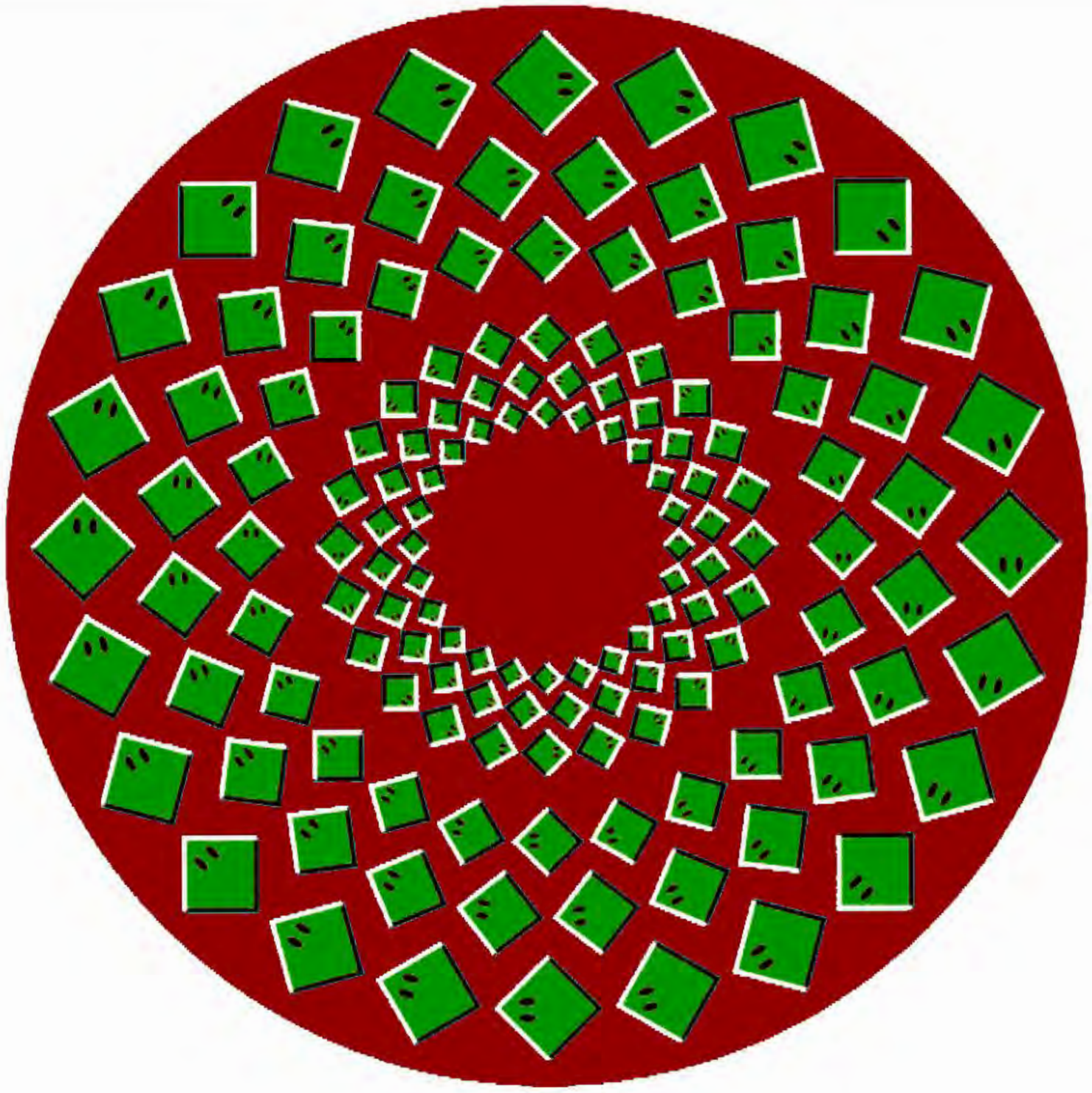


Image Processor

BackGround

50 120 50

0 120 0

Enabled

ForeGround

180 180 180

220 0 0

Enabled

Expander1

Apply

Reload and Apply

Rotate CW Rotate CCW

0°

Save Image

Load Image

Reload Image

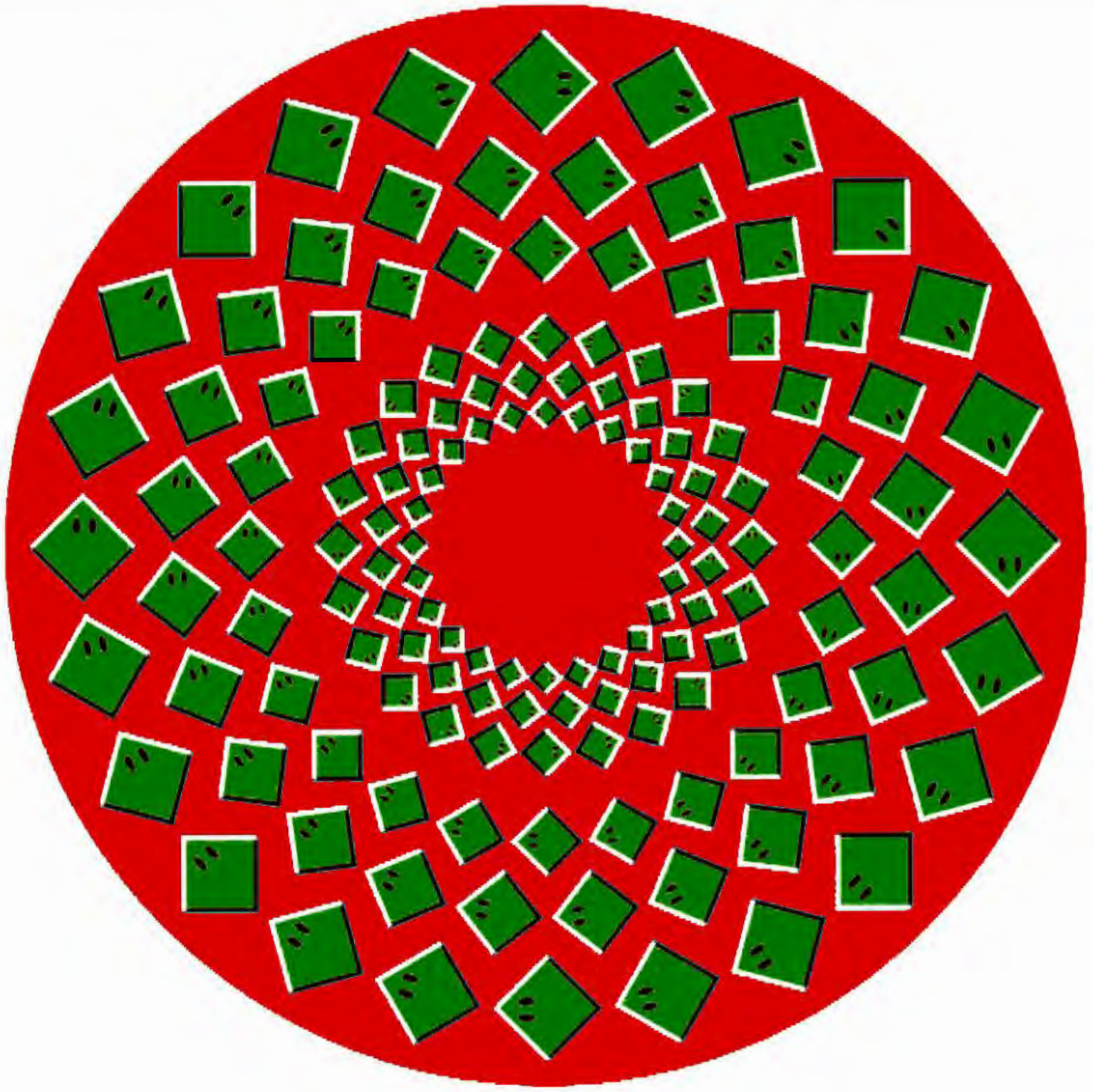


Image Processor

BackGround

50 | 120 | 50

0 | 100 | 0

Enabled

ForeGround

180 | 180 | 180

220 | 0 | 0

Enabled

Expander1

Apply

Reload and Apply

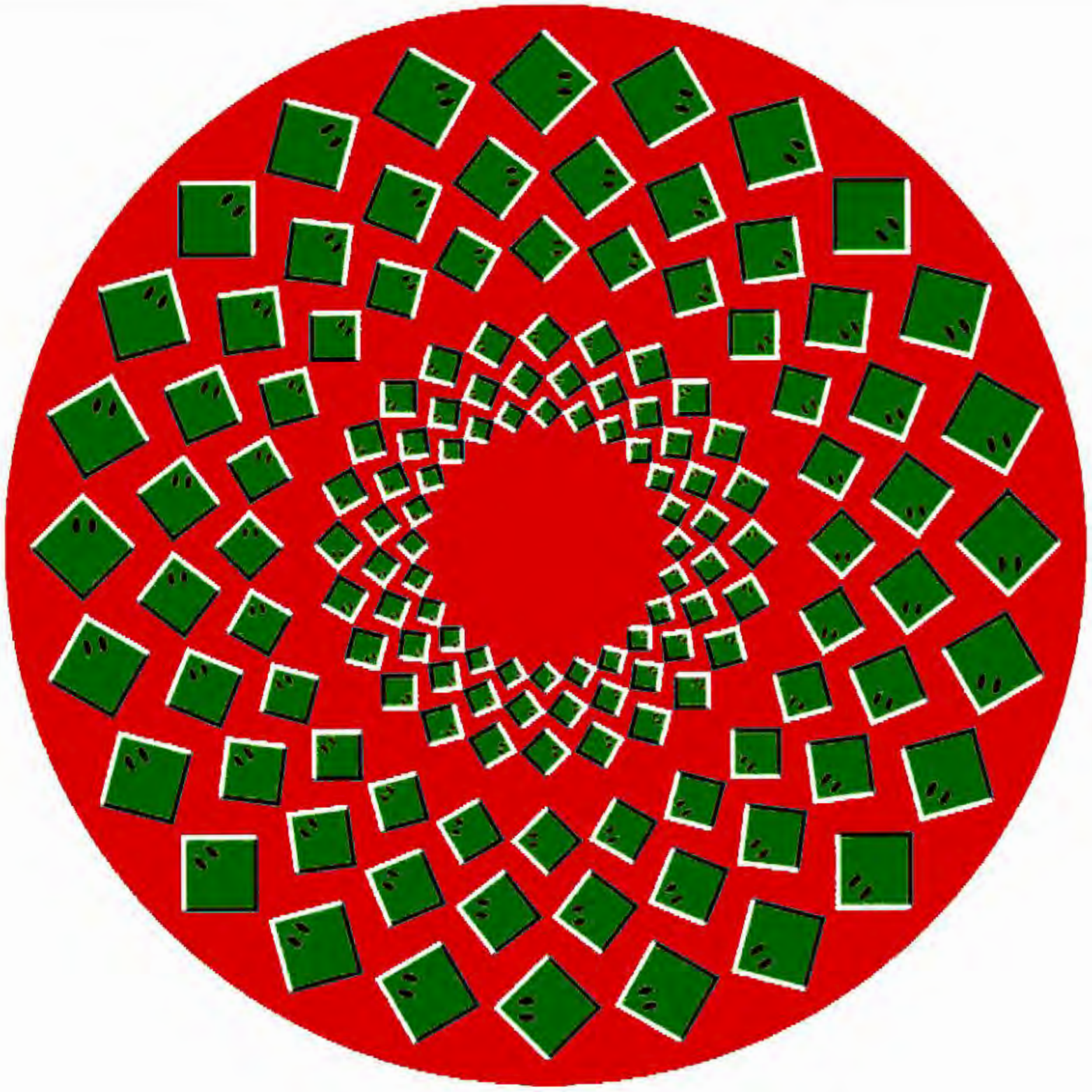
Rotate CW Rotate CCW

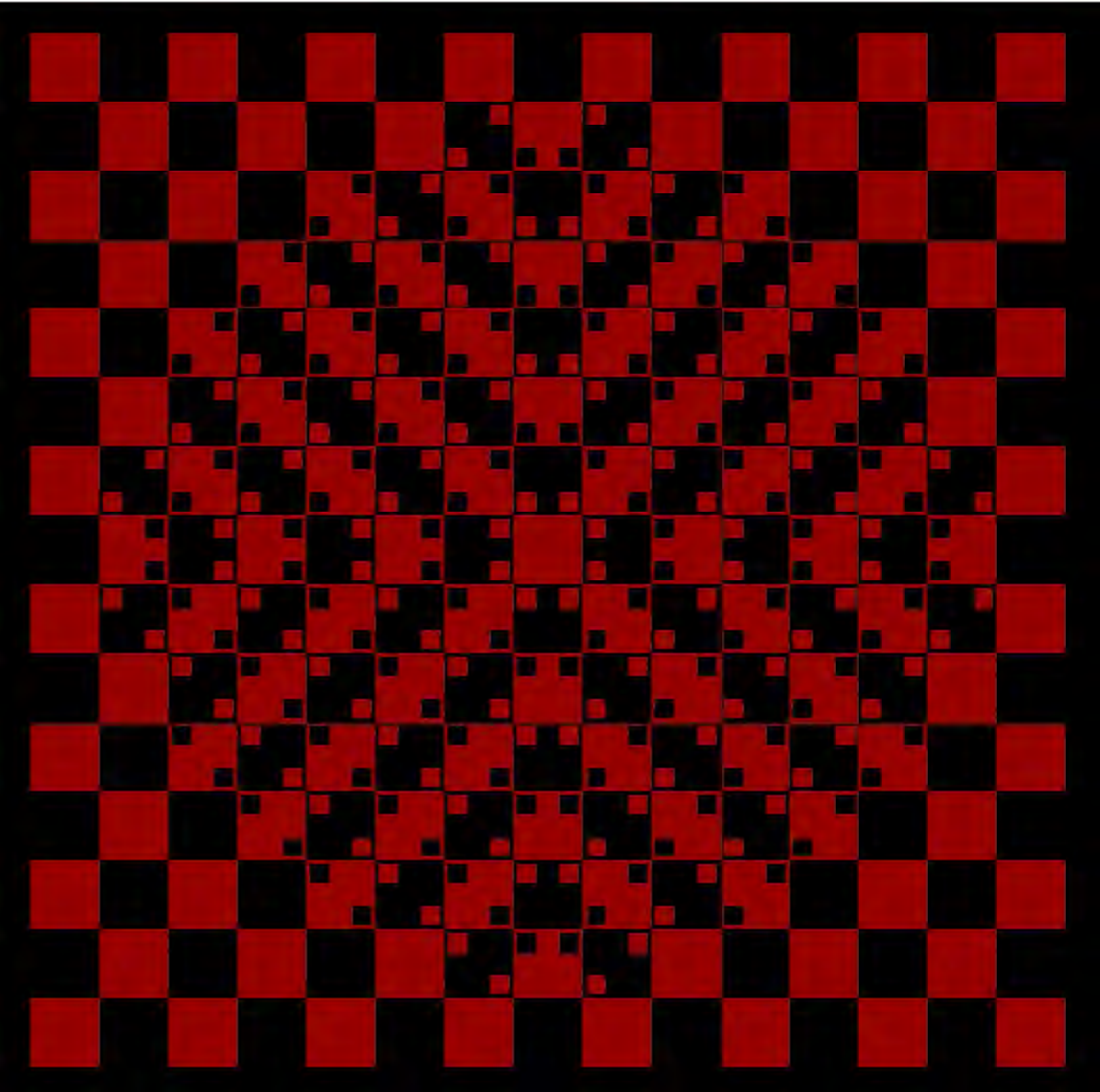
0°

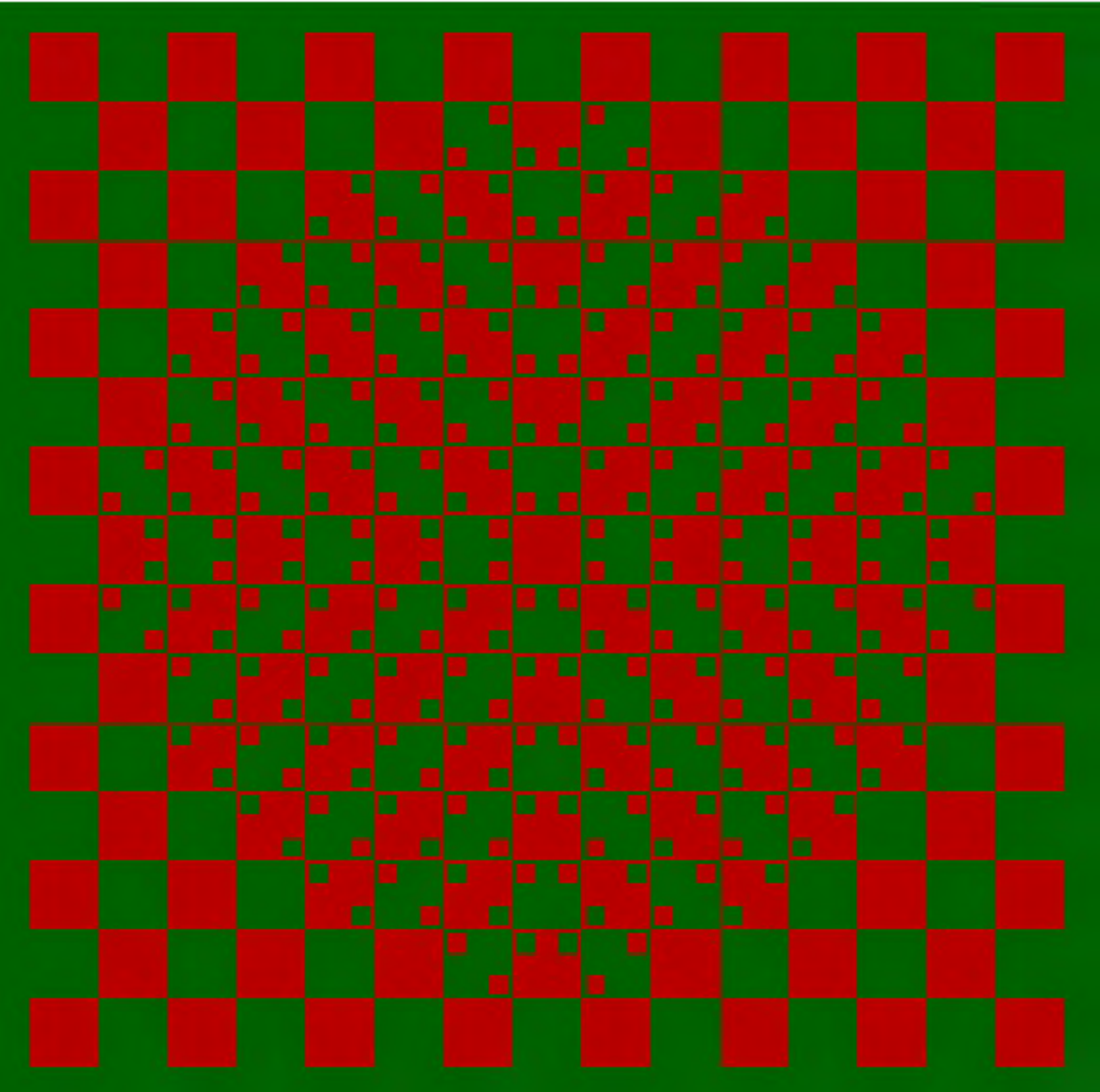
Save Image

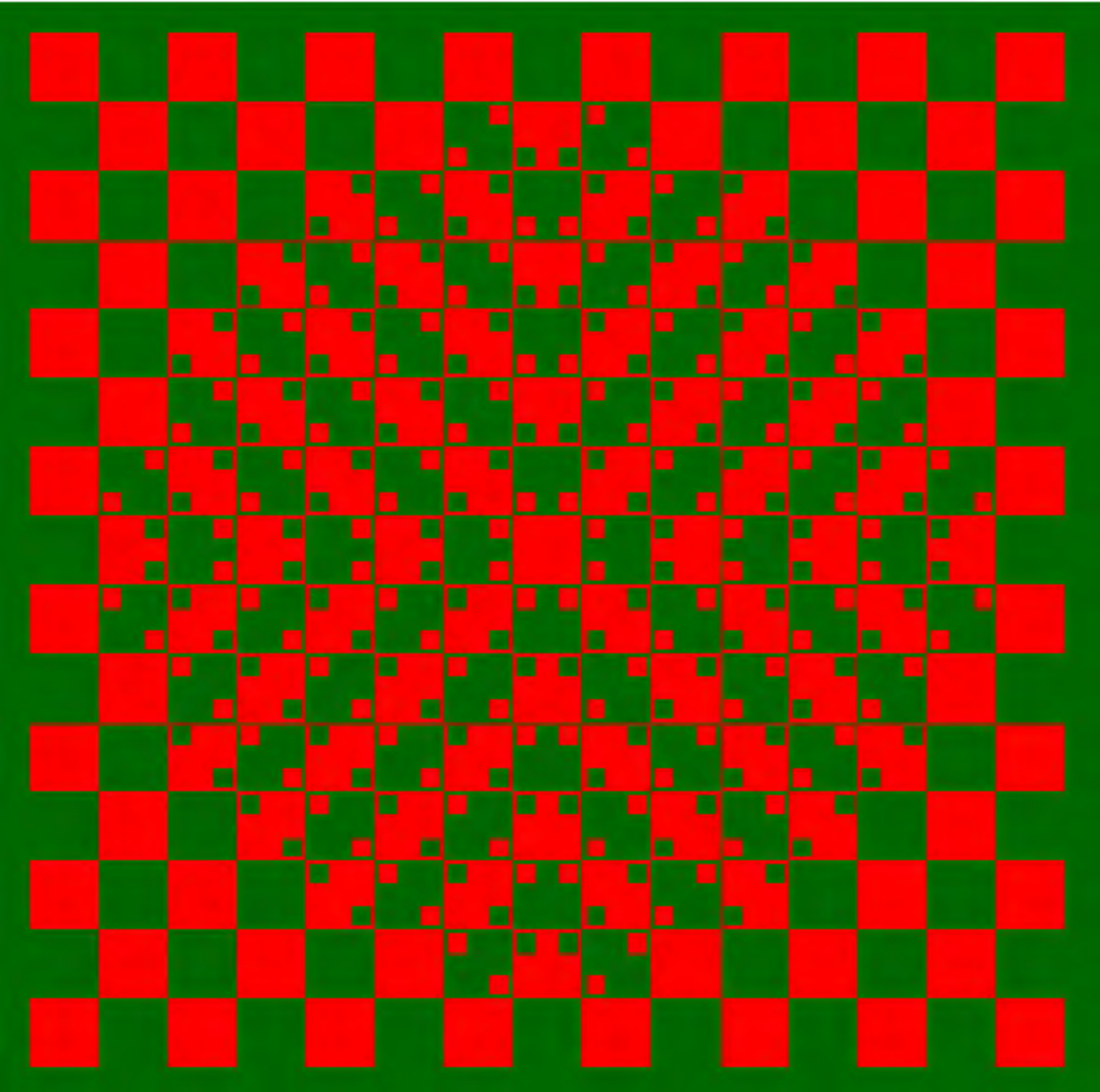
Load Image

Reload Image









```
unit ReportUnit;


interface
Var
  Joinarray40 : Array[1..40] OF STRING[13];
  Typearray40 : Array[1..40] OF STRING[1];
  Nowtime,Nowdate: String;
  Whatis,Race,Gender : String;

Procedure Report(Flag : Integer);
Procedure ReportBlocks(B1,B2,B3,B4,B5 : Boolean; Flag : Integer);
Procedure WriteIndDBFile;

implementation

uses ArrayFunctions, FileFunctions, SysUtils,StrUtils,IATWinFormUnit, Forms;//, System.IO;

Procedure Block1Report(Flag : Integer);
Begin
  Nowdate:=DatetoStr(Date);
  Nowtime:=TimetoStr(Time);
  Inc(GlobalCounter);
  IndScores[GlobalCounter].ID:=NameCode;
  IndScores[GlobalCounter].Block:=1;
  IndScores[GlobalCounter].Date:=Nowdate;
  IndScores[GlobalCounter].Time:=NowTime;
  IndScores[GlobalCounter].Gender:=Gend;
  IndScores[GlobalCounter].Age:=Age;
  IndScores[GlobalCounter].QResponse:=VRating;
  if COR then
    Begin
     if CCyan then IndScores[GlobalCounter].Colour:='P' Else IndScores[GlobalCounter].Colour:='M';
    End;
  if NOT (COR) then IndScores[GlobalCounter].Colour:='N';
  IF B1Subliminal Then
    Begin
      IndScores[GlobalCounter].SubTimer1:=IntToStr(TS2);
      IndScores[GlobalCounter].SubTimer2:=IntToStr(TS3);
    End
      Else
        Begin
          IndScores[GlobalCounter].SubTimer1:='000';
          IndScores[GlobalCounter].SubTimer2:='000';
        End;
  IndScores[GlobalCounter].Counter:=Counter;
  IndScores[GlobalCounter].Item:=Joinarray[Counter];
  IndScores[GlobalCounter].Diff:=Diff;
  IndScores[GlobalCounter].Flag:=IntToStr(Flag);
  IndScores[GlobalCounter].SubItem:=Joinarray[Counter];
  IndScores[GlobalCounter].BlockOrder:=Block1Order;
  Write(B1Out,Nowdate :10,'  ');
  Write(B1Out,Nowtime: 11,'  ');
```

```
  Write(B1Out,Namecode :6,' ');
  Write(B1Out,' 1');
  IF COR THEN
    BEGIN
      IF CCyan Then Write(B1Out,' C') Else Write(B1Out,' R');
    END;
  IF NOT (COR) Then Write(B1Out,' N');
  IF B1Subliminal Then Write(B1Out,' ',TS2: 3,' ',TS3: 3) ELSE
  Write(B1Out,' ','000',' ','000');
  Write(B1Out,Joinarray[Counter]:13,' ');
  Write(B1Out,'..... ');
//  Write(B1Out,Q1to5,' ');
  Write(B1Out,Counter:2,' ');
  Write(B1Out,Diff:6,' ');
  Write(B1Out,Flag,' ');
  Write(B1Out,Joinarray[Counter]:13);
  Write(B1Out,' ',Block1Order);
{  if MaskFlag then Write(B1Out,'        ','1')
  Else Write(B1Out,'        ','0');           }
//  Write(B1Out,'  ',SetupType);
  Writeln(B1Out);
End;

Procedure WhatRace;
Var
SWhat : String;
Begin
  SWhat:=LeftStr(Whatis,2);
  if SWhat='MB' Then Whatis:='M   B';
  if SWhat='FB' Then Whatis:='F   B';
  if Swhat='MW' then Whatis:='M   W';
  if SWhat='FW' then Whatis:='F   W';
End;

Procedure Block2Report(Flag : Integer);
Begin
  Nowdate:=DatetoStr(Date);
  Nowtime:=TimetoStr(Time);
  Inc(GlobalCounter);
  IndScores[GlobalCounter].ID:=NameCode;
  IndScores[GlobalCounter].Block:=2;
  IndScores[GlobalCounter].Date:=Nowdate;
  IndScores[GlobalCounter].Time:=NowTime;
  IndScores[GlobalCounter].Gender:=Gend;
  IndScores[GlobalCounter].Age:=Age;
  IndScores[GlobalCounter].QResponse:=VRating;
  if COR then
    Begin
     if CCyan then IndScores[GlobalCounter].Colour:='P' Else IndScores[GlobalCounter].Colour:='M';
    End;
  if NOT (COR) then IndScores[GlobalCounter].Colour:='N';
  IF B2Subliminal Then
    Begin
      IndScores[GlobalCounter].SubTimer1:=IntToStr(TS2);
```

```
      IndScores[GlobalCounter].SubTimer2:=IntToStr(TS3);
    End
      Else
        Begin
          IndScores[GlobalCounter].SubTimer1:='000';
          IndScores[GlobalCounter].SubTimer2:='000';
        End;
  IndScores[GlobalCounter].Counter:=Counter;
  IndScores[GlobalCounter].Item:=Joinarray[Counter];
  IndScores[GlobalCounter].Diff:=Diff;
  IndScores[GlobalCounter].Flag:=IntToStr(Flag);
  if (B2) AND (B2Subliminal) then IndScores[GlobalCounter].SubItem:=Subjoin[Counter];
  IF (B2) AND Not(B2Subliminal) Then
     IndScores[GlobalCounter].SubItem:=Joinarray[Counter];
  IndScores[GlobalCounter].BlockOrder:=Block2Order;
  Write(B2Out,Nowdate :10,'  ');
  Write(B2Out,Nowtime: 11,'  ');
  Write(B2Out,Namecode :6,' ');
  Write(B2Out,' 2');
  IF COR THEN
    BEGIN
      IF CCyan Then Write(B2Out,' C') Else Write(B2Out,' R');
    END;
  IF NOT (COR) Then Write(B2Out,' N');
  IF B2Subliminal Then Write(B2Out,' ',TS2: 3,' ',TS3: 3) ELSE
  Write(B2Out,' ','000',' ','000');
  Write(B2Out,Joinarray[Counter]:13,' ');
  Write(B2Out,'..... ');
//  Write(B2Out,Q1to5,' ');
  Write(B2Out,Counter:2,' ');
  Write(B2Out,Diff:6,' ');
  Write(B2Out,Flag,' ');
  IF (B2) AND (B2Subliminal) Then
     Write(B2Out,Subjoin[Counter] :13);
  IF (B2) AND Not(B2Subliminal) Then
     Write(B2Out,Joinarray[Counter]: 13);
     Whatis:=JoinArray[Counter];
  Write(B2Out,' ',Block2Order);
{  WhatRace;
  Write(B2Out,' ',Whatis);
  if MaskFlag then Write(B2Out,'  ','1')
   Else Write(B2Out,'  ',0);
  Write(B2Out,'  ',SetupType); }
  Writeln(B2Out);

End;

Procedure Block3Report(Flag : Integer);
Begin
  Nowdate:=DatetoStr(Date);
  Nowtime:=TimetoStr(Time);
  Inc(GlobalCounter);
  IndScores[GlobalCounter].ID:=NameCode;
  IndScores[GlobalCounter].Block:=3;
```

```
  IndScores[GlobalCounter].Date:=Nowdate;
  IndScores[GlobalCounter].Time:=NowTime;
  IndScores[GlobalCounter].Gender:=Gend;
  IndScores[GlobalCounter].Age:=Age;
  IndScores[GlobalCounter].QResponse:=VRating;
  if COR then
    Begin
     if CCyan then IndScores[GlobalCounter].Colour:='P' Else IndScores[GlobalCounter].Colour:='M';
    End;
  if NOT (COR) then IndScores[GlobalCounter].Colour:='N';
  IF B3Subliminal Then
    Begin
      IndScores[GlobalCounter].SubTimer1:=IntToStr(TS2);
      IndScores[GlobalCounter].SubTimer2:=IntToStr(TS3);
    End
      Else
        Begin
          IndScores[GlobalCounter].SubTimer1:='000';
          IndScores[GlobalCounter].SubTimer2:='000';
        End;
  IndScores[GlobalCounter].Counter:=Counter;
  IndScores[GlobalCounter].Item:=Joinarray40[Counter];
  IndScores[GlobalCounter].Diff:=Diff;
  IndScores[GlobalCounter].Flag:=IntToStr(Flag);
  Write(B3Out,Nowdate :10,'  ');
  Write(B3Out,Nowtime: 11,'  ');
  Write(B3Out,Namecode :6,' ');
  Write(B3Out,' 3');
  IF COR THEN
    BEGIN
      IF CCyan Then Write(B3Out,' C') Else Write(B3Out,' R');
    END;
  IF NOT (COR) Then Write(B3Out,' N');
  IF B3Subliminal Then Write(B3Out,' ',GS2: 3,' ',TS3: 3) ELSE
  Write(B3Out,' ','000',' ','000');
  Write(B3Out,Joinarray40[Counter]:13,' ');
  Write(B3Out,'..... ');
//  Write(B3Out,Q1to5,' ');
  Write(B3Out,Counter:2,' ');
  Write(B3Out,Diff:6,' ');
  Write(B3Out,Flag,' ');
  IF B3Subliminal THEN
      BEGIN
        IF (IDArray40[Counter]= 'FW') OR (IDArray40[Counter]='MW') THEN
        Begin
          Write(B3Out,JoinArray40[Counter]:13);
          Whatis:=JoinArray40[Counter];
          WhatRace;
          Write(B3Out,' ',Block3Order);
          Write(B3Out,' ',Whatis);
          IndScores[GlobalCounter].SubItem:=(JoinArray40[Counter]+' '+Whatis);
          IndScores[GlobalCounter].BlockOrder:=Block3Order;
        End;
        IF (IDArray40[Counter]='FB') Or (IDArray40[Counter]='MB') THEN
```

```
      Begin
       Write(B3Out,JoinArray40[Counter]:13);
       Whatis:=JoinArray40[Counter];
       WhatRace;
       Write(B3Out,' ',Block3Order);
       Write(B3Out,' ',Whatis);
       IndScores[GlobalCounter].SubItem:=(JoinArray40[Counter]+' '+Whatis);
       IndScores[GlobalCounter].BlockOrder:=Block3Order;
      End;
      IF (IDArray40[Counter]='P') THEN
       Begin
        Write(B3Out,JoinArray40[Counter]:13,'        ');
        IndScores[GlobalCounter].Item:=Joinarray40[Counter];
        IndScores[GlobalCounter].BlockOrder:=Block3Order;
       End;
      IF (IDArray40[Counter]='U') THEN
       Begin
        Write(B3Out,JoinArray40[Counter]:13,'        ');
        IndScores[GlobalCounter].Item:=Joinarray40[Counter];
        IndScores[GlobalCounter].BlockOrder:=Block3Order;
       End;
    END;
   IF NOT B3Subliminal THEN
      BEGIN
       IF (IDArray40[Counter]= 'W') THEN
        Begin
         Write(B3Out,JoinArray40[Counter]:13);
         IndScores[GlobalCounter].Item:=Joinarray40[Counter];
        End;
       IF (IDArray40[Counter]='B') THEN
        Begin
         Write(B3Out,Joinarray40[Counter]:13);
         IndScores[GlobalCounter].Item:=Joinarray40[Counter];
        End;
       IF (IDArray40[Counter]='P') THEN
        Begin
         Write(B3Out,JoinArray40[Counter]:13);
         IndScores[GlobalCounter].Item:=Joinarray40[Counter];
        End;
       IF (IDArray40[Counter]='U') THEN
       Begin
        Write(B3Out,JoinArray40[Counter]:13);
        IndScores[GlobalCounter].Item:=Joinarray40[Counter];
       End;
        Write(B3Out,' ',Block3Order);
        IndScores[GlobalCounter].BlockOrder:=Block3Order;
      END;
{   if MaskFlag then Write(B3Out,'  ','1')
   Else Write(B3Out,'  ',0);
   Write(B3Out,'  ',SetupType);}
  Writeln(B3Out);
End;
```

```
Procedure Block4Report(Flag : Integer);
Begin
  Nowdate:=DatetoStr(Date);
  Nowtime:=TimetoStr(Time);
  Inc(GlobalCounter);
  IndScores[GlobalCounter].ID:=NameCode;
  IndScores[GlobalCounter].Block:=4;
  IndScores[GlobalCounter].Date:=Nowdate;
  IndScores[GlobalCounter].Time:=NowTime;
  IndScores[GlobalCounter].Gender:=Gend;
  IndScores[GlobalCounter].Age:=Age;
  IndScores[GlobalCounter].QResponse:=VRating;
  if COR then
    Begin
     if CCyan then IndScores[GlobalCounter].Colour:='P' Else IndScores[GlobalCounter].Colour:='M';
