Uncertain Input Estimation with Application to KalmanTracking

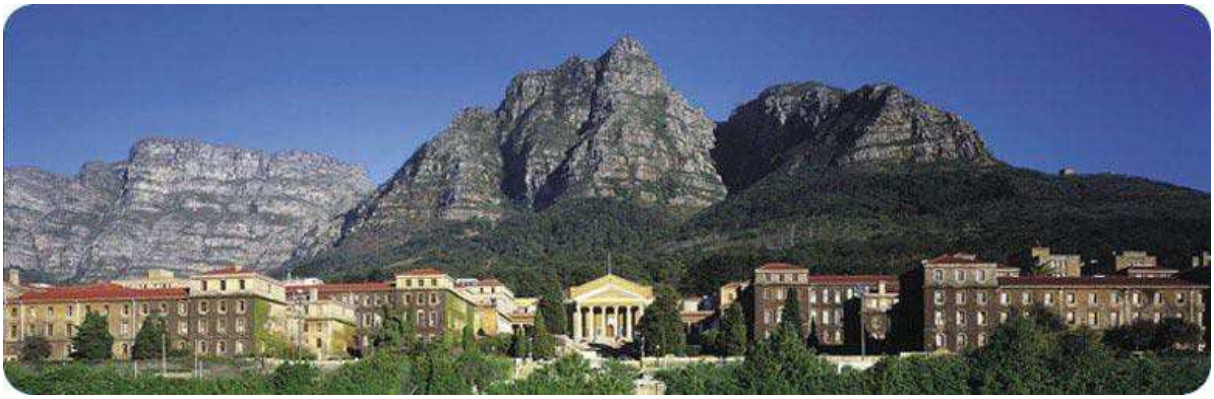
A DISSERTATION SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE, FACULTY OF SCIENCE AT THE UNIVERSITY OF CAPE TOWN IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF SCIENCE
IN
INFORMATION TECHNOLOGY

By

HUBERT TANGEE NASHENDA
JANUARY, 2011

SUPERVISED BY
DR. AUDREY J.W. MBOGHO



The author declares that the text and work presented in this Master thesis are original and that no sources other than those mentioned in the text and its references have been used in creating the Master thesis.

Acknowledgments

I would like to thank Prof. Dr. Audrey Mbogho for her advice, invaluable time and constructive feedback in the preparation of my research dissertation. Furthermore, I would also like to thank the various referees for their very constructive feedback and contributions.

Special thanks to my family and friends for the continuous encouragement and support provided in the compilation of this research dissertation.

Windhoek, Namibia,
January 2011
Hubert Nashenda

ABSTRACT

Many motion tracking systems average and integrate tracking measurements over a period of time in order to reduce the effects of device noise, external noise and other disturbances. The problem is that the target (user) is likely to be moving throughout the sample time, and this will in effect introduce additional "noise" (**uncertainty**) into the measurements. Other sources of uncertainty in tracking measurements are: poor signal, measurement noise, under-constrained measurements and additional movement from background sources or other parts of the person's body. Without filtering, noise can cause small variations in the estimated tracking positions (tracking drift) over time. Although many filters and algorithms are available to account for uncertainty due to noise, the Kalman filter (Kalman, 1960) has been chosen in this approach. The Kalman filter has been chosen due to its ability to estimate tracking positions, as well as its ability to account for uncertainty in the tracked object's position where it is occluded by other objects which are either stationary or moving.

In this research an inexpensive algorithm is presented which detects the slightest motion in a monitored environment and then tracks the motion or the target very accurately. The theory behind the approach is presented as well as a real life application.