    End;
  if NOT (COR) then IndScores[GlobalCounter].Colour:='N';
  IF B4Subliminal Then
    Begin
      IndScores[GlobalCounter].SubTimer1:=IntToStr(TS2);
      IndScores[GlobalCounter].SubTimer2:=IntToStr(TS3);
    End
      Else
        Begin
          IndScores[GlobalCounter].SubTimer1:='000';
          IndScores[GlobalCounter].SubTimer2:='000';
        End;
  IndScores[GlobalCounter].Counter:=Counter;
  IndScores[GlobalCounter].Item:=Joinarray[Counter];
  IndScores[GlobalCounter].Diff:=Diff;
  IndScores[GlobalCounter].Flag:=IntToStr(Flag);
  if (B4) AND (B4Subliminal) then IndScores[GlobalCounter].SubItem:=Subjoin[Counter];
  IF (B4) AND Not(B4Subliminal) Then
     IndScores[GlobalCounter].SubItem:=Joinarray[Counter];
  IndScores[GlobalCounter].BlockOrder:=Block4Order;
  Write(B4Out,Nowdate :10,'  ');
  Write(B4Out,Nowtime: 11,'  ');
  Write(B4Out,Namecode :6,' ');
  Write(B4Out,' 4');
  IF COR THEN
    BEGIN
      IF CCyan Then Write(B4Out,' C') Else Write(B4Out,' R');
    END;
  IF NOT (COR) Then Write(B4Out,' N');
  IF B4Subliminal Then Write(B4Out,' ',TS2: 3,' ',TS3: 3) ELSE
  Write(B4Out,' ','000',' ','000');
  Write(B4Out,Joinarray[Counter]:13,' ');
//  Write(B4Out,Q1to5,' ');
  Write(B4Out,'..... ');
  Write(B4Out,Counter:2,' ');
  Write(B4Out,Diff:6,' ');
  Write(B4Out,Flag,' ');
  IF (B4Subliminal) Then
      Write(B4Out,Subjoin[Counter] :13);
```

```
  IF Not(B4Subliminal) Then
     Write(B4Out,Joinarray[Counter]: 13);
{  Whatis:=JoinArray[Counter];
  WhatRace;   }
  Write(B4Out,' ',Block4Order);
{  Write(B4Out,' ',whatis);
  if MaskFlag then Write(B4Out,'  ','1')
  Else Write(B4Out,'  ',0);
  Write(B4Out,'  ',SetupType); }
  Writeln(B4Out);
End;

Procedure Block5Report(Flag : Integer);
Begin
  Nowdate:=DatetoStr(Date);
  Nowtime:=TimetoStr(Time);
  Inc(GlobalCounter);
  IndScores[GlobalCounter].ID:=NameCode;
  IndScores[GlobalCounter].Block:=5;
  IndScores[GlobalCounter].Date:=Nowdate;
  IndScores[GlobalCounter].Time:=NowTime;
  IndScores[GlobalCounter].Gender:=Gend;
  IndScores[GlobalCounter].Age:=Age;
  IndScores[GlobalCounter].QResponse:=VRating;
  if COR then
    Begin
     if CCyan then IndScores[GlobalCounter].Colour:='P' Else IndScores[GlobalCounter].Colour:='M';
    End;
  if NOT (COR) then IndScores[GlobalCounter].Colour:='N';
  IF B5Subliminal Then
    Begin
      IndScores[GlobalCounter].SubTimer1:=IntToStr(TS2);
      IndScores[GlobalCounter].SubTimer2:=IntToStr(TS3);
    End
      Else
        Begin
          IndScores[GlobalCounter].SubTimer1:='000';
          IndScores[GlobalCounter].SubTimer2:='000';
        End;
  IndScores[GlobalCounter].Counter:=Counter;
  IndScores[GlobalCounter].Item:=Joinarray40[Counter];
  IndScores[GlobalCounter].Diff:=Diff;
  IndScores[GlobalCounter].Flag:=IntToStr(Flag);
  Write(B5Out,Nowdate :10,'  ');
  Write(B5Out,Nowtime: 11,'  ');
  Write(B5Out,Namecode :6,' ');
  Write(B5Out,' 5');
  IF COR THEN
    BEGIN
      IF CCyan Then Write(B5Out,' C') Else Write(B5Out,' R');
    END;
  IF NOT (COR) Then Write(B5Out,' N');
  IF B5Subliminal Then Write(B5Out,' ',GS2: 3,' ',TS3: 3) ELSE
  Write(B5Out,' ','000',' ','000');
```

```
  Write(B5Out,Joinarray40[Counter]:13,' ');
//  Write(B5Out,Q1to5,' ');
  Write(B5Out,'..... ');
  Write(B5Out,Counter:2,' ');
  Write(B5Out,Diff:6,' ');
  Write(B5Out,Flag,' ');
  IF B5Subliminal THEN
     BEGIN
      IF (IDArray40[Counter]= 'FW') OR (IDArray40[Counter]='MW')
      THEN
      Begin
        Write(B5Out,JoinArray40[Counter]:13);
        Whatis:=JoinArray40[Counter];
        WhatRace;
        Write(B5Out,' ',Block5Order);
        Write(B5Out,' ',Whatis);
        IndScores[GlobalCounter].SubItem:=(JoinArray40[Counter]+' '+Whatis);
        IndScores[GlobalCounter].BlockOrder:=Block5Order;
      End;
      IF (IDArray40[Counter]='FB') Or (IDArray40[Counter]='MB') THEN
      Begin
        Write(B5Out,JoinArray40[Counter]:13);
        Whatis:=JoinArray40[Counter];
        WhatRace;
        Write(B5Out,' ',Block5Order);
        Write(B5Out,' ',Whatis);
        IndScores[GlobalCounter].SubItem:=(JoinArray40[Counter]+' '+Whatis);
        IndScores[GlobalCounter].BlockOrder:=Block5Order;
      End;
      IF (IDArray40[Counter]='P') THEN
      Begin
        Write(B5Out,JoinArray40[Counter]:13,'       ');
        IndScores[GlobalCounter].SubItem:=Joinarray40[Counter];
      End;
      IF (IDArray40[Counter]='U') THEN
      Begin
        Write(B5Out,JoinArray40[Counter]:13,'       ');
        IndScores[GlobalCounter].SubItem:=Joinarray40[Counter];
      End;
     END;
   IF NOT B5Subliminal THEN
      BEGIN
       IF (IDArray40[Counter]= 'W') THEN
         Begin
           Write(B5Out,JoinArray40[Counter]:13);
           IndScores[GlobalCounter].Item:=Joinarray40[Counter];
         End;
       IF (IDArray40[Counter]='B') THEN
         Begin
           Write(B5Out,Joinarray40[Counter]:13);
           IndScores[GlobalCounter].Item:=Joinarray40[Counter];
         End;
       IF (IDArray40[Counter]='P') THEN
         Begin
```

```
          Write(B5Out,JoinArray40[Counter]:13);
          IndScores[GlobalCounter].Item:=Joinarray40[Counter];
         End;
       IF (IDArray40[Counter]='U') THEN
         Begin
          Write(B5Out,JoinArray40[Counter]:13);
          IndScores[GlobalCounter].Item:=Joinarray40[Counter];
         End;
         Write(B5Out,' ',Block5Order);
         IndScores[GlobalCounter].BlockOrder:=Block5Order;
       END;
{      if MaskFlag then Write(B5Out,'  ','1')
      Else Write(B5Out,'  ',0);
  Write(B5Out,'  ',SetupType);  }
  Writeln(B5Out);
End;

Procedure WriteIndDBFile;
Var IDX : Integer;
Begin
  TWinForm1.UpdatePrompt;
  With TWinForm1.ADOIndTable Do
  Begin
    TWinForm1.ADOIndTable.TableName:='Ind'+Namecode;
    TWinForm1.ADOIndTable.Open;
    TWinForm1.ADOIndTable.Active:=true;
    for IDX := 1 to 140 do
      Begin
        Append;
        Fields.FieldByName('ID').Value:=IndScores[IDX].ID;
        Fields.FieldByName('Block').Value:=IndScores[IDX].Block;
        Fields.FieldByName('Date').Value:=IndScores[IDX].Date;
        Fields.FieldByName('Time').Value:=IndScores[IDX].Time;
        Fields.FieldByName('Gender').Value:=IndScores[IDX].Gender;
        Fields.FieldByName('Age').Value:=IndScores[IDX].Age;
        Fields.FieldByName('QResponse').Value:=IndScores[IDX].QResponse;
        Fields.FieldByName('Colour').Value:=IndScores[IDX].Colour;
        Fields.FieldByName('SubTimer1').Value:=IndScores[IDX].SubTimer1;
        Fields.FieldByName('SubTimer2').Value:=IndScores[IDX].SubTimer2;
        Fields.FieldByName('Counter').Value:=IndScores[IDX].Counter;
        Fields.FieldByName('Diff').Value:=IndScores[IDX].Diff;
        Fields.FieldByName('Item').Value:=IndScores[IDX].Item;
        Fields.FieldByName('SubItem').Value:=IndScores[IDX].SubItem;
        Fields.FieldByName('Flag').Value:=IndScores[IDX].Flag;
        Fields.FieldByName('BlockOrder').Value:=IndScores[IDX].BlockOrder;
 //       UpDateRecord;
//        Post;
        TWinForm1.Progressbar2.StepBy(1);
      End;
      UpDateRecord;
      Post;
  End;
End;
```

```
Procedure ReportBlocks(B1,B2,B3,B4,B5 : Boolean; Flag : Integer);
Begin
  If (B1) And (Counter=1) Then FileFunctions.OpenB1;
  If B1 then  Block1Report(Flag);
  IF (B1) And (Counter=BT1) Then FileFunctions.CloseB1;

  If (B2) And (Counter=1) Then FileFunctions.OpenB2;
  If B2 then  Block2Report(Flag);
  IF (B2) And (Counter=BT2) Then FileFunctions.CloseB2;

  IF (B3) And (Counter=1) Then FileFunctions.OpenB3;
  If B3 then Block3Report(Flag);
  If (B3) and (Counter=BT3) Then FileFunctions.CloseB3;

  If (B4) And (Counter=1) Then FileFunctions.OpenB4;
  IF B4 Then Block4Report(Flag);
  If (B4) And (Counter=BT4) Then   FileFunctions.CloseB4;

  If (B5) And (Counter=1) Then FileFunctions.OpenB5;
  If B5 Then Block5Report(Flag);
  If (B5) And (Counter=BT5) Then FileFunctions.CloseB5;
End;

Procedure Report(Flag: Integer);
//VAR Nowtime,Nowdate: String;
BEGIN
    Nowdate:=DatetoStr(Date);
    Nowtime:=TimetoStr(Time);
    Write(outfile,Nowdate :10,'  ');
    Write(outfile,Nowtime: 11,'  ');
    Write(outfile,Namecode :6,' ');
    IF (B1=True) Then
    BEGIN
      Write(Outfile,' 1');
      IF COR THEN
        BEGIN
          IF CCyan Then Write(Outfile,' C') Else Write(Outfile,' R');
        END;
      IF NOT (COR) Then Write(Outfile,' N');
      IF B1Subliminal Then Write(Outfile,' ',TS2: 3,' ',TS3: 3) ELSE
      Write(Outfile,' ','000',' ','000');
    END;

    IF (B2=True) THEN
    BEGIN
      Write(Outfile,' 2');
      IF COR THEN
        BEGIN
          IF CCyan Then Write(Outfile,' C') Else Write(Outfile,' R');
        END;
      IF NOT (COR) Then Write(Outfile,' N');
      IF B2Subliminal Then Write(Outfile,' ',TS2: 3,' ',TS3: 3) ELSE
      Write(Outfile,' ','000',' ','000');
```

```
END;

IF (B3=True) THEN
BEGIN
  Write(Outfile,' 3');
  IF COR THEN
    BEGIN
      IF CCyan Then Write(Outfile,' C') Else Write(Outfile,' R');
    END;
  IF NOT (COR) Then Write(Outfile,' N');
  IF B3Subliminal Then Write(Outfile,' ',TS2: 3,' ',TS3: 3) ELSE
  Write(Outfile,' ','000',' ','000');
END;

IF (B4=True) THEN
BEGIN
  Write(Outfile,' 4');
  IF COR THEN
    BEGIN
      IF CCyan Then Write(Outfile,' C') Else Write(Outfile,' R');
    END;
  IF NOT (COR) Then Write(Outfile,' N');
  IF B4Subliminal Then Write(Outfile,' ',TS2: 3,' ',TS3: 3) ELSE
  Write(Outfile,' ','000',' ','000');
END;

IF(B5=True) THEN
BEGIN
  Write(Outfile,' 5');
  IF COR THEN
    BEGIN
      IF CCyan Then Write(Outfile,' C') Else Write(Outfile,' R');
    END;
  IF NOT (COR) Then Write(Outfile,' N');
  IF B5Subliminal Then Write(Outfile,' ',TS2: 3,' ',TS3: 3) ELSE
  Write(Outfile,' ','000',' ','000');
END;

IF (B1=True) OR (B2=True) OR (B4=True) THEN
BEGIN
Write(Outfile,Joinarray[Counter]:13,' ');
END;

IF (B3=True) OR (B5=True) THEN
BEGIN
  Write(Outfile,Joinarray40[Counter]:13,' ');
END;

Write(Outfile,Q1to5,' ');
Write(Outfile,Counter:2,' ');
Write(Outfile,Diff:6,' ');
Write(Outfile,Flag,' ');
IF (B1) then
  Write(Outfile,Joinarray[Counter]:13);
```

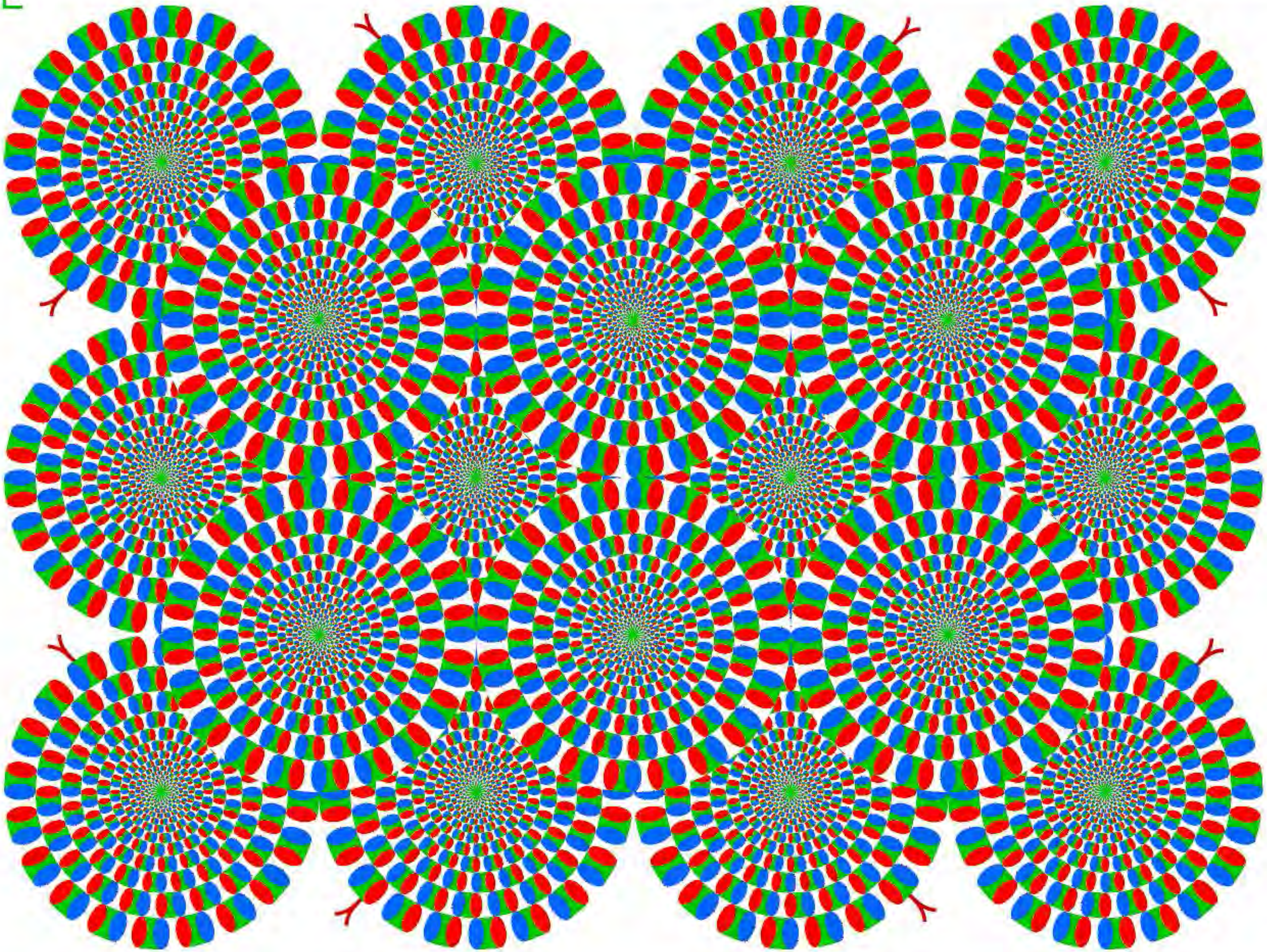
```
   IF (B2) AND (B2Subliminal) Then
     Write(Outfile,Subjoin[Counter] :13);
   IF (B2) AND Not(B2Subliminal) Then
     Write(Outfile,Joinarray[Counter]: 13);
   IF (B3) AND (B3Subliminal) THEN
     BEGIN
       IF (IDArray40[Counter]= 'W') THEN
         Write(Outfile,SubWhite[PrimeW]:13);
       IF (IDArray40[Counter]='B') THEN
         Write(Outfile,SubBlack[PrimeB]:13);
       IF (IDArray40[Counter]='P') THEN
         Write(Outfile,JoinArray40[Counter]:13);
       IF (IDArray40[Counter]='U') THEN
         Write(Outfile,JoinArray40[Counter]:13);
     END;
   IF (B3) AND NOT(B3Subliminal) THEN
       BEGIN
        IF (IDArray40[Counter]= 'W') THEN
          Write(Outfile,JoinArray40[Counter]:13);
        IF (IDArray40[Counter]='B') THEN
          Write(Outfile,Joinarray40[Counter]:13);
        IF (IDArray40[Counter]='P') THEN
          Write(Outfile,JoinArray40[Counter]:13);
        IF (IDArray40[Counter]='U') THEN
          Write(Outfile,JoinArray40[Counter]:13);
       END;
   IF (B4) AND (B4Subliminal) Then
     Write(Outfile,Subjoin[Counter] :13);
   IF (B4) AND Not(B4Subliminal) Then
     Write(Outfile,Joinarray[Counter]: 13);
   IF (B5) AND (B5Subliminal) THEN
     BEGIN
       IF (IDArray40[Counter]= 'W') THEN
         Write(Outfile,SubWhite[PrimeW]:13);
       IF (IDArray40[Counter]='B') THEN
         Write(Outfile,SubBlack[PrimeB]:13);
       IF (IDArray40[Counter]='P') THEN
         Write(Outfile,JoinArray40[Counter]:13);
       IF (IDArray40[Counter]='U') THEN
         Write(Outfile,JoinArray40[Counter]:13);
     END;
   IF (B5) AND NOT(B5Subliminal) THEN
       BEGIN
        IF (IDArray40[Counter]= 'W') THEN
          Write(Outfile,JoinArray40[Counter]:13);
        IF (IDArray40[Counter]='B') THEN
          Write(Outfile,Joinarray40[Counter]:13);
        IF (IDArray40[Counter]='P') THEN
          Write(Outfile,JoinArray40[Counter]:13);
        IF (IDArray40[Counter]='U') THEN
          Write(Outfile,JoinArray40[Counter]:13);
       END;
   Writeln(Outfile);
END;
```

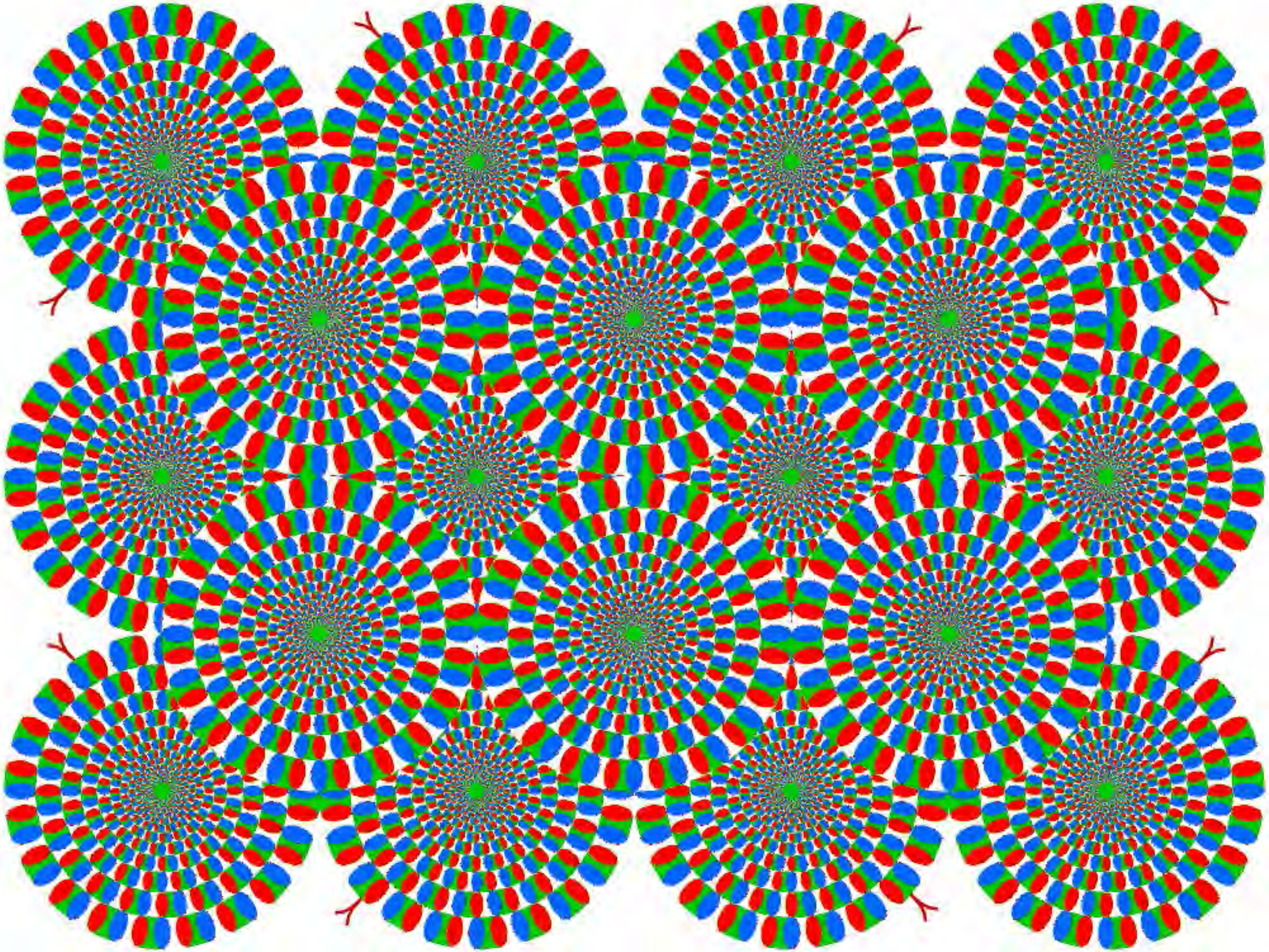
end.

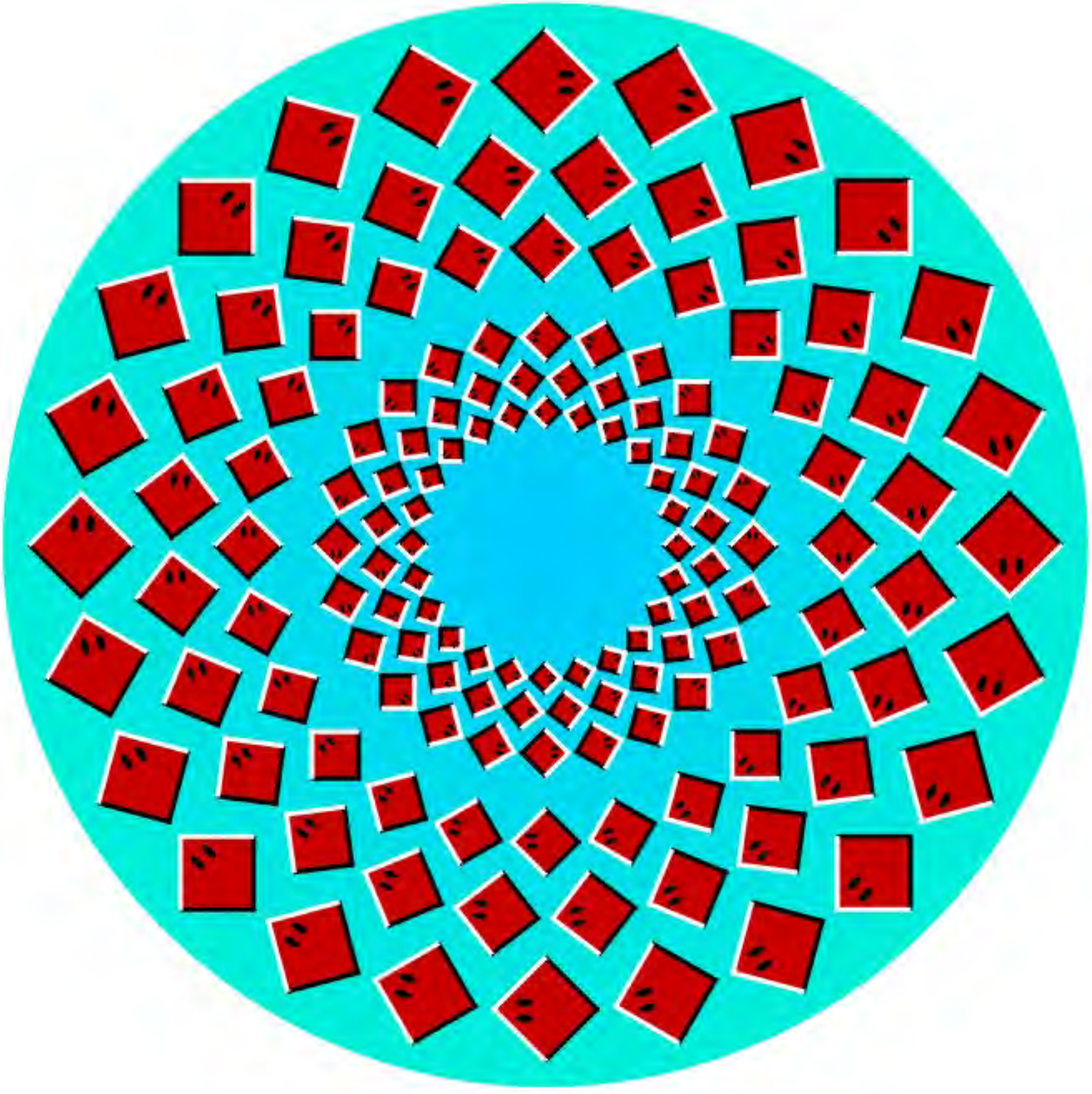
```
unit ResultUnit;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, TeeProcs, TeEngine, Chart, StdCtrls, Series, TeeFunci;

type
  TResults = class(TForm)
    Memo1: TMemo;
    Chart1: TChart;
    Series1: TBarSeries;
    Chart2: TChart;
    Series2: TLineSeries;
    SaveText: TButton;
    SaveBar: TButton;
    SaveLine: TButton;
    SaveDialog1: TSaveDialog;
    Series3: TLineSeries;
    procedure SaveTextClick(Sender: TObject);
    procedure SaveBarClick(Sender: TObject);
    procedure SaveLineClick(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Results: TResults;

implementation

{$R *.dfm}

procedure TResults.SaveBarClick(Sender: TObject);
begin
SaveDialog1.DefaultExt:='bmp';
if SaveDialog1.Execute then
Chart1.SaveToBitmapFile(SaveDialog1.FileName);
end;

procedure TResults.SaveLineClick(Sender: TObject);
begin
SaveDialog1.DefaultExt:='bmp';
if SaveDialog1.Execute then
Chart2.SaveToBitmapFile(SaveDialog1.FileName);
end;

procedure TResults.SaveTextClick(Sender: TObject);
begin
SaveDialog1.DefaultExt:='rtf';
```

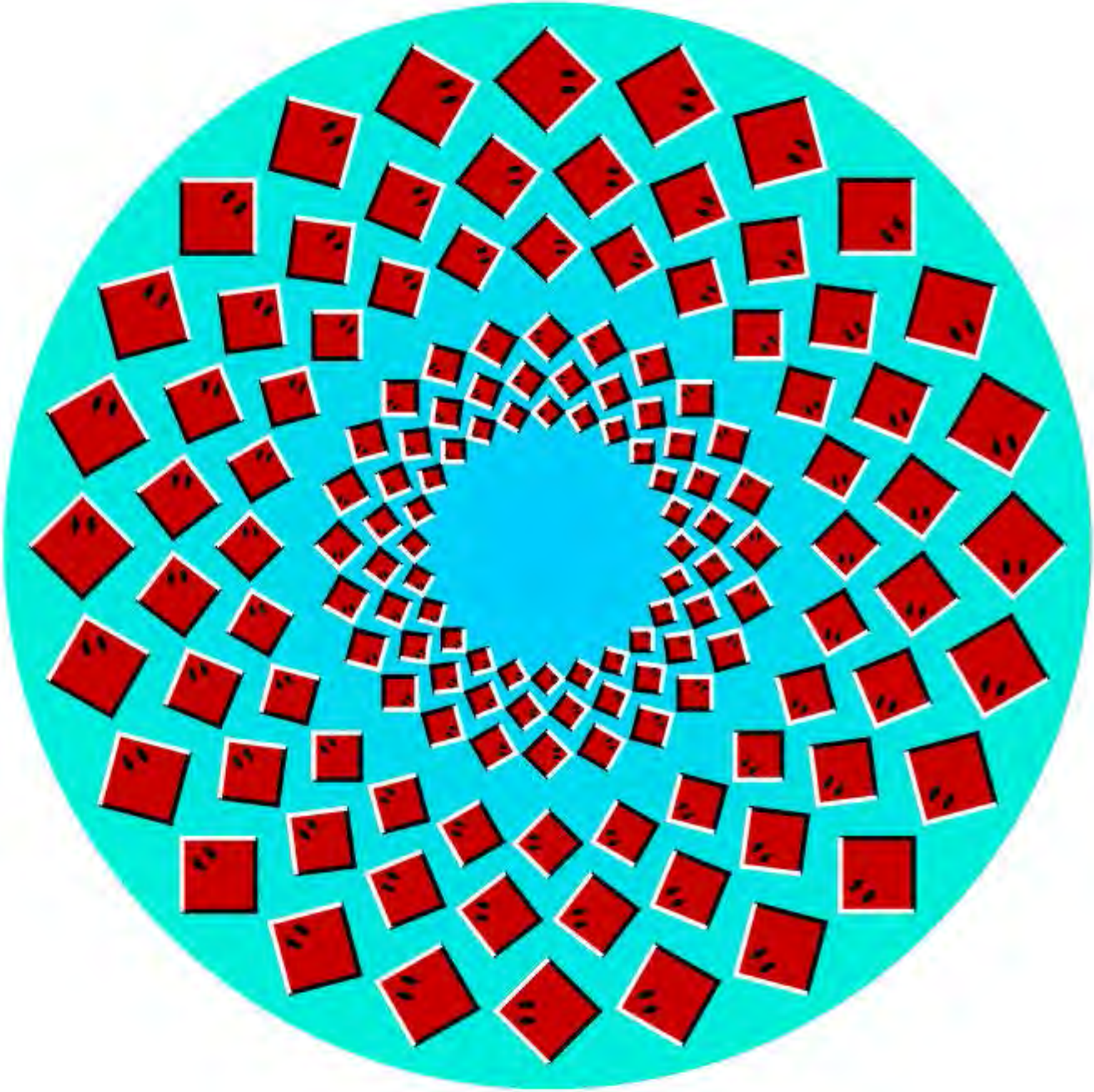
```
if SaveDialog1.Execute then
Memo1.Lines.SaveToFile(SaveDialog1.FileName);
end;

end.
```

E









Explore

Type

Case Processing Summary

		Cases					
		Valid		Missing		Total	
	Type	N	Percent	N	Percent	N	Percent
Dprime	BW	49	100.0%	0	.0%	49	100.0%
	M	50	100.0%	0	.0%	50	100.0%
	P	50	100.0%	0	.0%	50	100.0%

Descriptives

	Type			Statistic	Std. Error
Dprime	BW	Mean		.426867	.0701447
		95% Confidence Interval for Mean	Lower Bound	.285832	
			Upper Bound	.567903	
		5% Trimmed Mean		.426390	
		Median		.431200	
		Variance		.241	
		Std. Deviation		.4910132	
		Minimum		-.7695	
		Maximum		1.6073	
		Range		2.3768	
		Interquartile Range		.7846	
		Skewness		-.037	.340
		Kurtosis		-.205	.668
	M	Mean		.489824	.0713516
		95% Confidence Interval for Mean	Lower Bound	.346438	
			Upper Bound	.633210	
		5% Trimmed Mean		.481278	
		Median		.507850	
		Variance		.255	
		Std. Deviation		.5045318	
		Minimum		-.4583	
		Maximum		1.6073	
		Range		2.0656	
		Interquartile Range		.7794	
		Skewness		.290	.337
		Kurtosis		-.536	.662

Descriptives

	Type			Statistic	Std. Error
Dprime	P	Mean		.171552	.1116525
		95% Confidence Interval for Mean	Lower Bound	-.052822	
			Upper Bound	.395926	
		5% Trimmed Mean		.235556	
		Median		.176600	
		Variance		.623	
		Std. Deviation		.7895027	
		Minimum		-4.5515	
		Maximum		1.2754	
		Range		5.8269	
		Interquartile Range		.4728	
		Skewness		-4.332	.337
		Kurtosis		26.688	.662

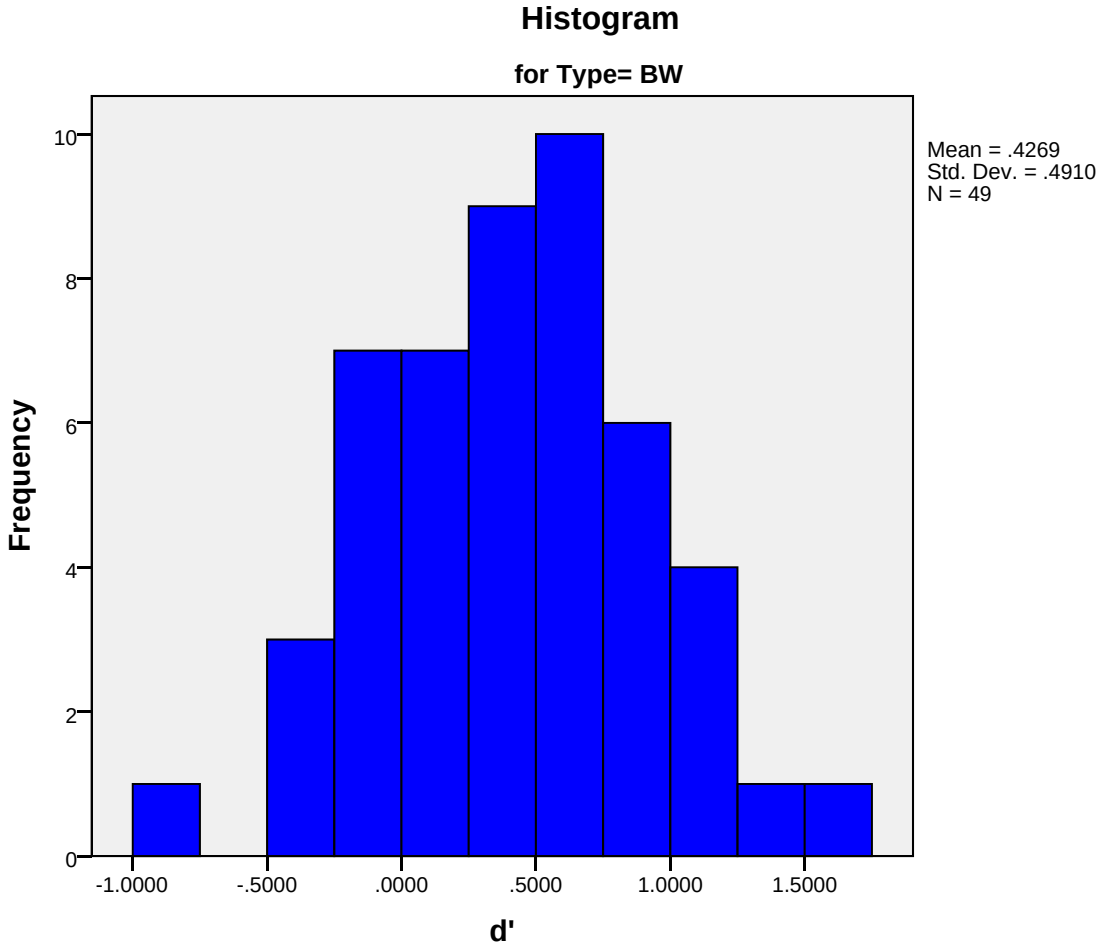
Tests of Normality

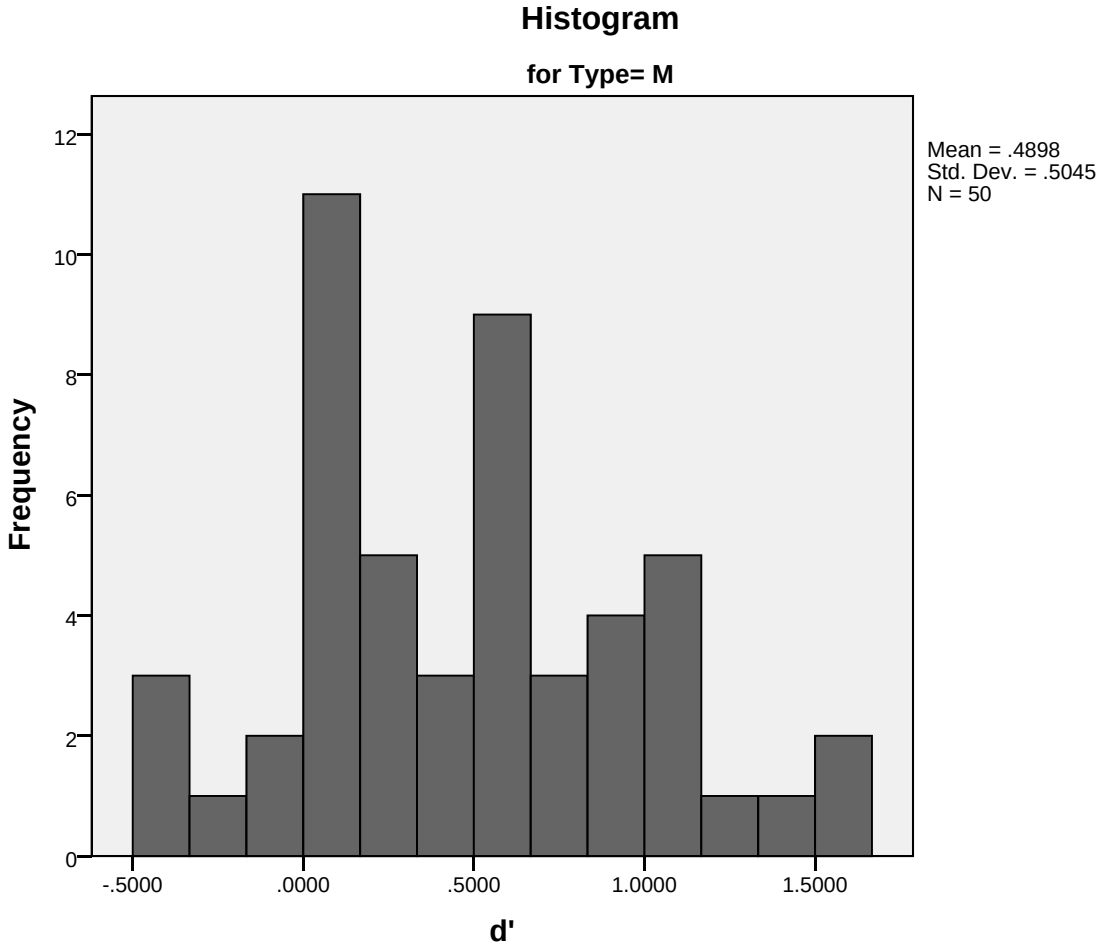
		Kolmogorov-Smirnov [a]			Shapiro-Wilk		
	Type	Statistic	df	Sig.	Statistic	df	Sig.
Dprime	BW	.070	49	.200*	.992	49	.977
	M	.127	50	.041	.972	50	.290
	P	.261	50	.000	.584	50	.000