Keywords: Kalman Filter Algorithm; Kalman Tracking; Uncertain Inputs; Single Object Tracking; Stochastic Estimation and Control; Noise; Optimisation

Table of Contents

LIST OF FIGURES

LIST OF ABBREVIATIONS AND SYMBOLS

k	Time index
x_k	State space
w_k	Input noise
R^n	n dimensional real space
$\in$	Element
Q_k	Input Noise Covariance Matrix
σ	Standard Deviation
A_k	State Transformation Matrix
$u_k = \mu_k(x_k)$	Input Stabiliser or Controller
μ_k	Input Stabiliser Policy
B_k	Matrix relating Input Controller to the current state
z_k	Measurement vector
H_k	Matrix relating state to measurement
v_k	Measurement noise
R_k	Measurement Covariance Matrix
I_k	Information Vector
$E\{\}$	Expectation
K_k	System generated gain
$p()$	Probability distribution
$N()$	Normal probability distribution
e	Error
P_k^-	A priori Error Covariance Matrix
P_k	A posteriori Error Covariance Matrix
Δt	Change in time

1. Introduction

1.1 Purpose of Research

In 1960 Rudolf Emil Kalman published his famous paper on a recursive solution to a discrete data linear filter which was based on state space techniques and recursive algorithms (Kalman, 1960). The Kalman Filter revolutionised the field of estimation as it can provide past, present and future state estimation even when the precise model of a system is unknown. Since its inception, the Kalman Filter has been the subject of extensive research, which led to the development of the extended Kalman (Merven et al. 2003) and Unscented Kalman Filter (Isard and Blake 1998) for non linear models and systems. Today the applications of the Kalman filter seem limitless and have contributed enormously, particularly in the field of tracking, navigation and computer aided motion.

The purpose of this research is to implement and test the Kalman filter. The approach involves a mathematical model of the tracking system and environment, in order to represent the system in a computational environment. This mathematical representation of the system is however inaccurate and presents many uncertainties to the tracking data. This study presents a Kalman filter as a viable solution to the uncertainties in the mathematical tracking system. The solution is discussed and developed further in the literature. The research attempts to answer three major questions on Kalman tracking which provide a greater understanding of Kalman tracking methods. These three major questions are: First, how accurate are rigid 2D object tracking methods when used for articulated (composed of rigid parts attached by links or joints) object tracking? Second, how does the tracking algorithm perform when treating occlusions (tracked object's visibility to the camera is obstructed by another object) as noise? Third, Is it possible to manipulate the Kalman input parameters in order to achieve more accurate tracking?

1.2 Theoretical Framework

Object tracking has received a considerable deal of interest due to the large number of potential applications. An important application is advanced human machine interfaces, where vision is required to complement speech recognition and natural language understanding in order to facilitate natural dialogue between humans and machines (Gavrila, 1999). Tracking is an important part of the entire process, as gesture recognition, facial expression analysis and face pose estimation are used to enable a machine to interact with the user and the environment.

Virtual reality and computer animation have also benefited from motion tracking. In order to insert and animate an avatar/character in the virtual world, the motion of the entity/object representing the avatar in the real world, has to be captured and reproduced in the virtual world. The captured motion data can then be used to steer the avatar in the virtual world. A perfect example of this implementation occurs in computer games such as FIFA 2010, where offline captured data of a real moving person is used to animate an autonomous soccer player in the virtual world.

Object tracking has also found applications in the automotive industry, such as implementations which provide vehicles with path planning and obstacle avoidance capabilities.

This research proposes the implementation of a tracking, surveillance or monitoring system using a digital camera connected to a computer. The purpose is to achieve this without any human intervention. The image passing through the camera to the computer is then represented on a pair of coordinate axes. In theory this application should provide an accurate representation of the location of any moving object being tracked, monitored or under surveillance in the coordinate realm. This forms the basis of robot vision. Unfortunately this implementation only works adequately in the theoretical world but fails almost completely when applied in the real world. The failure is due to the fact that the tracked or monitored image from the camera is represented inaccurately in the system model. The reason for the inaccuracy in projection is due to the large amount of noise present in the image data as well as inaccuracies in the mathematical model. Noise is present in image data due to calculations involved in transforming the real location of the object onto the pair of coordinate axes. These calculations may contain a component of error. This error is increased with each subsequent

frame obtained from the camera. Another reason for the presence of noise in the received image could be due to the effect of light from external sources as well as light reflecting off nearby objects. External light interferes with the light emitted by the tracked object, thus making interpretation of the measurement data inaccurate. The noise and disturbances have become part of the image forming the input data to the computational environment. Input model uncertainty arises in this case as there is no certainty on the accuracy of the data/image from the camera.