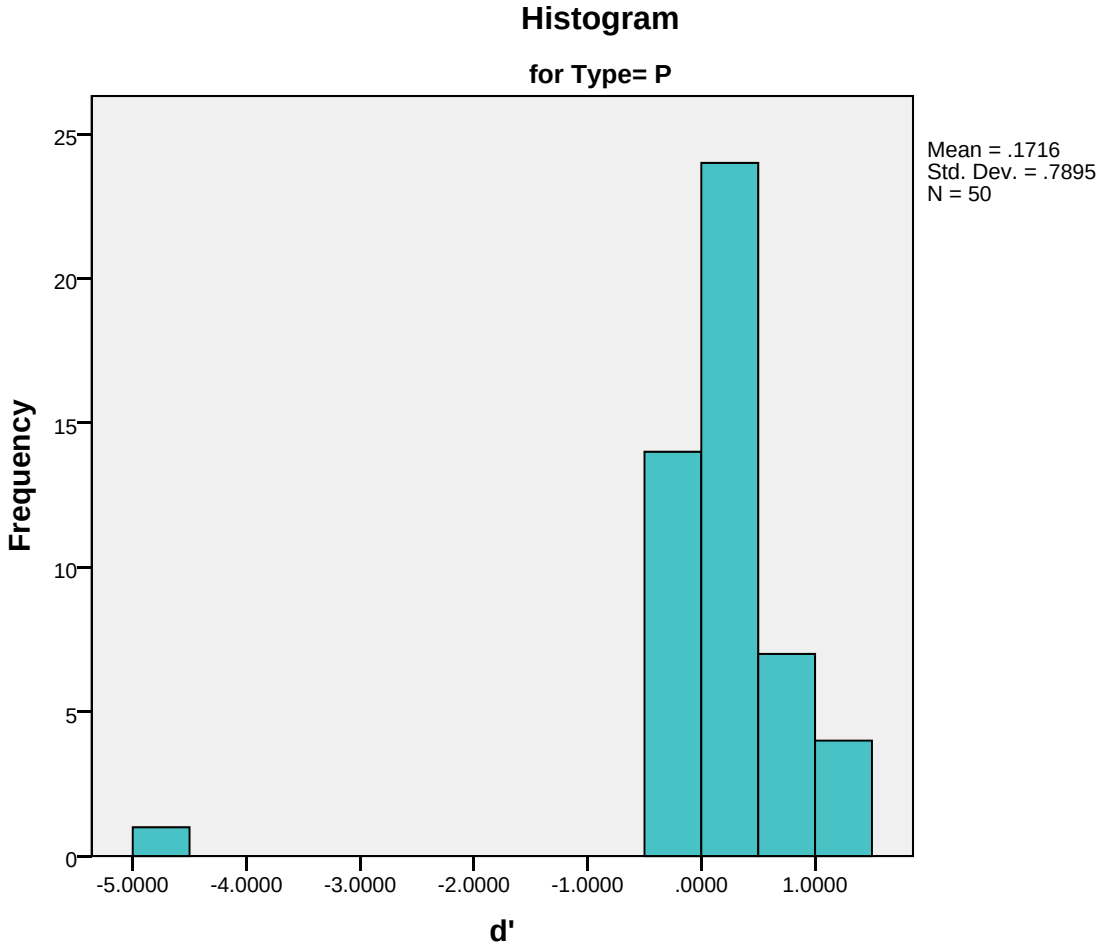
a. Lilliefors Significance Correction
*. This is a lower bound of the true significance.

Dprime

Histograms


Histogram
for Type= BW
Mean = .4269
Std. Dev. = .4910
N = 49
Frequency
10
8
6
4
2
0
-1.0000
-.5000
.0000
.5000
1.0000
1.5000
d'


Histogram
for Type= M
Mean = .4898
Std. Dev. = .5045
N = 50
Frequency
0
2
4
6
8
10
12
-.5000
.0000
.5000
1.0000
1.5000
d'


Histogram
for Type= P
Mean = .1716
Std. Dev. = .7895
N = 50
Frequency
25
20
15
10
5
0
-5.0000
-4.0000
-3.0000
-2.0000
-1.0000
.0000
1.0000
d'

```
unit ScoreAndUpDateM;

interface
Procedure ScoreMUpdate;
Procedure ScorePUpdate;
Procedure WritePreviousStage;
Procedure ScoreMStages;

implementation
Uses CPT,FileFunctions,DB,SysUtils;

Procedure EndupTime;
Begin
  StopTime:=Now;
End;

Procedure ScoreMStages;
Begin
   if Stage=1 then
     if (GlobalCounter>1) And (Scores1[GlobalCounter-1].KeyPressed<>1) then
     Begin
     EndUpTime;
       If Both[GlobalCounter-1].Value=0 Then
         Begin
           Scores1[GlobalCounter-1].FirstWordT:='Bigger';
           Scores1[GlobalCounter-1].SecondWordT:=Both[GlobalCounter-1].Word;
           Scores1[GlobalCounter-1].TimeE:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
           Scores1[GlobalCounter-1].Value:=0;
           Scores1[GlobalCounter-1].KeyPressed:=0;
           Scores1[GlobalCounter-1].ReportSeen:='False Negative';
           Scores1[GlobalCounter-1].TargetType:=Both[GlobalCounter-1].TargetType;
         End;
       If Both[GlobalCounter-1].Value=1 Then
         Begin
           if Both[GlobalCounter-1].Word<>'Null' then
             Scores1[GlobalCounter-1].FirstWordT:='Smaller'
               Else
                 Scores1[GlobalCounter-1].FirstWordT:='Null';
           Scores1[GlobalCounter-1].SecondWordT:=Both[GlobalCounter-1].Word;
           Scores1[GlobalCounter-1].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
           Scores1[GlobalCounter-1].Value:=1;
           Scores1[GlobalCounter-1].KeyPressed:=0;
           Scores1[GlobalCounter-1].ReportSeen:='True Negative';
           Scores1[GlobalCounter-1].TargetType:=Both[GlobalCounter-1].TargetType;
         End;
     WritePreviousStage;
     End;
     if Stage=2 then
     if (GlobalCounter>1) And (Scores2[GlobalCounter-1].KeyPressed<>1) then
     Begin
     EndUpTime;
       If Both[GlobalCounter-1].Value=0 Then
         Begin
```

```
      Scores2[GlobalCounter-1].FirstWordT:='Bigger';
      Scores2[GlobalCounter-1].SecondWordT:=Both[GlobalCounter-1].Word;
      Scores2[GlobalCounter-1].TimeE:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
      Scores2[GlobalCounter-1].Value:=0;
      Scores2[GlobalCounter-1].KeyPressed:=0;
      Scores2[GlobalCounter-1].ReportSeen:='False Negative';
      Scores2[GlobalCounter-1].TargetType:=Both[GlobalCounter-1].TargetType;
    End;
  If Both[GlobalCounter-1].Value=1 Then
    Begin
      if Both[GlobalCounter-1].Word<>'Null' then
        Scores2[GlobalCounter-1].FirstWordT:='Smaller'
          Else
            Scores2[GlobalCounter-1].FirstWordT:='Null';
      Scores2[GlobalCounter-1].SecondWordT:=Both[GlobalCounter-1].Word;
      Scores2[GlobalCounter-1].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
      Scores2[GlobalCounter-1].Value:=1;
      Scores2[GlobalCounter-1].KeyPressed:=0;
      Scores2[GlobalCounter-1].ReportSeen:='True Negative';
      Scores2[GlobalCounter-1].TargetType:=Both[GlobalCounter-1].TargetType;
    End;
    WritePreviousStage;
End;
if Stage=3 then
if (GlobalCounter>1) And (Scores3[GlobalCounter-1].KeyPressed<>1) then
Begin
EndUpTime;
  If Both[GlobalCounter-1].Value=0 Then
    Begin
      Scores3[GlobalCounter-1].FirstWordT:='Bigger';
      Scores3[GlobalCounter-1].SecondWordT:=Both[GlobalCounter-1].Word;
      Scores3[GlobalCounter-1].TimeE:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
      Scores3[GlobalCounter-1].Value:=0;
      Scores3[GlobalCounter-1].KeyPressed:=0;
      Scores3[GlobalCounter-1].ReportSeen:='False Negative';
      Scores3[GlobalCounter-1].TargetType:=Both[GlobalCounter-1].TargetType;
    End;
  If Both[GlobalCounter-1].Value=1 Then
    Begin
      if Both[GlobalCounter-1].Word<>'Null' then
        Scores3[GlobalCounter-1].FirstWordT:='Smaller'
          Else
            Scores3[GlobalCounter-1].FirstWordT:='Null';
      Scores3[GlobalCounter-1].SecondWordT:=Both[GlobalCounter-1].Word;
      Scores3[GlobalCounter-1].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
      Scores3[GlobalCounter-1].Value:=1;
      Scores3[GlobalCounter-1].KeyPressed:=0;
      Scores3[GlobalCounter-1].ReportSeen:='True Negative';
      Scores3[GlobalCounter-1].TargetType:=Both[GlobalCounter-1].TargetType;
    End;
    WritePreviousStage;
End;
if Stage=4 then
if (GlobalCounter>1) And (Scores4[GlobalCounter-1].KeyPressed<>1) then
```

```
    Begin
    EndUpTime;
      If Both[GlobalCounter-1].Value=0 Then
        Begin
          Scores4[GlobalCounter-1].FirstWordT:='Bigger';
          Scores4[GlobalCounter-1].SecondWordT:=Both[GlobalCounter-1].Word;
          Scores4[GlobalCounter-1].TimeE:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
          Scores4[GlobalCounter-1].Value:=0;
          Scores4[GlobalCounter-1].KeyPressed:=0;
          Scores4[GlobalCounter-1].ReportSeen:='False Negative';
          Scores4[GlobalCounter-1].TargetType:=Both[GlobalCounter-1].TargetType;
        End;
      If Both[GlobalCounter-1].Value=1 Then
        Begin
          if Both[GlobalCounter-1].Word<>'Null' then
            Scores4[GlobalCounter-1].FirstWordT:='Smaller'
              Else
                Scores4[GlobalCounter-1].FirstWordT:='Null';
          Scores4[GlobalCounter-1].SecondWordT:=Both[GlobalCounter-1].Word;
          Scores4[GlobalCounter-1].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
          Scores4[GlobalCounter-1].Value:=1;
          Scores4[GlobalCounter-1].KeyPressed:=0;
          Scores4[GlobalCounter-1].ReportSeen:='True Negative';
          Scores4[GlobalCounter-1].TargetType:=Both[GlobalCounter-1].TargetType;
        End;
        WritePreviousStage;
    End;
End;

Procedure WritePreviousStage;
Begin
    if Stage=1 then
      With Form1.ADOMTable1 Do
       Begin
         Append;
         Fields.FieldByName('Stage').Value:=Stage;
         Fields.FieldByName('ID').Value:=NameCode;
         Fields.FieldByName('TimeT').Value:=Scores1[GlobalCounter-1].TimeT;
         Fields.FieldByName('Stimulus Type').Value:=Scores1[GlobalCounter-1].FirstWordT;
         Fields.FieldByName('Stimulus Item').Value:=Scores1[GlobalCounter-1].SecondWordT;
         iF Scores1[GlobalCounter-1].MLatency>0 Then Fields.FieldByName('Response
Latency').Value:=Scores1[GlobalCounter-1].MLatency;
         iF Scores1[GlobalCounter-1].MElatency>0 Then Fields.FieldByName('Error
Latency').Value:=Scores1[GlobalCounter-1].MElatency;
         Fields.FieldByName('Non Target').Value:=Scores1[GlobalCounter-1].FirstWordF;
         Fields.FieldByName('Non Target Item').Value:=Scores1[GlobalCounter-1].SecondWordF;
         Fields.FieldByName('TimeE').Value:=Scores1[GlobalCounter-1].TimeE;
         Fields.FieldByName('TimeT').Value:=Scores1[GlobalCounter-1].TimeT;
         Fields.FieldByName('Value').Value:=Scores1[GlobalCounter-1].Value;
         Fields.FieldByName('Report Seen').Value:=Scores1[GlobalCounter-1].ReportSeen;
         Fields.FieldByName('Target type').Value:=Scores1[GlobalCounter-1].TargetType;
         Fields.FieldByName('Key Pressed').Value:=Scores1[GlobalCounter-1].KeyPressed;
         UpdateRecord;
         Post;
```

```
      End;
    if Stage=2 then
      With Form1.ADOMTable1 Do
      Begin
        Append;
        Fields.FieldByName('Stage').Value:=Stage;
        Fields.FieldByName('ID').Value:=NameCode;
        Fields.FieldByName('TimeT').Value:=Scores2[GlobalCounter-1].TimeT;
        Fields.FieldByName('Stimulus Type').Value:=Scores2[GlobalCounter-1].FirstWordT;
        Fields.FieldByName('Stimulus Item').Value:=Scores2[GlobalCounter-1].SecondWordT;
        iF Scores2[GlobalCounter-1].MLatency>0 Then Fields.FieldByName('Response
Latency').Value:=Scores2[GlobalCounter-1].MLatency;
        iF Scores2[GlobalCounter-1].MElatency>0 Then Fields.FieldByName('Error
Latency').Value:=Scores2[GlobalCounter-1].MElatency;
        Fields.FieldByName('Non Target').Value:=Scores2[GlobalCounter-1].FirstWordF;
        Fields.FieldByName('Non Target Item').Value:=Scores2[GlobalCounter-1].SecondWordF;
        Fields.FieldByName('TimeE').Value:=Scores2[GlobalCounter-1].TimeE;
        Fields.FieldByName('TimeT').Value:=Scores2[GlobalCounter-1].TimeT;
        Fields.FieldByName('Value').Value:=Scores2[GlobalCounter-1].Value;
        Fields.FieldByName('Report Seen').Value:=Scores2[GlobalCounter-1].ReportSeen;
        Fields.FieldByName('Target type').Value:=Scores2[GlobalCounter-1].TargetType;
        Fields.FieldByName('Key Pressed').Value:=Scores2[GlobalCounter-1].KeyPressed;
        UpdateRecord;
        Post;
      End;
    if Stage=3 then
      With Form1.ADOMTable1 Do
      Begin
        Append;
        Fields.FieldByName('Stage').Value:=Stage;
        Fields.FieldByName('ID').Value:=NameCode;
        Fields.FieldByName('TimeT').Value:=Scores3[GlobalCounter-1].TimeT;
        Fields.FieldByName('Stimulus Type').Value:=Scores3[GlobalCounter-1].FirstWordT;
        Fields.FieldByName('Stimulus Item').Value:=Scores3[GlobalCounter-1].SecondWordT;
        iF Scores3[GlobalCounter-1].MLatency>0 Then Fields.FieldByName('Response
Latency').Value:=Scores3[GlobalCounter-1].MLatency;
        iF Scores3[GlobalCounter-1].MElatency>0 Then Fields.FieldByName('Error
Latency').Value:=Scores3[GlobalCounter-1].MElatency;
        Fields.FieldByName('Non Target').Value:=Scores3[GlobalCounter-1].FirstWordF;
        Fields.FieldByName('Non Target Item').Value:=Scores3[GlobalCounter-1].SecondWordF;
        Fields.FieldByName('TimeE').Value:=Scores3[GlobalCounter-1].TimeE;
        Fields.FieldByName('TimeT').Value:=Scores3[GlobalCounter-1].TimeT;
        Fields.FieldByName('Value').Value:=Scores3[GlobalCounter-1].Value;
        Fields.FieldByName('Report Seen').Value:=Scores3[GlobalCounter-1].ReportSeen;
        Fields.FieldByName('Target type').Value:=Scores3[GlobalCounter-1].TargetType;
        Fields.FieldByName('Key Pressed').Value:=Scores3[GlobalCounter-1].KeyPressed;
        UpdateRecord;
        Post;
      End;
    if Stage=4 then
      With Form1.ADOMTable1 Do
      Begin
        Append;
        Fields.FieldByName('Stage').Value:=Stage;
```

```
        Fields.FieldByName('ID').Value:=NameCode;
        Fields.FieldByName('TimeT').Value:=Scores4[GlobalCounter-1].TimeT;
        Fields.FieldByName('Stimulus Type').Value:=Scores4[GlobalCounter-1].FirstWordT;
        Fields.FieldByName('Stimulus Item').Value:=Scores4[GlobalCounter-1].SecondWordT;
        iF Scores4[GlobalCounter-1].MLatency>0 Then Fields.FieldByName('Response
Latency').Value:=Scores4[GlobalCounter-1].MLatency;
        iF Scores4[GlobalCounter-1].MElatency>0 Then Fields.FieldByName('Error
Latency').Value:=Scores4[GlobalCounter-1].MElatency;
        Fields.FieldByName('Non Target').Value:=Scores4[GlobalCounter-1].FirstWordF;
        Fields.FieldByName('Non Target Item').Value:=Scores4[GlobalCounter-1].SecondWordF;
        Fields.FieldByName('TimeE').Value:=Scores4[GlobalCounter-1].TimeE;
        Fields.FieldByName('TimeT').Value:=Scores4[GlobalCounter-1].TimeT;
        Fields.FieldByName('Value').Value:=Scores4[GlobalCounter-1].Value;
        Fields.FieldByName('Report Seen').Value:=Scores4[GlobalCounter-1].ReportSeen;
        Fields.FieldByName('Target type').Value:=Scores4[GlobalCounter-1].TargetType;
        Fields.FieldByName('Key Pressed').Value:=Scores4[GlobalCounter-1].KeyPressed;
        UpdateRecord;
        Post;
       End;
End;

Procedure ScorePUpdate;
Begin
   StopClock(EndTime);
   Diff:=Round(EndTime);
   Form1.Errornumber.Caption:=IntToStr(Errors);
   Form1.KeyCounter.Caption:=IntToStr(Counter);
  If Pract[GlobalCounter].Value=1 Then
    Begin
      EndupTime;
      Form1.Latency.Caption:=IntToStr(Diff);
      Counter:=Counter+1;
      ScoresP[GlobalCounter].FirstWordT:='Target';
      ScoresP[GlobalCounter].SecondWordT:=Letter;
      ScoresP[GlobalCounter].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
      ScoresP[GlobalCounter].M1Latency:=Diff;
      ScoresP[GlobalCounter].R:=CPT.DR-Equi;
      ScoresP[GlobalCounter].G:=CPT.DG-Equi;
      ScoresP[GlobalCounter].B:=CPT.DB-Equi;
      ScoresP[GlobalCounter].KeyPressed:=1;
      ScoresP[GlobalCounter].Value:=1;
      ScoresP[GlobalCounter].ReportSeen:='True Positive';
//      ScoresP[Counter].It:=Iteration*Sizer.Interval;
      ScoresP[Counter].Width:=Form1.Target.Width;
      ScoresP[Counter].Height:=Form1.Target.Height;
      ScoresP[Counter].Area:=Form1.Target.Width*Form1.Target.Height;
      ScoresP[Counter].PicPro:=(ScoresP[Counter].Area*1000) DIV Howlong;
      P1Tot:=P1Tot + Diff;
      If PEnabled Then
      Begin
        Form1.Correct.Visible:=true;
        Form1.Incorrect.Visible:=false;
      End;
    End;
```

```
  If (Pract[GlobalCOunter].Value= 0) Then // scores error response
    BEGIN
      EndupTime;
      Form1.ELatency.Caption:=IntToStr(Diff);
      Errors:=Errors+1;
      ScoresP[GlobalCounter].FirstWordT:='Non-target';
      ScoresP[GlobalCounter].SecondWordT:='blank';
      ScoresP[GlobalCounter].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
      ScoresP[GlobalCounter].M1Elatency:=Diff;
      ScoresP[GlobalCounter].FirstWordF:='blank';
      ScoresP[GlobalCounter].SecondWordF:=LetterF;
      ScoresP[GlobalCounter].TimeE:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
      ScoresP[GlobalCounter].R:=CPT.DR-Equi;
      ScoresP[GlobalCounter].G:=CPT.DG-Equi;
      ScoresP[GlobalCounter].B:=CPT.DB-Equi;
      ScoresP[GlobalCounter].KeyPressed:=1;
      ScoresP[GlobalCounter].Value:=0;
      ScoresP[GlobalCounter].ReportSeen:='False Positive';
//    ScoresP[Counter].It:=Iteration*Sizer.Interval;
    ScoresP[Counter].Width:=Form1.Target.Width;
    ScoresP[Counter].Height:=Form1.Target.Height;
    ScoresP[Counter].Area:=Form1.Target.Width*Form1.Target.Height;
    ScoresP[Counter].PicPro:=(ScoresP[Counter].Area*1000) DIV Howlong;
    P1ETot:=P1ETot + Diff;
    Form1.Correct.Visible:=false;
    If PEnabled then
      Begin
        Form1.Incorrect.Visible:=true;
      End;
  END;
  With Form1.ADOPTable1 Do
    Begin
      Append;
      Fields.FieldByName('ID').Value:=Namecode;
      Fields.FieldByName('TimeT').Value:=ScoresP[GlobalCounter].TimeT;
      Fields.FieldByName('Stimulus Type').Value:=ScoresP[GlobalCounter].FirstWordT;
      Fields.FieldByName('Stimulus Item').Value:=ScoresP[GlobalCounter].SecondWordT;
      iF ScoresP[GlobalCounter].M1Latency>0 Then Fields.FieldByName('Response
Latency').Value:=ScoresP[GlobalCounter].M1Latency;
      iF ScoresP[GlobalCounter].M1Elatency>0 Then Fields.FieldByName('Error
Latency').Value:=ScoresP[GlobalCounter].M1Elatency;
      Fields.FieldByName('Non Target').Value:=ScoresP[GlobalCounter].FirstWordF;
      Fields.FieldByName('Non Target Item').Value:=ScoresP[GlobalCounter].SecondWordF;
      Fields.FieldByName('TimeE').Value:=ScoresP[GlobalCounter].TimeE;
      Fields.FieldByName('Value').Value:=ScoresP[GlobalCounter].Value;
      Fields.FieldByName('Report Seen').Value:=ScoresP[GlobalCounter].ReportSeen;
      Fields.FieldByName('Key Pressed').Value:=ScoresP[GlobalCounter].KeyPressed;
      Fields.FieldByName('R').Value:=ScoresP[GlobalCounter].R;
      Fields.FieldByName('G').Value:=ScoresP[GlobalCounter].G;
      Fields.FieldByName('B').Value:=ScoresP[GlobalCounter].B;
      Form1.ADOPTable1.UpdateRecord;
      Post;
    End;
End;
```

```
Procedure ScoreMUpdate;
Begin
   CPT.StopClock(EndTime);
   Diff:=Round(EndTime);
   Stage:=StrToInt(Phase);
   if Stage=1 then
   Begin
    If Both[GlobalCounter].Value=0 Then
    Begin
     EndupTime;
     Form1.Latency.Caption:=IntToStr(Diff);
     Scores1[GlobalCounter].FirstWordT:='Bigger';
     Scores1[GlobalCounter].SecondWordT:=Letter;
     Scores1[GlobalCounter].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
     Scores1[GlobalCounter].MLatency:=Diff;
     Scores1[GlobalCounter].It:=Iteration*(Form1.Sizer.Interval);
     Scores1[GlobalCounter].Value:=1;
     Scores1[GlobalCounter].KeyPressed:=1;
     Scores1[GlobalCounter].ReportSeen:='True Positive';
     Scores1[GlobalCounter].TargetType:=Both[GlobalCounter].TargetType;
     if Both[GlobalCounter].TargetType=0 then
      Begin
       MScores[Stage].MTotM:=MScores[Stage].MTotM+Diff;
       MScores[Stage].MScoreM:=MScores[Stage].MScoreM+1;
       Form1.KeyCounter.Caption:=IntToStr(MScores[Stage].MScoreM);
      End;
     if Both[GlobalCounter].TargetType=1 then
      Begin
       MScores[Stage].MTotP:=MScores[Stage].MTotP+Diff;
       MScores[Stage].MScoreP:=MScores[Stage].MScoreP+1;
       Form1.KeyCounterP.Caption:=IntToStr(MScores[Stage].MScoreP);
      End;
     If MEnabled
      Then Form1.Correct.Visible:=true;
     Form1.Incorrect.Visible:=false;
    End;
    If (Both[GlobalCounter].Value=1) Then // scores error response
    BEGIN
     EndupTime;
     Form1.ELatency.Caption:=IntToStr(Diff);
     If Both[GlobalCounter].Word<>'Null' Then
     Scores1[GlobalCounter].FirstWordT:='Smaller'
      Else
       Scores1[GlobalCounter].FirstWordT:='Null';
     Scores1[GlobalCounter].SecondWordT:=Letter;
     Scores1[GlobalCounter].MElatency:=Diff;
     Scores1[GlobalCounter].EIt:=Iteration*(Form1.Sizer.Interval);
     Scores1[GlobalCounter].Value:=0;
     Scores1[GlobalCounter].KeyPressed:=1;
     Scores1[GlobalCounter].TimeE:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
     Scores1[GlobalCounter].TargetType:=Both[GlobalCounter].TargetType;
     If Both[GlobalCounter].Word<>'Null' Then
```

```
    Scores1[GlobalCounter].ReportSeen:='False Positive'
     Else
      Scores1[GlobalCounter].ReportSeen:='False Null';
  //Score M errors and increment the ELatency Array
  If (Both[GlobalCounter].Word<>'Null') AND (Both[GlobalCounter].TargetType=0) then
    Begin
     MScores[Stage].MErrorsM:=MScores[Stage].MErrorsM+1;
     Form1.ErrorNumber.Caption:=IntToStr(MScores[Stage].MErrorsM);
     MScores[Stage].ELatencyM:=MScores[Stage].ELatencyM+Diff;
    End;
  // Score M NullErrors
  IF (Both[GlobalCounter].Word='Null') AND (Both[GlobalCounter].TargetType=0) then
      Begin
       MScores[Stage].NullErrorsM:=MScores[Stage].NullErrorsM+1;
       Form1.NullErrorsM.Caption:=IntToStr(MScores[Stage].NullErrorsM);
      End;
 // Score P Errors and increment the ELatency Array
  If (Both[GlobalCounter].Word<>'Null') AND (Both[GlobalCounter].TargetType=1) Then
    Begin
     MScores[Stage].MErrorsP:=MScores[Stage].MErrorsP+1;
     Form1.ErrorNumberP.Caption:=IntToStr(MScores[Stage].MErrorsP);
     MScores[Stage].ELatencyP:=MScores[Stage].ELatencyP+Diff;
    End;
 // Score P Null Errors
  IF (Both[GlobalCounter].Word='Null') AND (Both[GlobalCounter].TargetType=1) then
      Begin
       MScores[Stage].NullErrorsP:=MScores[Stage].NullErrorsP+1;
       Form1.NullErrorsP.Caption:=IntToStr(MScores[Stage].NullErrorsP);
      End;
  Form1.Correct.Visible:=false;
  If MEnabled Then Form1.Incorrect.Visible:=true;
 END;
End;
if Stage=2 then
Begin
 If Both[GlobalCounter].Value=0 Then
 Begin
  EndupTime;
  Form1.Latency.Caption:=IntToStr(Diff);
  Scores2[GlobalCounter].FirstWordT:='Bigger';
  Scores2[GlobalCounter].SecondWordT:=Letter;
  Scores2[GlobalCounter].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
  Scores2[GlobalCounter].MLatency:=Diff;
  Scores2[GlobalCounter].It:=Iteration*(Form1.Sizer.Interval);
  Scores2[GlobalCounter].Value:=1;
  Scores2[GlobalCounter].KeyPressed:=1;
  Scores2[GlobalCounter].ReportSeen:='True Positive';
  Scores2[GlobalCounter].TargetType:=Both[GlobalCounter].TargetType;
  if Both[GlobalCounter].TargetType=0 then
    Begin
     MScores[Stage].MTotM:=MScores[Stage].MTotM+Diff;
     MScores[Stage].MScoreM:=MScores[Stage].MScoreM+1;
     Form1.KeyCounter.Caption:=IntToStr(MScores[Stage].MScoreM);
    End;
```

```
    if Both[GlobalCounter].TargetType=1 then
      Begin
        MScores[Stage].MTotP:=MScores[Stage].MTotP+Diff;
        MScores[Stage].MScoreP:=MScores[Stage].MScoreP+1;
        Form1.KeyCounterP.Caption:=IntToStr(MScores[Stage].MScoreP);
      End;
    If MEnabled
      Then Form1.Correct.Visible:=true;
    Form1.Incorrect.Visible:=false;
  End;
  If (Both[GlobalCounter].Value=1) Then // scores error response
  BEGIN
    EndupTime;
    Form1.ELatency.Caption:=IntToStr(Diff);
    If Both[GlobalCounter].Word<>'Null' Then
    Scores2[GlobalCounter].FirstWordT:='Smaller'
      Else
        Scores2[GlobalCounter].FirstWordT:='Null';
    Scores2[GlobalCounter].SecondWordT:=Letter;
//      Scores1[GlobalCounter].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
    Scores2[GlobalCounter].MElatency:=Diff;
    Scores2[GlobalCounter].EIt:=Iteration*(Form1.Sizer.Interval);
    Scores2[GlobalCounter].Value:=0;
    Scores2[GlobalCounter].KeyPressed:=1;
    Scores2[GlobalCounter].TimeE:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
    Scores2[GlobalCounter].TargetType:=Both[GlobalCounter].TargetType;
    If Both[GlobalCounter].Word<>'Null' Then
      Scores2[GlobalCounter].ReportSeen:='False Positive'
        Else
          Scores2[GlobalCounter].ReportSeen:='False Null';
    //Score M errors and increment the ELatency Array
    If (Both[GlobalCounter].Word<>'Null') AND (Both[GlobalCounter].TargetType=0) then
       Begin
         MScores[Stage].MErrorsM:=MScores[Stage].MErrorsM+1;
         Form1.ErrorNumber.Caption:=IntToStr(MScores[Stage].MErrorsM);
         MScores[Stage].ELatencyM:=MScores[Stage].ELatencyM+Diff;
       End;
    // Score M NullErrors
    IF (Both[GlobalCounter].Word='Null') AND (Both[GlobalCounter].TargetType=0) then
         Begin
           MScores[Stage].NullErrorsM:=MScores[Stage].NullErrorsM+1;
           Form1.NullErrorsM.Caption:=IntToStr(MScores[Stage].NullErrorsM);
         End;
   // Score P Errors and increment the ELatency Array
    If (Both[GlobalCounter].Word<>'Null') AND (Both[GlobalCounter].TargetType=1) Then
      Begin
        MScores[Stage].MErrorsP:=MScores[Stage].MErrorsP+1;
        Form1.ErrorNumberP.Caption:=IntToStr(MScores[Stage].MErrorsP);
        MScores[Stage].ELatencyP:=MScores[Stage].ELatencyP+Diff;
      End;
   // Score P Null Errors
    IF (Both[GlobalCounter].Word='Null') AND (Both[GlobalCounter].TargetType=1) then
         Begin
           MScores[Stage].NullErrorsP:=MScores[Stage].NullErrorsP+1;
```

```
          Form1.NullErrorsP.Caption:=IntToStr(MScores[Stage].NullErrorsP);
         End;
     Form1.Correct.Visible:=false;
     If MEnabled Then Form1.Incorrect.Visible:=true;
   END;
  End;
  if Stage=3 then
  Begin
   If Both[GlobalCounter].Value=0 Then
   Begin
     EndupTime;
     Form1.Latency.Caption:=IntToStr(Diff);
     Scores3[GlobalCounter].FirstWordT:='Bigger';
     Scores3[GlobalCounter].SecondWordT:=Letter;
     Scores3[GlobalCounter].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
     Scores3[GlobalCounter].MLatency:=Diff;
     Scores3[GlobalCounter].It:=Iteration*(Form1.Sizer.Interval);
     Scores3[GlobalCounter].Value:=1;
     Scores3[GlobalCounter].KeyPressed:=1;
     Scores3[GlobalCounter].ReportSeen:='True Positive';
     Scores3[GlobalCounter].TargetType:=Both[GlobalCounter].TargetType;
     if Both[GlobalCounter].TargetType=0 then
       Begin
         MScores[Stage].MTotM:=MScores[Stage].MTotM+Diff;
         MScores[Stage].MScoreM:=MScores[Stage].MScoreM+1;
         Form1.KeyCounter.Caption:=IntToStr(MScores[Stage].MScoreM);
       End;
     if Both[GlobalCounter].TargetType=1 then
       Begin
         MScores[Stage].MTotP:=MScores[Stage].MTotP+Diff;
         MScores[Stage].MScoreP:=MScores[Stage].MScoreP+1;
         Form1.KeyCounterP.Caption:=IntToStr(MScores[Stage].MScoreP);
       End;
     If MEnabled
       Then Form1.Correct.Visible:=true;
     Form1.Incorrect.Visible:=false;
   End;
   If (Both[GlobalCounter].Value=1) Then // scores error response
   BEGIN
     EndupTime;
     Form1.ELatency.Caption:=IntToStr(Diff);
     If Both[GlobalCounter].Word<>'Null' Then
     Scores3[GlobalCounter].FirstWordT:='Smaller'
       Else
         Scores3[GlobalCounter].FirstWordT:='Null';
     Scores3[GlobalCounter].SecondWordT:=Letter;
//     Scores1[GlobalCounter].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
     Scores3[GlobalCounter].MElatency:=Diff;
     Scores3[GlobalCounter].EIt:=Iteration*(Form1.Sizer.Interval);
     Scores3[GlobalCounter].Value:=0;
     Scores3[GlobalCounter].KeyPressed:=1;
     Scores3[GlobalCounter].TimeE:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
     Scores3[GlobalCounter].TargetType:=Both[GlobalCounter].TargetType;
     If Both[GlobalCounter].Word<>'Null' Then
```

```
    Scores3[GlobalCounter].ReportSeen:='False Positive'
     Else
      Scores3[GlobalCounter].ReportSeen:='False Null';
  //Score M errors and increment the ELatency Array
  If (Both[GlobalCounter].Word<>'Null') AND (Both[GlobalCounter].TargetType=0) then
    Begin
     MScores[Stage].MErrorsM:=MScores[Stage].MErrorsM+1;
     Form1.ErrorNumber.Caption:=IntToStr(MScores[Stage].MErrorsM);
     MScores[Stage].ELatencyM:=MScores[Stage].ELatencyM+Diff;
    End;
  // Score M NullErrors
  IF (Both[GlobalCounter].Word='Null') AND (Both[GlobalCounter].TargetType=0) then
      Begin
       MScores[Stage].NullErrorsM:=MScores[Stage].NullErrorsM+1;
       Form1.NullErrorsM.Caption:=IntToStr(MScores[Stage].NullErrorsM);
      End;
 // Score P Errors and increment the ELatency Array
  If (Both[GlobalCounter].Word<>'Null') AND (Both[GlobalCounter].TargetType=1) Then
    Begin
     MScores[Stage].MErrorsP:=MScores[Stage].MErrorsP+1;
     Form1.ErrorNumberP.Caption:=IntToStr(MScores[Stage].MErrorsP);
     MScores[Stage].ELatencyP:=MScores[Stage].ELatencyP+Diff;
    End;
 // Score P Null Errors
  IF (Both[GlobalCounter].Word='Null') AND (Both[GlobalCounter].TargetType=1) then
      Begin
       MScores[Stage].NullErrorsP:=MScores[Stage].NullErrorsP+1;
       Form1.NullErrorsP.Caption:=IntToStr(MScores[Stage].NullErrorsP);
      End;
  Form1.Correct.Visible:=false;
  If MEnabled Then Form1.Incorrect.Visible:=true;
 END;
End;
if Stage=4 then
Begin
 If Both[GlobalCounter].Value=0 Then
 Begin
  EndupTime;
  Form1.Latency.Caption:=IntToStr(Diff);
  Scores4[GlobalCounter].FirstWordT:='Bigger';
  Scores4[GlobalCounter].SecondWordT:=Letter;
  Scores4[GlobalCounter].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
  Scores4[GlobalCounter].MLatency:=Diff;
  Scores4[GlobalCounter].It:=Iteration*(Form1.Sizer.Interval);
  Scores4[GlobalCounter].Value:=1;
  Scores4[GlobalCounter].KeyPressed:=1;
  Scores4[GlobalCounter].ReportSeen:='True Positive';
  Scores4[GlobalCounter].TargetType:=Both[GlobalCounter].TargetType;
  if Both[GlobalCounter].TargetType=0 then
    Begin
     MScores[Stage].MTotM:=MScores[Stage].MTotM+Diff;
     MScores[Stage].MScoreM:=MScores[Stage].MScoreM+1;
     Form1.KeyCounter.Caption:=IntToStr(MScores[Stage].MScoreM);
    End;
```

```
      if Both[GlobalCounter].TargetType=1 then
        Begin
          MScores[Stage].MTotP:=MScores[Stage].MTotP+Diff;
          MScores[Stage].MScoreP:=MScores[Stage].MScoreP+1;
          Form1.KeyCounterP.Caption:=IntToStr(MScores[Stage].MScoreP);
        End;
      If MEnabled
        Then Form1.Correct.Visible:=true;
      Form1.Incorrect.Visible:=false;
    End;
    If (Both[GlobalCounter].Value=1) Then // scores error response
    BEGIN
      EndupTime;
      Form1.ELatency.Caption:=IntToStr(Diff);
      If Both[GlobalCounter].Word<>'Null' Then
      Scores4[GlobalCounter].FirstWordT:='Smaller'
        Else
          Scores4[GlobalCounter].FirstWordT:='Null';
      Scores4[GlobalCounter].SecondWordT:=Letter;
//      Scores1[GlobalCounter].TimeT:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
      Scores4[GlobalCounter].MElatency:=Diff;
      Scores4[GlobalCounter].EIt:=Iteration*(Form1.Sizer.Interval);
      Scores4[GlobalCounter].Value:=0;
      Scores4[GlobalCounter].KeyPressed:=1;
      Scores4[GlobalCounter].TimeE:=FormatDateTime('hh:nn:ss  ',StopTime-StartTime);
      Scores4[GlobalCounter].TargetType:=Both[GlobalCounter].TargetType;
      If Both[GlobalCounter].Word<>'Null' Then
        Scores4[GlobalCounter].ReportSeen:='False Positive'
          Else
            Scores4[GlobalCounter].ReportSeen:='False Null';
      //Score M errors and increment the ELatency Array
      If (Both[GlobalCounter].Word<>'Null') AND (Both[GlobalCounter].TargetType=0) then
         Begin
           MScores[Stage].MErrorsM:=MScores[Stage].MErrorsM+1;
           Form1.ErrorNumber.Caption:=IntToStr(MScores[Stage].MErrorsM);
           MScores[Stage].ELatencyM:=MScores[Stage].ELatencyM+Diff;
         End;
      // Score M NullErrors
      IF (Both[GlobalCounter].Word='Null') AND (Both[GlobalCounter].TargetType=0) then
           Begin
             MScores[Stage].NullErrorsM:=MScores[Stage].NullErrorsM+1;
             Form1.NullErrorsM.Caption:=IntToStr(MScores[Stage].NullErrorsM);
           End;
     // Score P Errors and increment the ELatency Array
      If (Both[GlobalCounter].Word<>'Null') AND (Both[GlobalCounter].TargetType=1) Then
        Begin
          MScores[Stage].MErrorsP:=MScores[Stage].MErrorsP+1;
          Form1.ErrorNumberP.Caption:=IntToStr(MScores[Stage].MErrorsP);
          MScores[Stage].ELatencyP:=MScores[Stage].ELatencyP+Diff;
        End;
     // Score P Null Errors
      IF (Both[GlobalCounter].Word='Null') AND (Both[GlobalCounter].TargetType=1) then
           Begin
             MScores[Stage].NullErrorsP:=MScores[Stage].NullErrorsP+1;
```

```
            Form1.NullErrorsP.Caption:=IntToStr(MScores[Stage].NullErrorsP);
           End;
       Form1.Correct.Visible:=false;
       If MEnabled Then Form1.Incorrect.Visible:=true;
     END;
   End;
   if Stage=1 then
   Begin
     With Form1.ADOMTable1 Do
       Begin
         Append;
         Fields.FieldByName('Stage').Value:=Stage;
         Fields.FieldByName('ID').Value:=NameCode;
         Fields.FieldByName('TimeT').Value:=Scores1[GlobalCounter].TimeT;
         Fields.FieldByName('Stimulus Type').Value:=Scores1[GlobalCounter].FirstWordT;
         Fields.FieldByName('Stimulus Item').Value:=Scores1[GlobalCounter].SecondWordT;
         iF Scores1[GlobalCounter].MLatency>0 Then Fields.FieldByName('Response
Latency').Value:=Scores1[GlobalCounter].MLatency;
         iF Scores1[GlobalCounter].MElatency>0 Then Fields.FieldByName('Error
Latency').Value:=Scores1[GlobalCounter].MElatency;
         Fields.FieldByName('Non Target').Value:=Scores1[GlobalCounter].FirstWordF;
         Fields.FieldByName('Non Target Item').Value:=Scores1[GlobalCounter].SecondWordF;
         Fields.FieldByName('TimeE').Value:=Scores1[GlobalCounter].TimeE;
         If IsSizer Then
           Begin
//           if True then ?
             If (Scores1[GlobalCounter].It<>0) Then
               Fields.FieldByName('Target Response Iteration  Latency').Value:=Scores1[GlobalCounter].It;
             If (Scores1[GlobalCounter].EIt<>0) Then
               Fields.FieldByName('Error Response Iteration Latency').Value:=Scores1[GlobalCounter].EIt;
           End;
         Fields.FieldByName('Value').Value:=Scores1[GlobalCounter].Value;
         Fields.FieldByName('Report Seen').Value:=Scores1[GlobalCounter].ReportSeen;
         Fields.FieldByName('Target type').Value:=Scores1[GlobalCounter].TargetType;
         Fields.FieldByName('Key Pressed').Value:=Scores1[GlobalCounter].KeyPressed;
         UpdateRecord;
         Post;
       End;
   End;
   if Stage=2 then
   Begin
     With Form1.ADOMTable1 Do
       Begin
         Append;
         Fields.FieldByName('Stage').Value:=Stage;
         Fields.FieldByName('ID').Value:=NameCode;
         Fields.FieldByName('TimeT').Value:=Scores2[GlobalCounter].TimeT;
         Fields.FieldByName('Stimulus Type').Value:=Scores2[GlobalCounter].FirstWordT;
         Fields.FieldByName('Stimulus Item').Value:=Scores2[GlobalCounter].SecondWordT;
         iF Scores2[GlobalCounter].MLatency>0 Then Fields.FieldByName('Response
Latency').Value:=Scores2[GlobalCounter].MLatency;
         iF Scores2[GlobalCounter].MElatency>0 Then Fields.FieldByName('Error
Latency').Value:=Scores2[GlobalCounter].MElatency;
         Fields.FieldByName('Non Target').Value:=Scores2[GlobalCounter].FirstWordF;
```

```
      Fields.FieldByName('Non Target Item').Value:=Scores2[GlobalCounter].SecondWordF;
      Fields.FieldByName('TimeE').Value:=Scores2[GlobalCounter].TimeE;
      Fields.FieldByName('TimeT').Value:=Scores2[GlobalCounter].TimeT;
      If IsSizer Then
        Begin
         if True then
          If (Scores2[GlobalCounter].It<>0) Then
            Fields.FieldByName('Target Response Iteration  Latency').Value:=Scores2[GlobalCounter].It;
          If (Scores2[GlobalCounter].EIt<>0) Then
            Fields.FieldByName('Error Response Iteration Latency').Value:=Scores2[GlobalCounter].EIt;
        End;
      Fields.FieldByName('Value').Value:=Scores2[GlobalCounter].Value;
      Fields.FieldByName('Report Seen').Value:=Scores2[GlobalCounter].ReportSeen;
      Fields.FieldByName('Target type').Value:=Scores2[GlobalCounter].TargetType;
      Fields.FieldByName('Key Pressed').Value:=Scores2[GlobalCounter].KeyPressed;
      UpdateRecord;
      Post;
    End;
  End;
  if Stage=3 then
  Begin
   With Form1.ADOMTable1 Do
    Begin
      Append;
      Fields.FieldByName('Stage').Value:=Stage;
      Fields.FieldByName('ID').Value:=NameCode;
      Fields.FieldByName('TimeT').Value:=Scores3[GlobalCounter].TimeT;
      Fields.FieldByName('Stimulus Type').Value:=Scores3[GlobalCounter].FirstWordT;
      Fields.FieldByName('Stimulus Item').Value:=Scores3[GlobalCounter].SecondWordT;
      iF Scores3[GlobalCounter].MLatency>0 Then Fields.FieldByName('Response
Latency').Value:=Scores3[GlobalCounter].MLatency;
      iF Scores3[GlobalCounter].MElatency>0 Then Fields.FieldByName('Error
Latency').Value:=Scores3[GlobalCounter].MElatency;
      Fields.FieldByName('Non Target').Value:=Scores3[GlobalCounter].FirstWordF;
      Fields.FieldByName('Non Target Item').Value:=Scores3[GlobalCounter].SecondWordF;
      Fields.FieldByName('TimeE').Value:=Scores3[GlobalCounter].TimeE;
      Fields.FieldByName('TimeT').Value:=Scores3[GlobalCounter].TimeT;
      If IsSizer Then
        Begin
         if True then
          If (Scores3[GlobalCounter].It<>0) Then
            Fields.FieldByName('Target Response Iteration  Latency').Value:=Scores3[GlobalCounter].It;
          If (Scores3[GlobalCounter].EIt<>0) Then
            Fields.FieldByName('Error Response Iteration Latency').Value:=Scores3[GlobalCounter].EIt;
        End;
      Fields.FieldByName('Value').Value:=Scores3[GlobalCounter].Value;
      Fields.FieldByName('Report Seen').Value:=Scores3[GlobalCounter].ReportSeen;
      Fields.FieldByName('Target type').Value:=Scores3[GlobalCounter].TargetType;
      Fields.FieldByName('Key Pressed').Value:=Scores3[GlobalCounter].KeyPressed;
      UpdateRecord;
      Post;
    End;
  End;
  if Stage=4 then
```

```
   Begin
     With Form1.ADOMTable1 Do
       Begin
         Append;
         Fields.FieldByName('Stage').Value:=Stage;
         Fields.FieldByName('ID').Value:=NameCode;
         Fields.FieldByName('TimeT').Value:=Scores4[GlobalCounter].TimeT;
         Fields.FieldByName('Stimulus Type').Value:=Scores4[GlobalCounter].FirstWordT;
         Fields.FieldByName('Stimulus Item').Value:=Scores4[GlobalCounter].SecondWordT;
         iF Scores4[GlobalCounter].MLatency>0 Then Fields.FieldByName('Response
Latency').Value:=Scores4[GlobalCounter].MLatency;
         iF Scores4[GlobalCounter].MElatency>0 Then Fields.FieldByName('Error
Latency').Value:=Scores4[GlobalCounter].MElatency;
         Fields.FieldByName('Non Target').Value:=Scores4[GlobalCounter].FirstWordF;
         Fields.FieldByName('Non Target Item').Value:=Scores4[GlobalCounter].SecondWordF;
         Fields.FieldByName('TimeE').Value:=Scores4[GlobalCounter].TimeE;
         Fields.FieldByName('TimeT').Value:=Scores4[GlobalCounter].TimeT;
           Begin
             if True then
               If (Scores4[GlobalCounter].It<>0) Then
                 Fields.FieldByName('Target Response Iteration  Latency').Value:=Scores4[GlobalCounter].It;
               If (Scores4[GlobalCounter].EIt<>0) Then
                 Fields.FieldByName('Error Response Iteration Latency').Value:=Scores4[GlobalCounter].EIt;
           End;
         Fields.FieldByName('Value').Value:=Scores4[GlobalCounter].Value;
         Fields.FieldByName('Report Seen').Value:=Scores4[GlobalCounter].ReportSeen;
         Fields.FieldByName('Target type').Value:=Scores4[GlobalCounter].TargetType;
         Fields.FieldByName('Key Pressed').Value:=Scores4[GlobalCounter].KeyPressed;
         UpdateRecord;
         Post;
       End;
   End;
End;

end.
```