The camera, computer, light, environment and tracked object can be thought of as being a system. A system design approach is presented in this research in order to compensate for the inaccuracies inherent in the input image and the system model. The approach presented in this dissertation utilises a Kalman filter to model the system and to account for any uncertainty. Since 1960, significant progress has occurred in the applications of Kalman filters for estimation purposes. The Kalman filter achieves this by representing the system model in the form of state space vectors and matrices. State space allows a mathematical representation of a model as a set of inputs, output and state variables related by first order differential equations. The variables are represented as vectors. The differential and algebraic equations are represented in matrix form (Williams, 1962).

The Kalman filter attempts to optimally estimate the position of the moving object. The noise present in the system, measurement noise and input noise are represented by means of an input covariance matrix and a noise measurement covariance matrix. The effects of noise are smoothed out when an image is taken, as the filter provides an estimate of the moving object's position by taking account of the noise present in each image or received measurement. The Kalman filter is chosen as a solution for accurate computerised monitoring, surveillance and object tracking, due to the fact that the filter provides an intelligent means of integrating measurement data into an estimate. It achieves this by recognising that measurements are noisy and that they should be ignored at times or only have minimal impact on the data/position estimates.

The development of this dissertation is guided by Bovik (2009), Nikolaidis (2009), Konrad (2009), Tekalp (2009) and Bertsekas (1971). The knowledge of the authors on video processing, motion tracking, motion detection and motion estimation, provide a robust foundation on which to develop the literature review.

The Design of the Kalman filter utilised in the practical aspect of this dissertation is based on the approaches of Karna et al. (2007), Merven et al. (2003) and Freeston (2002). These authors were considered for the design of the Kalman filter, because of their simple approaches towards 2D Kalman tracking.

Thus the theory is presented in order to explain the mathematical model used for the tracking system. The Kalman filter is used to overcome inaccuracies and limitations in the system model.

1.3 Scope of Research

The literature related to the topic was introduced and extended to enable the design of a mathematical model of the real time tracking system as well as a Kalman filter. A computational algorithm was then written to represent both the mathematical tracking model and the Kalman filter. The software package used to model Kalman Tracking is called "MatLab". MatLab is a numerical computing and programming language developed by MATHWORKS. This software is utilised due to the ease of world modelling that it enables. A digital camera placed in a room records activities in the room. The camera is then connected to the computer running MatLab. This will form the complete tracking system. The movement of a person through the room is then tracked for various designs of the Kalman filter. This will show the ability of the Kalman Filter to estimate the location of the person in the presence of uncertainties and noise. The test will also facilitate the determination of the optimum design parameters for the Kalman filter for tracking purposes. The approach to the research will be guided by a set of questions, hypotheses and methods. The dissertation commences with the following questions, hypotheses, data and methods:

QUESTIONS, HYPOTHESES, DATA AND METHODS

1. How accurate are 2D rigid object tracking methods when used for articulated (composed of rigid parts attached by links or joints) object tracking?

Hypothesis: Methods used for tracking rigid objects in the 2D image plane can be extended to accurately track an articulated object by focusing on the largest centre of mass.

Methods: Tracking occurs in the 2D image plane by placing an (x, y) coordinate system upon

each processed image. The centres of mass of each part of the articulated object (limb, torso, head in the case of humans) is determined. The algorithm then focuses on the largest centre of mass and uses that to track the articulated object. The 2 dimensional tracking methods are similar to those utilised by Karma et. al (2007), Merven et. al (2003) and Freeston (2002). The movement of a person or tracked object in a room is expressed as linear equations. The linear equations are then expressed in a Kalman filter. The algorithm for the Kalman filter and linear equations are programmed in MatLab. The MatLab code will then form a position estimate on the same image as that of the tracked object. This will provide a visual display of the performance of the Kalman position estimates.