Login

Database login

Dear Participant

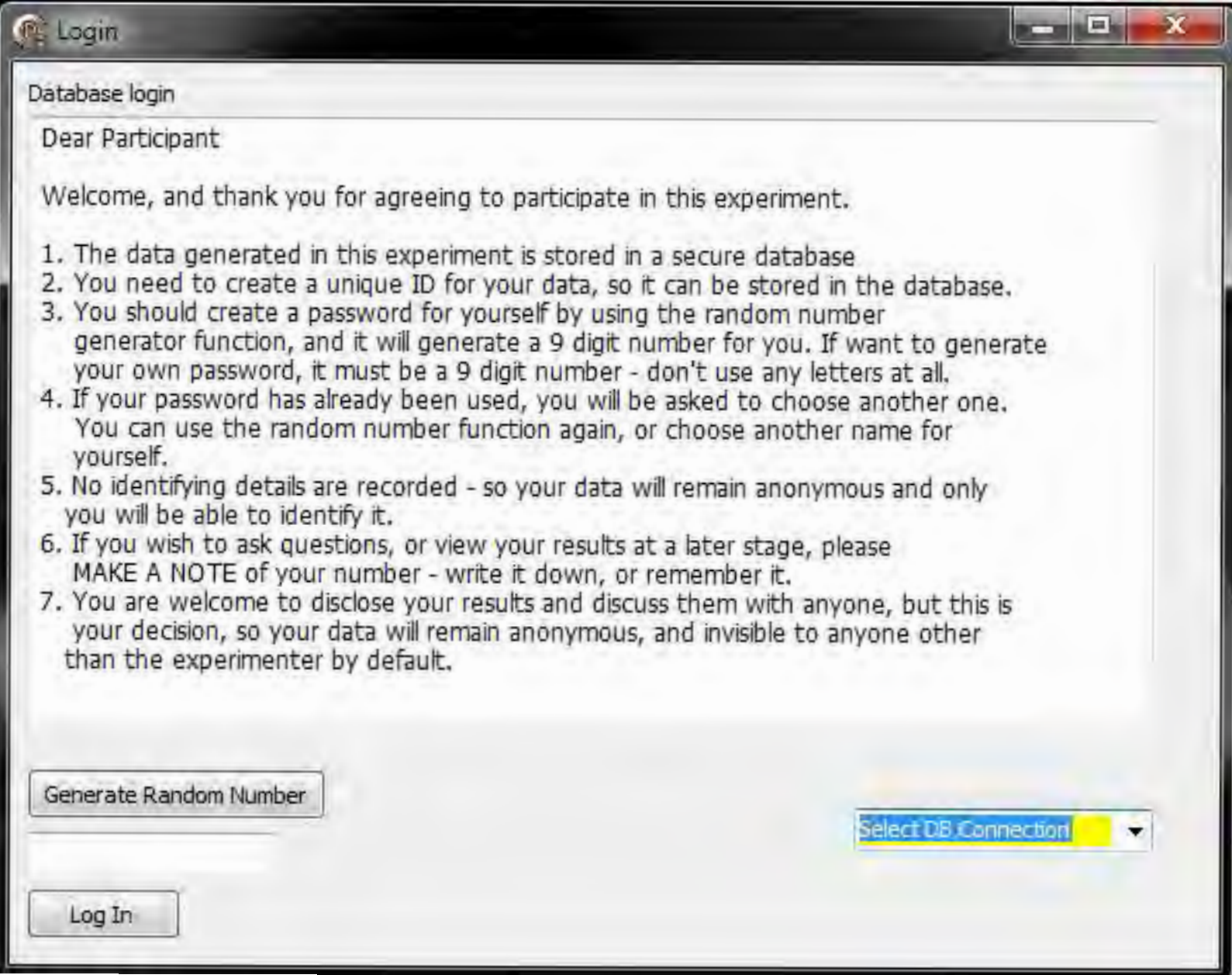
Welcome, and thank you for agreeing to participate in this experiment.

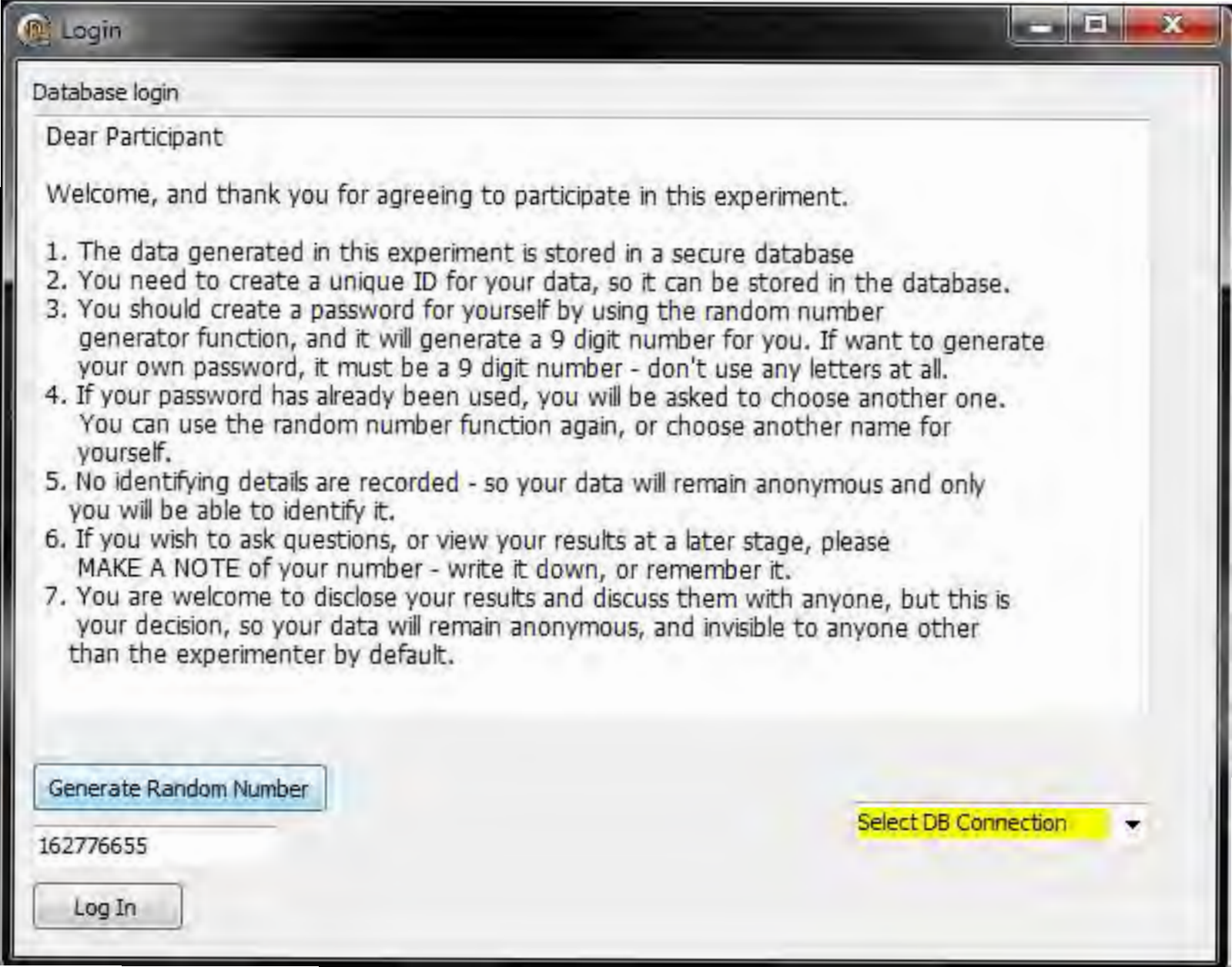
1. The data generated in this experiment is stored in a secure database
2. You need to create a unique ID for your data, so it can be stored in the database.
3. You should create a password for yourself by using the random number generator function, and it will generate a 9 digit number for you. If want to generate your own password, it must be a 9 digit number - don't use any letters at all.
4. If your password has already been used, you will be asked to choose another one. You can use the random number function again, or choose another name for yourself.
5. No identifying details are recorded - so your data will remain anonymous and only you will be able to identify it.
6. If you wish to ask questions, or view your results at a later stage, please MAKE A NOTE of your number - write it down, or remember it.
7. You are welcome to disclose your results and discuss them with anyone, but this is your decision, so your data will remain anonymous, and invisible to anyone other than the experimenter by default.

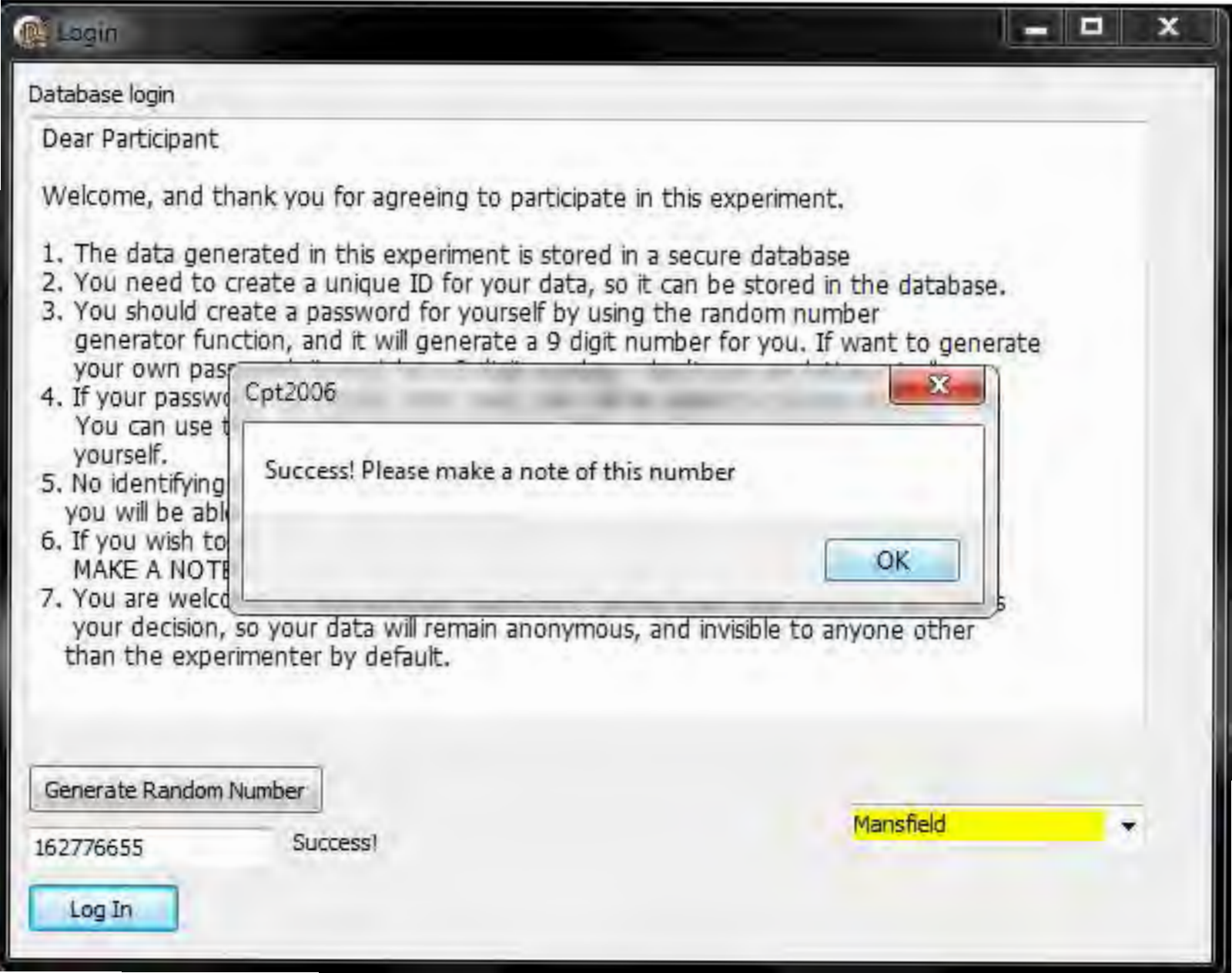
Generate Random Number

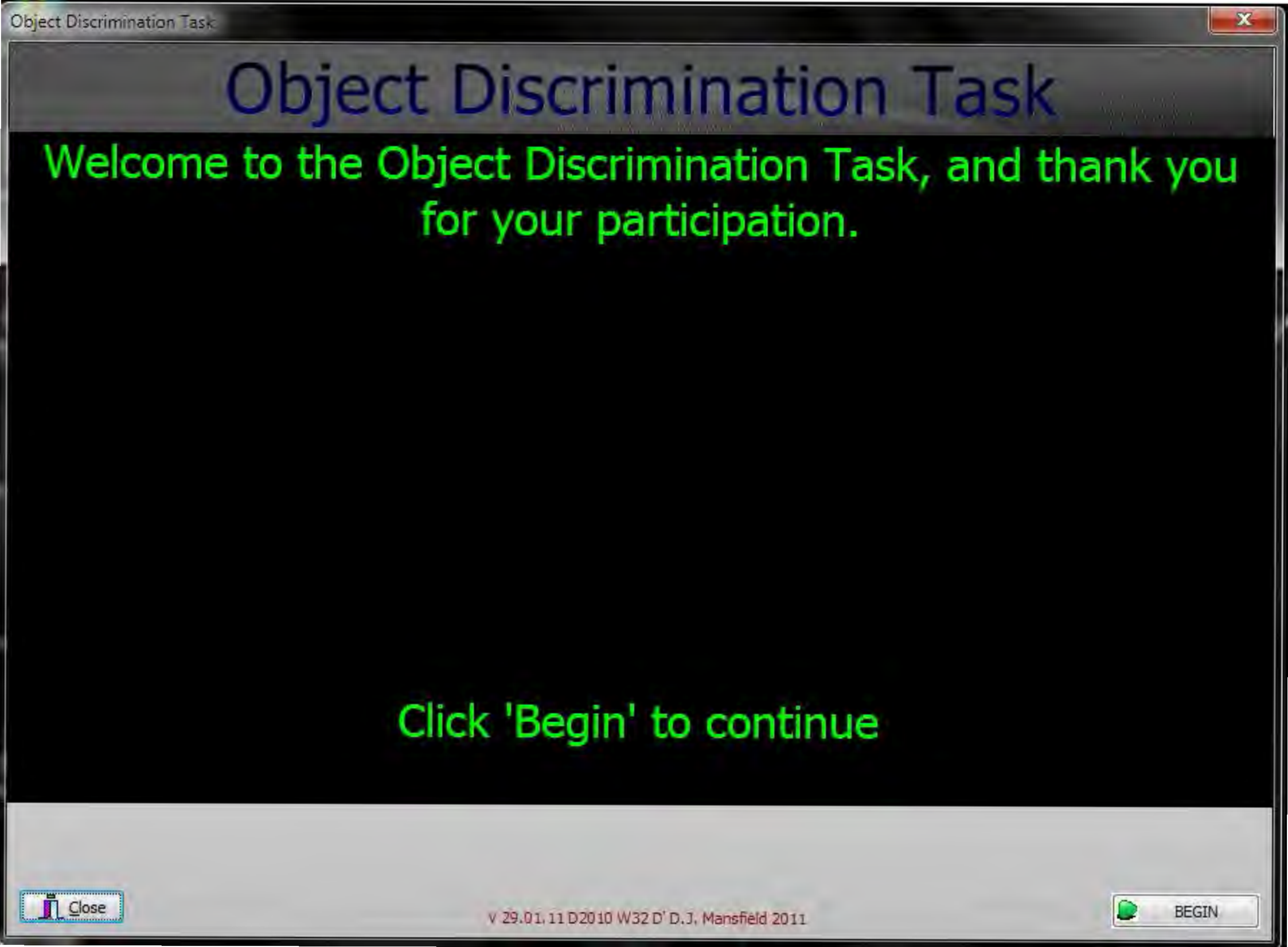
Select DB Connection

Log In


Login
Database login
Dear Participant
Welcome, and thank you for agreeing to participate in this experiment.
1. The data generated in this experiment is stored in a secure database
2. You need to create a unique ID for your data, so it can be stored in the database.
3. You should create a password for yourself by using the random number generator function, and it will generate a 9 digit number for you. If want to generate your own password, it must be a 9 digit number - don't use any letters at all.
4. If your password has already been used, you will be asked to choose another one. You can use the random number function again, or choose another name for yourself.
5. No identifying details are recorded - so your data will remain anonymous and only you will be able to identify it.
6. If you wish to ask questions, or view your results at a later stage, please MAKE A NOTE of your number - write it down, or remember it.
7. You are welcome to disclose your results and discuss them with anyone, but this is your decision, so your data will remain anonymous, and invisible to anyone other than the experimenter by default.
Generate Random Number
Select DB Connection
Log In


Login
Database login
Dear Participant
Welcome, and thank you for agreeing to participate in this experiment.
1. The data generated in this experiment is stored in a secure database
2. You need to create a unique ID for your data, so it can be stored in the database.
3. You should create a password for yourself by using the random number generator function, and it will generate a 9 digit number for you. If want to generate your own password, it must be a 9 digit number - don't use any letters at all.
4. If your password has already been used, you will be asked to choose another one. You can use the random number function again, or choose another name for yourself.
5. No identifying details are recorded - so your data will remain anonymous and only you will be able to identify it.
6. If you wish to ask questions, or view your results at a later stage, please MAKE A NOTE of your number - write it down, or remember it.
7. You are welcome to disclose your results and discuss them with anyone, but this is your decision, so your data will remain anonymous, and invisible to anyone other than the experimenter by default.
Generate Random Number
Select DB Connection
162776655
Log In


Login
Database login
Dear Participant
Welcome, and thank you for agreeing to participate in this experiment.
1. The data generated in this experiment is stored in a secure database
2. You need to create a unique ID for your data, so it can be stored in the database.
3. You should create a password for yourself by using the random number generator function, and it will generate a 9 digit number for you. If want to generate your own pas
4. If your passwo
You can use t
yourself.
5. No identifying
you will be abl
6. If you wish to
MAKE A NOTE
7. You are welco
your decision, so your data will remain anonymous, and invisible to anyone other than the experimenter by default.
Cpt2006
Success! Please make a note of this number
OK
Generate Random Number
162776655
Success!
Mansfield
Log In


Object Discrimination Task
Object Discrimination Task
Welcome to the Object Discrimination Task, and thank you for your participation.
Click 'Begin' to continue
Close
v 29.01.11 D2010 W32 D' D.J. Mansfield 2011
BEGIN

Object Discrimination Task

Introduction

The experiment and software has been designed by D.J. Mansfield as part of a PhD research project, and if you have any questions, please don't hesitate to contact me at the following email address: Mansfieldd@ukzn.ac.za

The experiment has several stages:

1. A colour vision screening assessment.

2. A brightness/contrast calibration procedure.

3. A colour calibration procedure.

4. An experiment in which pictures of everyday objects are presented in different colours and styles which you have to respond to and categorise by pressing a key.

There are 4 quite long blocks of trials in stage 4, so you do need to maintain your concentration. *You can take a short break between the blocks of trials in stage 4, but you will not be able to stop testing during any other stage in the experiment.*

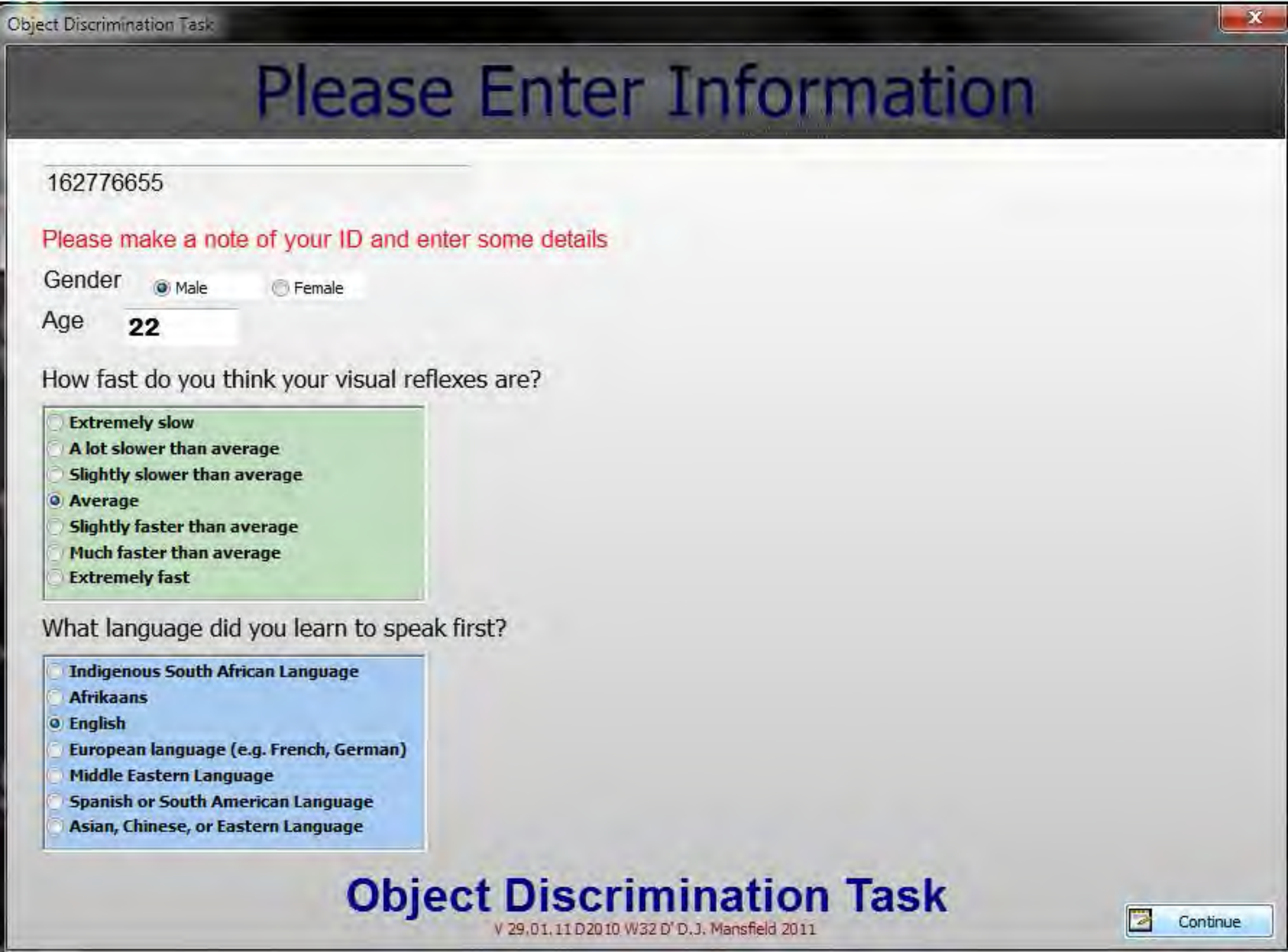
Close

Object Discrimination Task

V 29.01.11 D2010 W32 D' D.J. Mansfield 2011

Next


Object Discrimination Task
Click Information button
Object Discrimination Task
V 29.01.11 D2010 W32 D' D.J. Mansfield 2011
Close
Information


Object Discrimination Task
Please Enter Information
162776655
Please make a note of your ID and enter some details
Gender
Male
Female
Age
22
How fast do you think your visual reflexes are?
Extremely slow
A lot slower than average
Slightly slower than average
Average
Slightly faster than average
Much faster than average
Extremely fast
What language did you learn to speak first?
Indigenous South African Language
Afrikaans
English
European language (e.g. French, German)
Middle Eastern Language
Spanish or South American Language
Asian, Chinese, or Eastern Language
Object Discrimination Task
V 29.01.11 D2010 W32 D' D.J. Mansfield 2011
Continue


Object Discrimination Task
Click Vision Test Button
Object Discrimination Task
V 29.01.11 D2010 W32 D' D.J. Mansfield 2011
Vision Test

Object Discrimination Task

Colour Vision Test

This next procedure is a screening test for colour blindness.

Colour blindness is common - especially amongst men, and is not a serious disability.

The test is based on a public domain demonstration by Terence Waggoner (http://colorvisiontesting.com/ishihara.htm) using plates from the Ishihara Colour Vision Test.

It is not intended to be fully accurate, but will give some indications if you have a form of colour blindness.

The test may give you useful information about your colour vision ability, and is potentially important for this experiment.

Consent

Decline

Object Discrimination Task

V 29.01.11 D2010 W32 D' D.J. Mansfield 2011

Contrast Calibration

STAGE 2 Contrast Calibration INSTRUCTIONS

In this stage, you will be shown a series of pictures. Sometimes the pictures will be very difficult to see - the purpose of this phase is to determine the threshold at which the pictures are visible to you.

When you RECOGNISE the object in the picture PRESS A KEY (any key)

If you DO NOT RECOGNISE the object in the picture DO NOTHING, and just wait for the next one.

Sometimes no picture is presented, but just a blank - this is intentional.

The pictures are presented in various conditions - sometimes the details are lighter than the background, and sometimes they are darker.

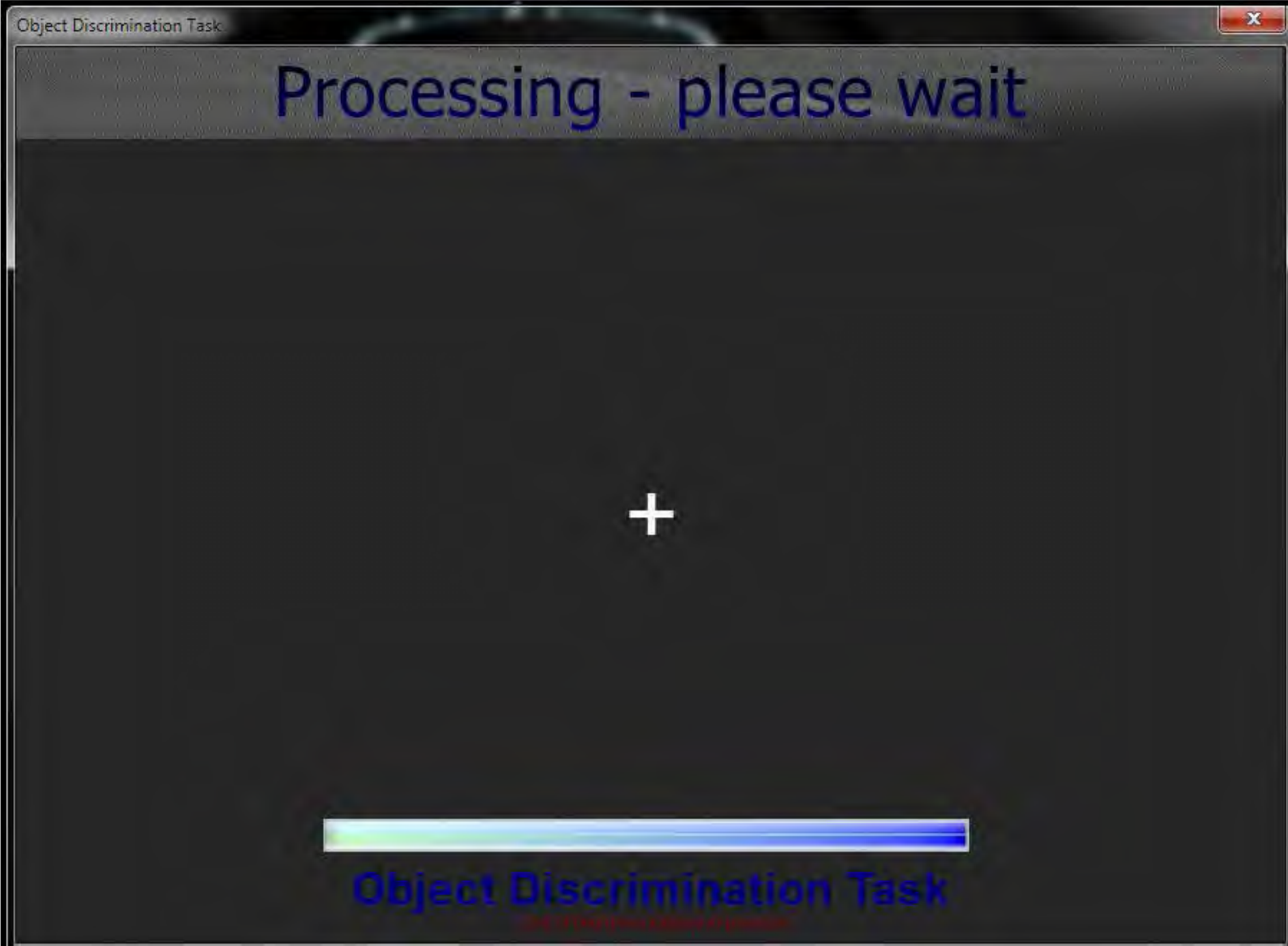
The important thing is to simply respond every time you recognise the object.

Click 'Calibrate Contrast' button to continue

Object Discrimination Task

V 29.01.11 D2010 W32 D' D.J. Mansfield 2011

Calibrate Contrast


Object Discrimination Task
Processing - please wait
Object Discrimination Task

Object Discrimination Task
Done!
Object Discrimination Task
Next Stage...

Object Discrimination Task

Colour Calibration

STAGE 3 Colour Calibration INSTRUCTIONS

In this stage, you will see a flashing character in the middle of the screen. The program cycles through different colour combinations, and frequencies. The purpose of this phase is to determine the level at which the **colours** appear to **merge**, instead of **flickering**.

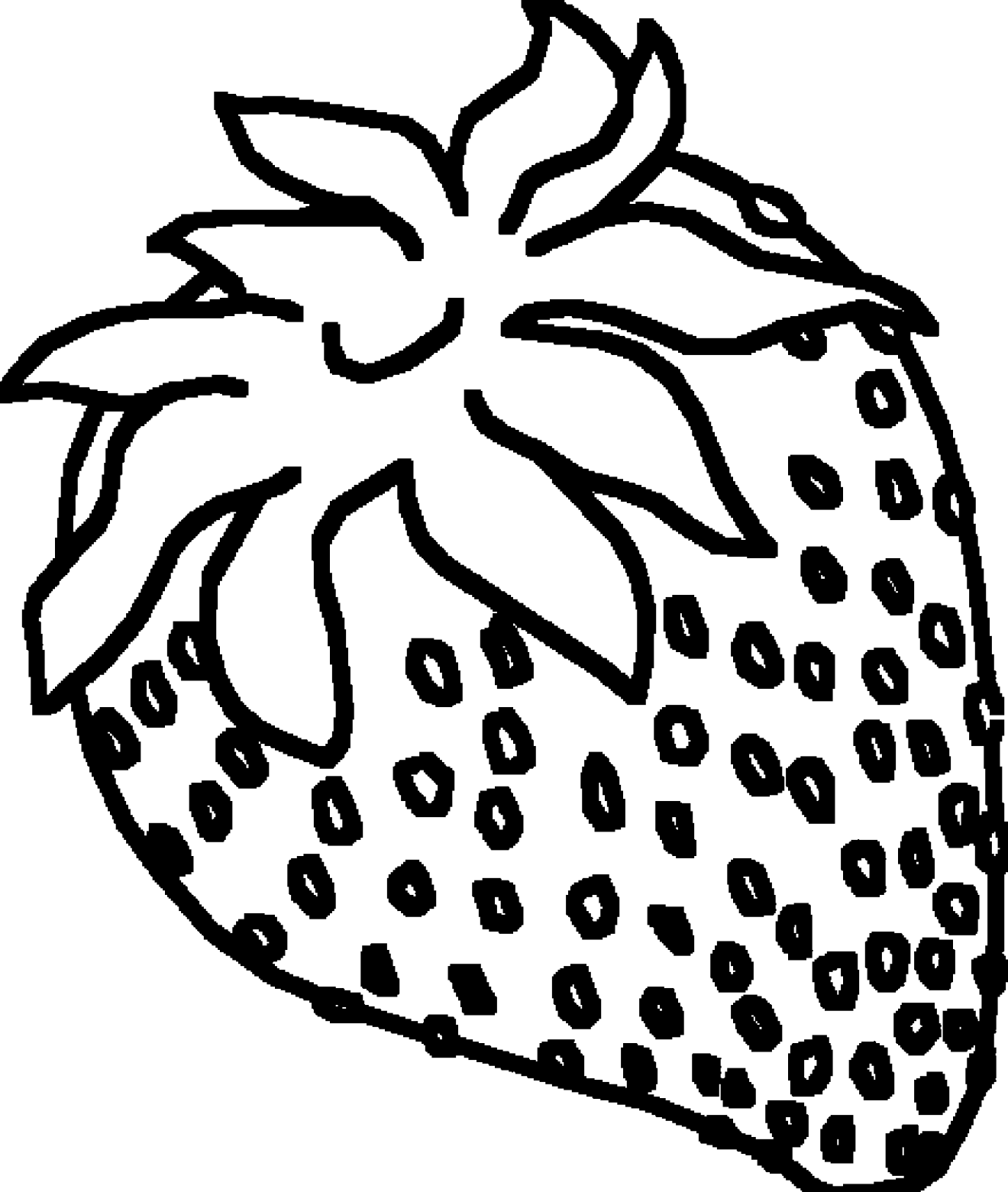
There are a large number of trials, and you should monitor the display and
- every time you notice that the flickering disappears (and the colours seem to merge)
PRESS A KEY (any key) **PRESS THE KEY AS MANY TIMES AS YOU SEE THE MERGING, BUT NOT MORE THAN ABOUT ONCE A SECOND.**

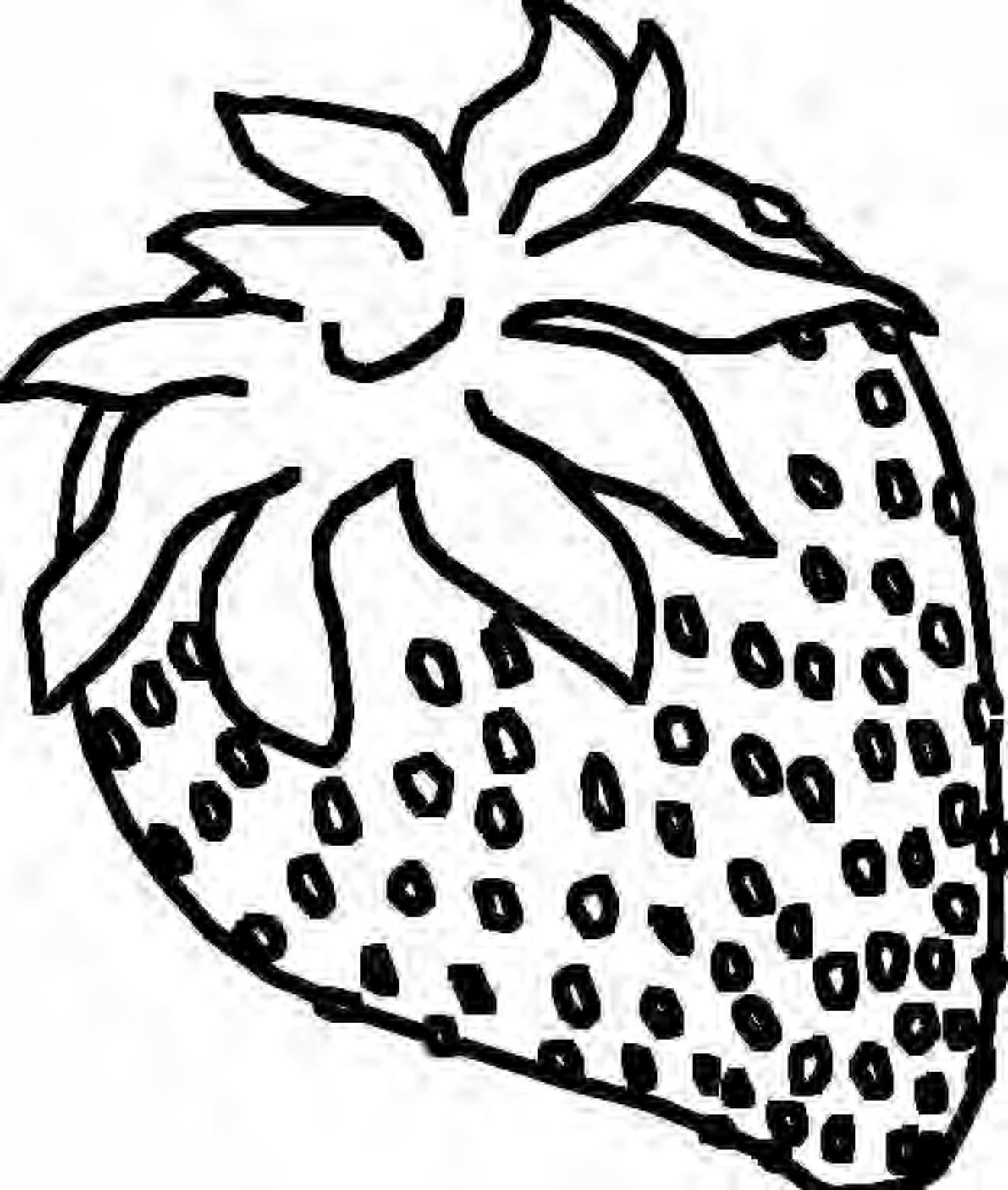
- You will find that some colour combinations cause the flickering to merge more than others. If the flickering is noticeable, **DO NOTHING,** until you notice that they merge in another colour combination or frequency.

- The flickering will not disappear completely, and there may be some 'visual noise'. You will see a count of your responses for each condition on a response matrix, displayed on the RHS of the screen.

Object Discrimination Task

Calibrate Colour





Simulator

BackGround

R G B Enabled

255 255 255

255 255 255

Choose Colour

ForeGround

R G B Enabled

0 5 50

0 255 0

Choose Colour

Colour Combinations

Snake

M Colours

Change Form Colour

Blue-Green

Files

Open File ReOpen Save File

Threaded Colour Convert Update File Location

Position

Vertical

Stretch

Size

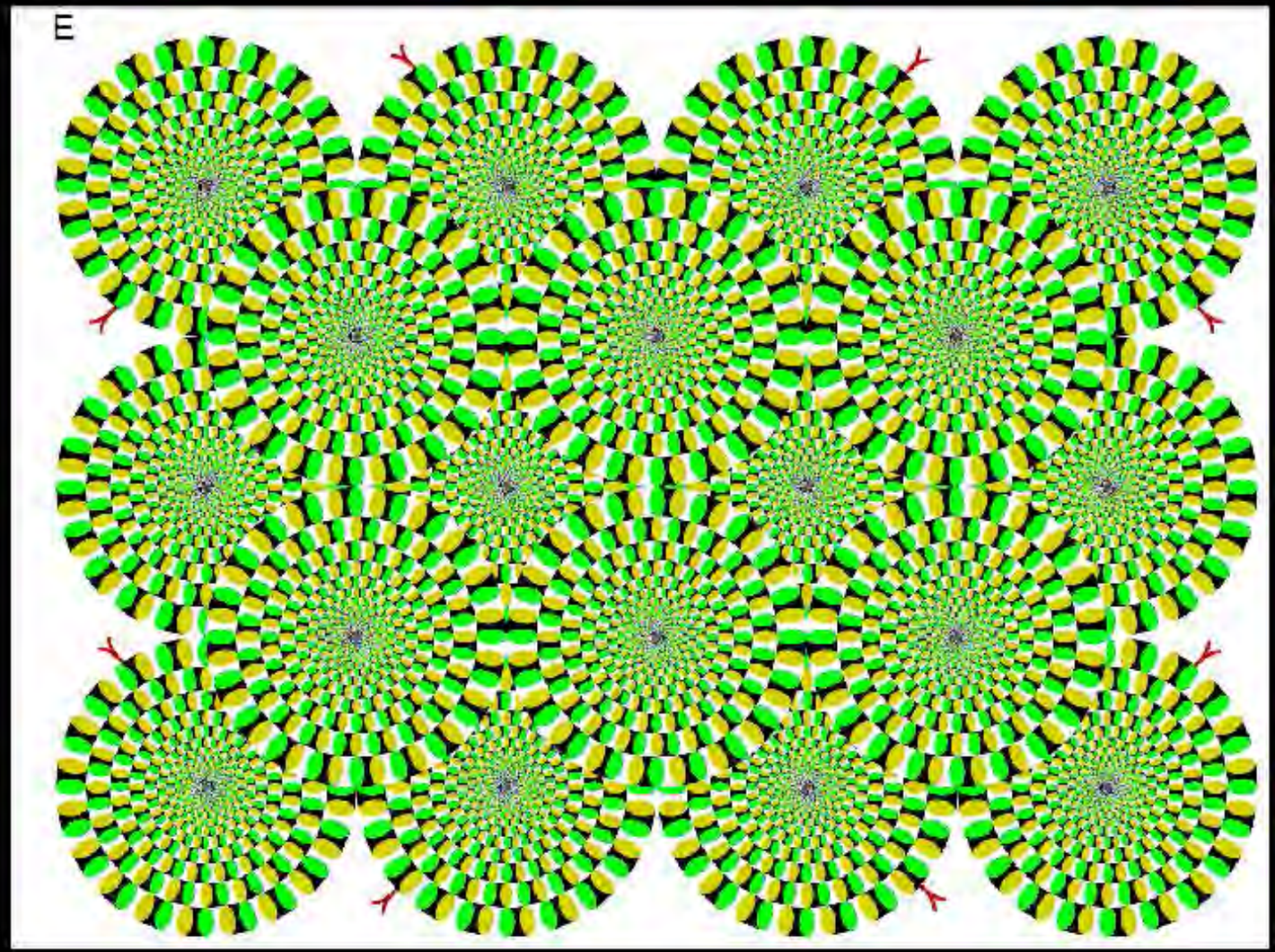
80

360

Horizontal

480

B 00 G FF R 00



```
unit Sorting;


interface
Uses Cpt,Sysutils;

Procedure SortAll(LTargets,LNon : Integer; PracTargets: TargetArray; PracNonTargets : TNonTargets;
Both : TBoth); External
'CPT2010SortingDll.dll';
Procedure SortPractice(LPracSensible,LPracSilly : Integer; PracSensible: TPracTargets;
PracNonT : TpracNonTargets; Pract : TPract); External
'CPT2010SortingDll.dll';
Procedure TrueFalse(Var Flag: Boolean); External
'TF.dll';
Procedure SortTargetType(Var Both : TBoth; Arraylength : Integer);

implementation

Procedure SortTargetType(Var Both : TBoth; Arraylength : Integer);
Var HowLong,LoopIdx,MaxM,MaxP : Integer;
Flag,MComplete,PComplete : Boolean;
Begin
  HowLong:=Arraylength Div 2;
  // Initialise all to 6
  for LoopIdx:=1 To ArrayLength do
    Begin
      Both[LoopIdx].TargetType:=6;
    End;
  MaxM:=0;
  MaxP:=0;
  MComplete:=False;
  PComplete:=False;
  for LoopIdx := 1 To ArrayLength do
    Begin
      if MaxM>Howlong then
        MComplete:=True;
      if MaxP>Howlong then
        PComplete:=True;
      TrueFalse(Flag);
      if Flag=True then
      Begin
        if Not(MComplete) then
        Begin
          Both[LoopIdx].TargetType:=0;
          MaxM:=MaxM+1;
        End;
      End;
      if Flag=False then
      Begin
        if Not(PComplete) then
        Begin
          Both[LoopIdx].TargetType:=1;
          MaxP:=MaxP+1;
```

```
      End;
    End;
  End;
  for LoopIdx:=1 To ArrayLength do
    Begin
      if Both[LoopIdx].TargetType=6 then
        Begin
          if MComplete then Both[LoopIdx].TargetType:=1;
          if PComplete then Both[LoopIdx].TargetType:=0;
        End;
    End;
End;


end.
```