The Kalman system model describes the (x, y) coordinates of the person being tracked, as well as measurements z taken by the Kalman filter in a 4D state vector X in state space. The state space representation (also known as the "time-domain approach") provides a convenient and compact way to model and analyse systems with multiple inputs and outputs (Williams, 1962)

2. How does the tracking algorithm perform when treating occlusions as noise?

Hypothesis: The tracking algorithm can provide a good estimate of the location of the tracked object where all or some parts of the tracked object are not visible due to obstruction from another object

Methods: A moving object/person will be tracked in a room by a static camera. An occlusion is introduced to the tracked environment by having the tracked object move behind a wall in the scene. The tracked object will thus not be visible to the camera until it re-emerges from behind the wall.

3. Is it possible to manipulate the Kalman input parameters in order to achieve more accurate tracking?

Hypothesis: The values or parameters of the Kalman input covariance matrix can be adjusted to have an effect on the accuracy of the Kalman output estimation. Thus values exist which can enable optimum Kalman estimation.

Methods: Values chosen within a certain range will be utilised for the Kalman input covariance matrix, in order to observe the effect on position estimates. The range is

determined from 0 to a value (in this case 100) where all other subsequent values provide the same Kalman output. The values utilised for the input covariance matrix were: 0, 1, 10, 50 and 100. In this manner a relationship between Kalman output estimates and the Kalman input covariance matrix values is determined.

1.4 Limitations and Delimitations

In this research, the Kalman tracker is implemented for offline processing and tracking. The Kalman filter is utilised on a recorded video clip and not in a real life setting. The purpose of this research is merely to illuminate its usefulness.

The tracking of multiple objects through a digital camera was not implemented, as this would have required sophisticated computer algorithms and resources. Only one object is tracked for the purpose of illustrating the usefulness of a Kalman filter in tracking mechanisms. The algorithm used can be utilised in real time, but in my implementation it was implemented offline, using recorded data.

2. Literature Review

Motion tracking aims to obtain the trajectory of moving objects or in certain cases the trajectory of the camera. Motion tracking methods attempt to determine relationships between tracked objects in consecutive frames. Relationships are determined using image information parameters such as position, velocity or colour etc. Motion tracking is different from object detection which aims to estimate the object's orientation or position within an image. Detection and tracking are however not unrelated. The research carried out by Yilmaz et. al (2006), indicates that detection is involved in one of the two major approaches that one can use to obtain a tracking algorithm. According to their research with the first method, object detection is performed on each frame and subsequently correspondence is sought between objects detected in subsequent frames. This allows a trajectory to be obtained for each tracked object. The second approach discussed in their research, combines the detection and correspondence finding steps. The objects position and/or orientation in the next frame is then predicted rather than detected, by means of information obtained in the previous frame.

A significant application of motion tracking is smart surveillance. Smart systems detect the motion, as well as classify the motion as being either human or non-human. Such systems could perhaps also perform facial recognition and tracking for access control using single or multiple cameras. Other smart systems are able to perform human behaviour analysis to detect suspicious behaviour (e.g. in front of an ATM). Thus the large number of security cameras installed in security sensitive areas can be used as a tool for efficient real time automated surveillance. Furthermore, motion tracking allows human operators to search archived video for specific video patterns without actually viewing the video (Nikolaidis et al. 2009).

The basic methodology pertaining to how a computer can monitor an environment through a camera, without the aid of other external sensors is depicted in Figure 1 on the next page.