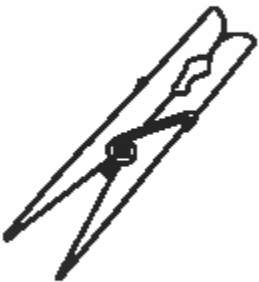
```
library CPT2010SortingDll;

{ Important note about DLL memory management: ShareMem must be the
  first unit in your library's USES clause AND your project's (select
  Project-View Source) USES clause if your DLL exports any procedures or
  functions that pass strings as parameters or function results. This
  applies to all strings passed to and from your DLL--even those that
  are nested in records and classes. ShareMem is the interface unit to
  the BORLNDMM.DLL shared memory manager, which must be deployed along
  with your DLL. To avoid using BORLNDMM.DLL, pass string information
  using PChar or ShortString parameters. }

uses
  SysUtils,
  Classes;

Type
 Temp50 = Record
 NonTarget: String[10];
 NonTValue: Integer;
 End;

 PracticeList = Record
 Word: String[10];
 Value: Integer;
 Timing : Integer;
 End;

  NonTargetList = Record
  NonTString : String[10];
  NonTargetV : Integer;
 End;

  Combined = Record
  Word: String[10];
  Value: Integer;
  TargetType : Integer;
 End;

 TargetRecord = Record
 TargetString : String[10];
 TargetValue : Integer;
End;

TTarget = Array Of TargetRecord;
TNonTarget = Array OF NonTargetList;
TBoth = Array Of Combined;
TTargetRec = Array Of String;
TNonTargetRec = Array Of String;
TPract = Array OF PracticeList;

Var
CheckArray100 : Array OF Boolean;
```

```
Checkarray150 : Array OF BOOLEAN;
CheckArray50 : Array OF Boolean;
Practice150 : Array OF Boolean;
TempNonTarget : Array OF Temp50;


{$R *.res}

Procedure GetRandom150(VAR Index : Integer; Arraylength : Integer);
BEGIN
REPEAT
Index := Random(Arraylength)+1;
UNTIL (Index >0) AND (Index <(ArrayLength+1));
END;

Procedure GetRandom50(VAR Index : Integer; LNonSense : Integer);
BEGIN
REPEAT
Index := Random(LNonSense)+1;
UNTIL (Index >0) AND (Index <(LNonSense+1));
END;

Procedure GetRandom30(VAR Index : Integer);
BEGIN
REPEAT
Index := Random(30)+1;
UNTIL (Index >0) AND (Index <31)
END;

Procedure GetRandom10(VAR Index : Integer);
BEGIN
REPEAT
Index := Random(10)+1;
UNTIL (Index >0) AND (Index <11);
END;

Procedure RD150(ArrayLength: Integer);
Var Loops: Integer;
Begin
SetLength (Checkarray150, ArrayLength+1);
 FOR Loops := 1 TO Arraylength DO
  BEGIN
   Checkarray150[Loops]:=False;
  END;
End;

Procedure RD100(LSense: Integer);
Var Loops: Integer;
Begin
SetLength(CheckArray100,LSense+1);
 FOR Loops := 1 TO LSense DO
  BEGIN
   Checkarray100[Loops]:=False;
  END;
```

```
End;

Procedure RD50(LNonSense: Integer);
Var
Loops : Integer;
BEgin
  SetLength(CheckArray50,LNonSense+1);
  For Loops:=1 To LNonSense+1 Do
  Begin
    CheckArray50[Loops]:=False;
  End;
End;


Procedure PD150(ArrayLength: Integer);
Var
Loops : Integer;
BEgin
  For Loops:=1 To ArrayLength Do
  Begin
    Practice150[Loops]:=False;
  End;
End;


Procedure SortPractice(LPracSense,LPracNonSense : Integer; PracSense: TTargetRec;
PracNonSense: TNonTargetRec; Pract : TPract);
Var
Index, Loops, Arraylength   : Integer;
Assigned : Boolean;
Begin
ArrayLength:=LPracSense+LPracNonSense;
SetLength(Practice150,Arraylength+1);
//SetLength(Pract,ArrayLength+1);
Pd150(ArrayLength);
FOR Loops := 1 TO LPracSense DO
   BEGIN
     Assigned := False;
     REPEAT
        GetRandom150(Index,ArrayLength);
        IF (Practice150[Index] = False) THEN
          BEGIN
            Pract[Index].Word := PracSense[Loops];
            If Pract[Index].Word='blank' then
              Pract[Index].Value:=0
              Else
                Pract[Index].Value:=1;
            Practice150[Index]:=True;
            Assigned:=True;
          END;
        UNTIL Assigned = True;
   END;
FOR Loops := 1 TO LPracNonSense DO
   BEGIN
```

```
    Assigned := False;
    REPEAT
      GetRandom150(Index,ArrayLength);
      IF (Practice150[Index] = False) THEN
        BEGIN
          Pract[Index].Word := PracNonSense[Loops];
          If Pract[Index].Word='blank' then
            Pract[Index].Value:=0
            Else
              Pract[Index].Value:=1;
          Practice150[Index]:=True;
          Assigned:=True;
        END;
      UNTIL Assigned = True;
  END;
End;

// NonSense array assigned into tempNonSense in random order
Procedure SortNonSense(LNonSense : Integer; Var NonSense: TNonTarget);
Var Loops,Index : Integer;
Assigned : Boolean;
Begin
RD50(LNonSense);
SetLength(TempNonTarget,LNonSense+1);
For Loops:= 1 To LNonSense Do
Begin
  Assigned:=false;
  REPEAT
    GetRandom50(Index,LNonSense);
     IF (Checkarray50[Index] = False) THEN
       BEGIN
         TempNonTarget[Index].NonTarget:= NonSense[Loops].NonTString;
         TempNonTarget[Index].NonTValue:= NonSense[Loops].NonTargetV;
         Checkarray50[Index]:= True;
         Assigned := True;
       END;
  UNTIL Assigned = True;
End;
// Chuck all the values back into the original array
Begin
  For Loops:=1 to LNonSense DO
  Begin
    NonSense[Loops].NonTString:= TempNonTarget[Loops].NonTarget;
    NonSense[Loops].NonTargetV:= TempNonTarget[Loops].NonTValue;
  End;
End;
End;

Procedure SortAll(LSense,LNonSense : Integer; Sense : TTarget; NonSense: TNonTarget;
Both : TBoth);
VAR
Index, Loops, Fifty, ArrayLength, Onefifty : Integer;
Assigned : Boolean;
BEGIN
```

```
Arraylength:=LSense+LNonSense;
SortNonSense(LNonSense,NonSense);
RD100(LSense);
RD150(ArrayLength);
//Setlength(Both,ArrayLength+1);
  FOR Loops := 1 TO LSense DO
    BEGIN
      Assigned := False;
      REPEAT
        GetRandom150(Index,ArrayLength);
          IF (Checkarray150[Index] = False) THEN
            BEGIN
              Both[Index].Word:= Sense[Loops].TargetString;
              Both[Index].Value:= Sense[Loops].TargetValue;
              Checkarray150[Index]:= True;
              Assigned := True;
            END;
       UNTIL Assigned = True;
    END;

  Fifty:=1;
  OneFifty:=1;

 Repeat
    If CheckArray150[OneFifty]=True then Onefifty:=OneFifty+1;
    If CheckArray150[OneFifty]=False Then
      Begin
        Both[OneFifty].Word:=NonSense[Fifty].NonTString;
        Both[OneFifty].Value:=NonSense[Fifty].NonTargetV;
        Fifty:=Fifty+1;
        OneFifty:=OneFifty+1;
        Assigned:=True;
       END;
  Until Fifty=LNonSense+1;
END;

Exports SortAll,SortPractice;
Begin
Randomize;
end.
```



```
unit Thread;

interface

uses
  Classes {$IFDEF MSWINDOWS} , Windows {$ENDIF}, CPT,Graphics,SysUtils,StrUtils,ExtCtrls;

type
  ImageThread = class(TThread)
  Private
  Var
  FR,FG,FB,ToR,ToG,ToB,FBR,FBG,FBB,ToBR,ToBG,ToBB : Integer;
  IsBlue,IsGreen : Boolean;
  Box1,Box2 : Boolean;
  FName : AnsiString;
  BitMap : TBitMap;
    type
      TMyPixelDescriptor = record
      Blue: Byte;
      Green: Byte;
      Red: Byte;
     end;
  PMyPixelArray = ^TMyPixelArray;
  TMyPixelArray = array[0..32767] of TMyPixelDescriptor;
  THMyPictureArray = array of TMyPixelArray;
  Procedure UpdatePicture;
  Procedure CallFile;
  procedure Execute; override;
  protected
  Public
  Constructor Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean; Filename : ShortString);
  end;

type
  ImageThreadM = class(TThread)
  Private
  Var
  FR,FG,FB,ToR,ToG,ToB,FBR,FBG,FBB,ToBR,ToBG,ToBB : Integer;
  IsBlue,IsGreen : Boolean;
  Box1,Box2 : Boolean;
  FName : AnsiString;
  BitMap : TBitMap;
    type
      TMyPixelDescriptor = record
      Blue: Byte;
      Green: Byte;
      Red: Byte;
     end;
  PMyPixelArray = ^TMyPixelArray;
  TMyPixelArray = array[0..32767] of TMyPixelDescriptor;
  THMyPictureArray = array of TMyPixelArray;
  Procedure UpdatePictureM;
```

```
  Procedure CallFileM;
  procedure Execute; override;
  protected
  Public
  Constructor Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean; Filename : ShortString);
  Var MStream : TMemoryStream;
  end;

type
  ImageThreadAllM = class(TThread)
  Private
  Var
  FR,FG,FB,ToR,ToG,ToB,FBR,FBG,FBB,ToBR,ToBG,ToBB : Integer;
  IsBlue,IsGreen : Boolean;
  Box1,Box2 : Boolean;
  FRName,TRName,FFileList : AnsiString;
  BitMapAll : TBitMap;
    type
      TMyPixelDescriptor = record
      Blue: Byte;
      Green: Byte;
      Red: Byte;
     end;
  PMyPixelArray = ^TMyPixelArray;
  TMyPixelArray = array[0..32767] of TMyPixelDescriptor;
  THMyPictureArray = array of TMyPixelArray;
  Procedure UpdateForm;
  procedure Execute; override;
  protected
    BitMapList : TStrings;
  Public
  Constructor Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean; Filename1,Filename2,FileList : ShortString);
  end;

  type
  ImageThreadAllP = class(TThread)
  Private
  Var
  FR,FG,FB,ToR,ToG,ToB,FBR,FBG,FBB,ToBR,ToBG,ToBB : Integer;
  IsBlue,IsGreen : Boolean;
  Box1,Box2 : Boolean;
  FRName,TRName,FFileList : AnsiString;
  BitMapAll : TBitMap;
    type
      TMyPixelDescriptor = record
      Blue: Byte;
      Green: Byte;
      Red: Byte;
     end;
  PMyPixelArray = ^TMyPixelArray;
  TMyPixelArray = array[0..32767] of TMyPixelDescriptor;
  THMyPictureArray = array of TMyPixelArray;
```

```
  Procedure UpdateForm;
  procedure Execute; override;
  protected
    BitMapList : TStrings;
  Public
  Constructor Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean; Filename1,Filename2,FileList : ShortString);
  end;


implementation

Procedure ImageThreadM.CallFileM;
begin
  BitMap:=TBitMap.Create;
  BitMap.LoadFromFile(FName);
 end;

Procedure ImageThread.CallFile;
begin
  BitMap:=TBitMap.Create;
  BitMap.LoadFromFile(FName);
end;

Procedure ImageThreadAllM.UpdateForm;
begin
   Form1.BtnPInstructions.Visible:=True;
   Form1.Banner1.Caption.Text:='Done!';
   Form1.Label1.Caption:='Next Stage...';
   Form1.ProgressBar1.Position:=0;
   Form1.ProgressBar1.Visible:=False;
   Cpt.Phase:='CP';
   BitmapAll.Free;
   Form1.LblR.Visible:=False;
   Form1.LblG.Visible:=False;
   Form1.LblB.Visible:=False;
   Form1.LblR1.Visible:=False;
   Form1.LblG1.Visible:=False;
   Form1.LblB1.Visible:=False;
   Form1.UpdateCursor;
end;

Procedure ImageThreadAllP.UpdateForm;
begin
   Form1.BtnStartTest.visible:=False;
   Form1.BtnTestInstructions.Visible:=True;
   Form1.Banner1.Caption.Text:='Done!';
   Form1.Label1.Caption:='Next Stage...';
   Form1.ProgressBar1.Visible:=False;
   Form1.LblFlickerHz.Visible:=false;
   Form1.Stimulus.visible:=false;
   Form1.MatrixPanel.Visible:=false;
   BitmapAll.Free;
   Form1.LblR.Visible:=False;
```

```
   Form1.LblG.Visible:=False;
   Form1.LblB.Visible:=False;
   Form1.LblR1.Visible:=False;
   Form1.LblG1.Visible:=False;
   Form1.LblB1.Visible:=False;
   Form1.UpdateCursor;
end;

constructor ImageThreadAllP.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean;Filename1,Filename2,FileList : ShortString);
Begin
  FR:=AFR;
  FG:=AFG;
  FB:=AFB;
  ToR:=AToR;  //AToR:=DR;
  ToG:=AToG;  //AToG:=DG;
  ToB:=AToB;  //AToB:=DB;
  FBR:=AFBR;  // AFBR:=FSR;
  FBG:=AFBG;  //AFBG:=FSG;
  FBB:=AFBB;  //AFBB:=FSB;
  ToBR:=AToBR;
  ToBG:=AToBG;
  ToBB:=AToBB;
  Box1:=ABox1;
  Box2:=ABox2;
  FRName := FileName1;
  TRName := FileName2;
  FFileList:=FileList;
  FreeOnTerminate:=True;
  Inherited Create(False);
End;

Procedure ImageThreadAllP.Execute;
Var
  p : PMyPixelArray;
  x : Integer;
  y : Integer;
  R,G,B,Idx : Integer;
  FromFileName,ToFileName : AnsiString;
begin
  BitMapList:=TStringList.Create;
  BitMapList.LoadFromFile(FFileList);
  BitMapAll:=TBitMap.Create;
  for Idx := 0 to BitMapList.Count - 1 do
  Begin
  FromFilename:=FRName+BitMapList[IdX];
  ToFilename:=TRName+BitMapList[IdX];
  BitMapAll.LoadFromFile(FromFilename);
  for Y := 0 to BitMapAll.Height - 1 do
  Begin
    p:= BitMapAll.ScanLine[y];
    for x := 0 to BitMapAll.Width - 1 do
     Begin
       With p[x] do
```

```
      Begin
        R:=Red;
        G:=Green;
        B:=Blue;
        if Box2  then
         if ((R>=FBR) AND (G>=FBG) AND (B>=FBB))then //is it Lighter?
          Begin
            Red:=ToBR;
            Green:=ToBG;
            Blue:=ToBB;
          End;
        if Box1 then
          if ((R<=FR) AND (G<=FG) AND (B<=FB)) then  //is it darker?
          Begin
            Green:=ToG;
            Red:=ToR;
            Blue:=ToB;
          End;
      End;  // with p[x]
    End;
  End;
  BitMapAll.SaveToFile(ToFilename);
  End;
  if Terminated  then Exit;
  Synchronize(UpdateForm);
end;

constructor ImageThreadAllM.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean;Filename1,Filename2,FileList : ShortString);
Begin
  FR:=AFR;
  FG:=AFG;
  FB:=AFB;
  ToR:=AToR;  //AToR:=DR;
  ToG:=AToG;  //AToG:=DG;
  ToB:=AToB;  //AToB:=DB;
  FBR:=AFBR;  // AFBR:=FSR;
  FBG:=AFBG;  //AFBG:=FSG;
  FBB:=AFBB;  //AFBB:=FSB;
  ToBR:=AToBR;
  ToBG:=AToBG;
  ToBB:=AToBB;
  Box1:=ABox1;
  Box2:=ABox2;
  FRName := FileName1;
  TRName := FileName2;
  FFileList:=FileList;
  FreeOnTerminate:=True;
  Inherited Create(False);
End;

Procedure ImageThreadAllM.Execute;
Var
  p : PMyPixelArray;
```

```
  x : Integer;
  y : Integer;
  R,G,B,Idx : Integer;
  FromFileName,ToFileName : AnsiString;
begin
  BitMapList:=TStringList.Create;
  BitMapList.LoadFromFile(FFileList);
  BitMapAll:=TBitMap.Create;
  for Idx := 0 to BitMapList.Count - 1 do
  Begin
  FromFilename:=FRName+BitMapList[IdX];
  ToFilename:=TRName+BitMapList[IdX];
  BitMapAll.LoadFromFile(FromFilename);
  for Y := 0 to BitMapAll.Height - 1 do
  Begin
    p:= BitMapAll.ScanLine[y];
    for x := 0 to BitMapAll.Width - 1 do
     Begin
       With p[x] do
       Begin
         R:=Red;
         G:=Green;
         B:=Blue;
         if Box2  then
          if ((R>=FBR) AND (G>=FBG) AND (B>=FBB))then //is it Lighter?
           Begin
             Red:=ToBR;
             Green:=ToBG;
             Blue:=ToBB;
           End;
         if Box1 then
           if ((R<=FR) AND (G<=FG) AND (B<=FB)) then  //is it darker?
           Begin
             Green:=ToG;
             Red:=ToR;
             Blue:=ToB;
           End;
       End;  // with p[x]
     End;
  End;
  BitMapAll.SaveToFile(ToFilename);
  End;
  if Terminated  then Exit;
  Synchronize(UpdateForm);
end;

constructor ImageThreadM.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean;Filename : ShortString);
Begin
  FR:= AFR;
  FG:=AFG;
  FB:=AFB;
  ToR:=AToR;
  ToG:=AToG;
```

```
  ToB:=AToB;
  FBR:=AFBR;
  FBG:=AFBG;
  FBB:=AFBB;
  ToBR:=AToBR;
  ToBG:=AToBG;
  ToBB:=AToBB;
  Box1:=ABox1;
  Box2:=ABox2;
  FName := FileName;
  FreeOnTerminate:=True;
  Inherited Create(False);
End;

constructor ImageThread.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean; Filename : ShortString);
Begin
  FR:= AFR;
  FG:=AFG;
  FB:=AFB;
  ToR:=AToR;
  ToG:=AToG;
  ToB:=AToB;
  FBR:=AFBR;
  FBG:=AFBG;
  FBB:=AFBB;
  ToBR:=AToBR;
  ToBG:=AToBG;
  ToBB:=AToBB;
  Box1:=ABox1;
  Box2:=ABox2;
  FName := FileName;
  FreeOnTerminate:=True;
  Inherited Create(False);
End;

Procedure ImageThreadM.UpdatePictureM;
begin
  BitMap.SaveToStream(MStream);
  BitMap.SaveToStream(CPT.MStream);
  Cpt.MStream.LoadFromStream(MStream);
  MStream.free;
  Bitmap.Free;
  Form1.Fixation.Enabled:=True;
end;

Procedure ImageThreadM.Execute;
Var
  p : PMyPixelArray;
  x : Integer;
  y : Integer;
  R,G,B : Integer;
begin
  BitMap:=TBitMap.Create;
```

```
  BitMap.LoadFromFile(FName);
  for Y := 0 to BitMap.Height - 1 do
  Begin
    p:= BitMap.ScanLine[y];
    for x := 0 to BitMap.Width - 1 do
     Begin
       With p[x] do
       Begin
         R:=Red;
         G:=Green;
         B:=Blue;
         if Box2  then
          if ((R>=FBR) AND (G>=FBG) AND (B>=FBB))then //is it Lighter?
           Begin
             Red:=ToBR;
             Green:=ToBG;
             Blue:=ToBB;
           End;
         if Box1 then
           if ((R<=FR) AND (G<=FG) AND (B<=FB)) then  //is it darker?
           Begin
             Green:=ToG;
             Red:=ToR;
             Blue:=ToB;
           End;
       End;  // with p[x]
     End;
  End;
  MStream:=TMemoryStream.Create;
  BitMap.SaveToStream(MStream);
  if Terminated  then Exit;
  Synchronize(UpdatePictureM);
end;


Procedure ImageThread.Execute;
Var
  p : PMyPixelArray;
  x : Integer;
  y : Integer;
  R,G,B : Integer;
begin
  BitMap:=TBitMap.Create;
  BitMap.LoadFromFile(FName);
  for Y := 0 to BitMap.Height - 1 do
  Begin
    p:= BitMap.ScanLine[y];
    for x := 0 to BitMap.Width - 1 do
     Begin
       With p[x] do
       Begin
         R:=Red;
         G:=Green;
         B:=Blue;
```

```
        if Box2  then
         if ((R<=FBR) AND (G>=FBG) AND (B>=FBB))then //is it blue
          Begin
            Red:=ToBR;
            Green:=ToBG;
            Blue:=ToBB;
          End;
        if Box1 then
          if ((R<=FR) AND (G>=FG) AND (B<=FB)) then  //is it green?
          Begin
            Green:=ToG;
            Red:=ToR;
            Blue:=ToB;
          End;
      End;  // with p[x]
    End;
  End;
  if Terminated  then Exit;
  Synchronize(UpdatePicture);
end;

Procedure ImageThread.UpdatePicture;
begin
  AppendStr(FName,'1');
  BitMap.SaveToFile(FName);
  Bitmap.Free;
end;

end.
```

```
unit Thread;

interface

uses
  Classes {$IFDEF MSWINDOWS}, Windows {$ENDIF}, Unit1, SysUtils,StrUtils, FMX.Types;//, Vcl.Graphics;

type
  ImageThread = class(TThread)
  Private
  Var
  FR,FG,FB,ToR,ToG,ToB,FBR,FBG,FBB,ToBR,ToBG,ToBB : Integer;
  IsBlue,IsGreen : Boolean;
  Box1,Box2 : Boolean;
  FName : AnsiString;
  BitMap : TBitMap;
  I: Integer;
  T: Byte;
  PixelRecs: PAlphaColorRecArray;
  Procedure UpdatePicture;
  Procedure CallFile;
  procedure Execute; override;
  protected
  Public
  Constructor Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean; Filename : ShortString);
  end;

type
  ImageThreadM = class(TThread)
  Private
  Var
  FR,FG,FB,ToR,ToG,ToB,FBR,FBG,FBB,ToBR,ToBG,ToBB : Integer;
  IsBlue,IsGreen : Boolean;
  Box1,Box2 : Boolean;
  FName : AnsiString;
  BitMap : TBitMap;
    type
      TMyPixelDescriptor = record
      Blue: Byte;
      Green: Byte;
      Red: Byte;
     end;
  PMyPixelArray = ^TMyPixelArray;
  TMyPixelArray = array[0..32767] of TMyPixelDescriptor;
  THMyPictureArray = array of TMyPixelArray;
  Procedure UpdatePictureM;
  Procedure CallFileM;
  procedure Execute; override;
  protected
  Public
  Constructor Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean; Filename : ShortString);
```

```
  end;


implementation

Procedure ImageThreadM.CallFileM;
begin
  BitMap:=TBitMap.CreateFromFile(FName);
  BitMap.LoadFromFile(FName);
end;

Procedure ImageThread.CallFile;
begin
  BitMap:=TBitMap.CreateFromFile(FName);
  BitMap.LoadFromFile(FName);
end;

constructor ImageThreadM.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean;Filename : ShortString);
Begin
  FR:= AFR;
  FG:=AFG;
  FB:=AFB;
  ToR:=AToR;
  ToG:=AToG;
  ToB:=AToB;
  FBR:=AFBR;
  FBG:=AFBG;
  FBB:=AFBB;
  ToBR:=AToBR;
  ToBG:=AToBG;
  ToBB:=AToBB;
  Box1:=ABox1;
  Box2:=ABox2;
  FName := FileName;
  FreeOnTerminate:=True;
  Inherited Create(False);
End;

constructor ImageThread.Create(AFR,AFG,AFB,AToR,AToG,AToB,AFBR,AFBG,AFBB,AToBR,AToBG,
  AToBB : Integer; ABox1,ABox2 : Boolean; Filename : ShortString);
Begin
  FR:= AFR;
  FG:=AFG;
  FB:=AFB;
  ToR:=AToR;
  ToG:=AToG;
  ToB:=AToB;
  FBR:=AFBR;
  FBG:=AFBG;
  FBB:=AFBB;
  ToBR:=AToBR;
  ToBG:=AToBG;
  ToBB:=AToBB;
```

```
  Box1:=ABox1;
  Box2:=ABox2;
  FName := FileName;
  FreeOnTerminate:=True;
  Inherited Create(False);
End;

Procedure ImageThreadM.UpdatePictureM;
begin
  Form1.Image1.Bitmap.Canvas.CreateFromBitmap(BitMap);
  AppendStr(FName,'1');
  Bitmap.Free;
end;

Procedure ImageThreadM.Execute;
Var
  PixelRecs: PAlphaColorRecArray;
  I : Integer;
  p : PMyPixelArray;
  x : Integer;
  y : Integer;
  T : Byte;
begin
  BitMap:=TBitMap.CreateFromFile(FName);
  BitMap.LoadFromFile(FName);
  PixelRecs := PAlphaColorRecArray(BitMap.StartLine);
for I := 0 to Bitmap.Width * Bitmap.Height - 1 do
  Begin
    PixelRecs := PAlphaColorRecArray(Bitmap.StartLine);
      with PixelRecs[I] do
        Begin
          if Box2  then
           if ((R<=FBR) AND (G<=FBG) AND (B<=FBB))then //is it Lighter?
            Begin
              R:=ToBR;
              G:=ToBG;
              B:=ToBB;
            End;
          if Box1 then
            if ((R<=FR) AND (G<=FG) AND (B<=FB)) then  //is it darker?
            Begin
              G:=ToG;
              R:=ToR;
              B:=ToB;
            End;
        End;  // with p[x]
     End;
  if Terminated  then Exit;
  Synchronize(UpdatePictureM);
end;


Procedure ImageThread.Execute;
Var
```

```
  PixelRecs: PAlphaColorRecArray;
  SourceLine, TargetLine: PAlphaColorArray;
  I : Integer;
  p : PMyPixelArray;
  x : Integer;
  y : Integer;
  R,G,B : Integer;
begin
  BitMap:=TBitMap.Create(Form1.Image1.Bitmap.Width,Form1.Image1.Bitmap.Height);
    Bitmap.Assign(Form1.Image1.Bitmap);
  for I := 0 to Bitmap.Width * Bitmap.Height - 1 do
  Begin
    PixelRecs := PAlphaColorRecArray(Bitmap.StartLine);
       with PixelRecs[I] do
        Begin
          if Box2  then
           if ((R<=FBR) AND (G>=FBG) AND (B>=FBB))then //is it blue
            Begin
              R:=ToBR;
              G:=ToBG;
              B:=ToBB;
            End;
          if Box1 then
            if ((R<=FR) AND (G>=FG) AND (B<=FB)) then  //is it green?
            Begin
              G:=ToG;
              R:=ToR;
              B:=ToB;
            End;
        End;
  End;
  if Terminated  then Exit;
  Synchronize(UpdatePicture);
end;

Procedure ImageThread.UpdatePicture;
begin
  Form1.Image1.Visible:=False;
  AppendStr(FName,'1');
  Form1.ImageViewer1.Bitmap:=Bitmap;
  Bitmap.Free;
end;

end.
```

```
unit TimerUnit;

interface

uses
Windows,SysUtils;

var
  CPUClock: extended;
  FStartTicks: Int64;
  FLastTicks: Int64;
  FStartStopConstant: Int64;

Procedure Calibrate(Var ClockS: Extended);
Procedure StartClock;
Procedure StopClock(Var ETime: Extended);

implementation

function GetCPUTick: Int64;
asm
  DB $0F,$31
end;

function CalibrateCPU: Int64;
var
  t: cardinal;
  PriorityClass, Priority: Integer;
begin
  PriorityClass := GetPriorityClass(GetCurrentProcess);
  Priority := GetThreadPriority(GetCurrentThread);
  SetPriorityClass(GetCurrentProcess, REALTIME_PRIORITY_CLASS);
  SetThreadPriority(GetCurrentThread, THREAD_PRIORITY_TIME_CRITICAL);

    t := GetTickCount;
    while t=GetTickCount do;

    Result := GetCPUTick;
    while GetTickCount<(t+400) do;
    Result := GetCPUTick - result;
    CPUClock := 2.5e-6*Result;

  SetThreadPriority(GetCurrentThread, Priority);
  SetPriorityClass(GetCurrentProcess, PriorityClass);
end;

Procedure Calibrate(Var ClockS: Extended);
BEgin
  CalibrateCPU;
  ClockS:=CPUClock;
End;
```

```
Procedure StartClock;
Begin
  FStartTicks:=GetCPUTick;
End;


Procedure StopClock(Var ETime: Extended);
Begin
    FLastTicks := GetCPUTick - FStartTicks;// - FStartStopConstant;
    ETime:=((FLastTicks/CPUClock)/1000);
End;

Begin
CalibrateCPU;
end.
```