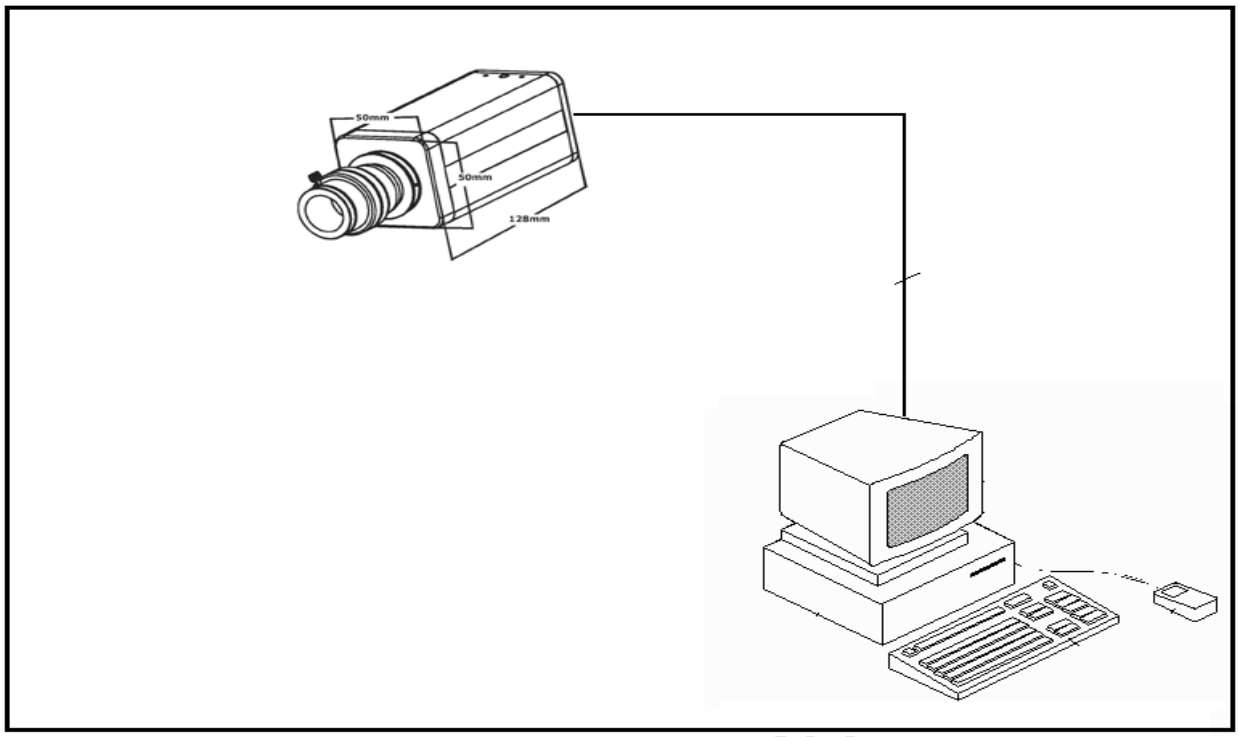


Figure 1. Computerised monitoring of the environment

A digital camera is connected to a computer. The camera acts as a sensor in this implementation. The computer then processes the image obtained through the camera for monitoring purposes.

In figure 2 a room is monitored by a digital camera. Figure 2 shows the person being tracked and monitored enclosed within an ellipse. In figure 2 the centre of mass of the person is indicated by the green arrow. The green arrow will represent a vector plot of the centre of mass of the tracked object's location.

The room in figure 2 has a 2 dimensional pair of coordinate axes (x, y) superimposed upon it. The location of the centre of mass of any object is then represented as a point on a coordinate plot using MatLab. MatLab utilises a spatial coordinate or pixel coordinate system to represent points on an image. The pixel coordinate system allows a pixel to be treated as a discrete unit, uniquely identified by a single coordinate pair, (x, y) (Mathworks, 1998).

Figure 2. Coordinate axes for 3D room

Figure 3 illustrates the MatLab spatial coordinate system used for images. Notice that *y* increases downward.

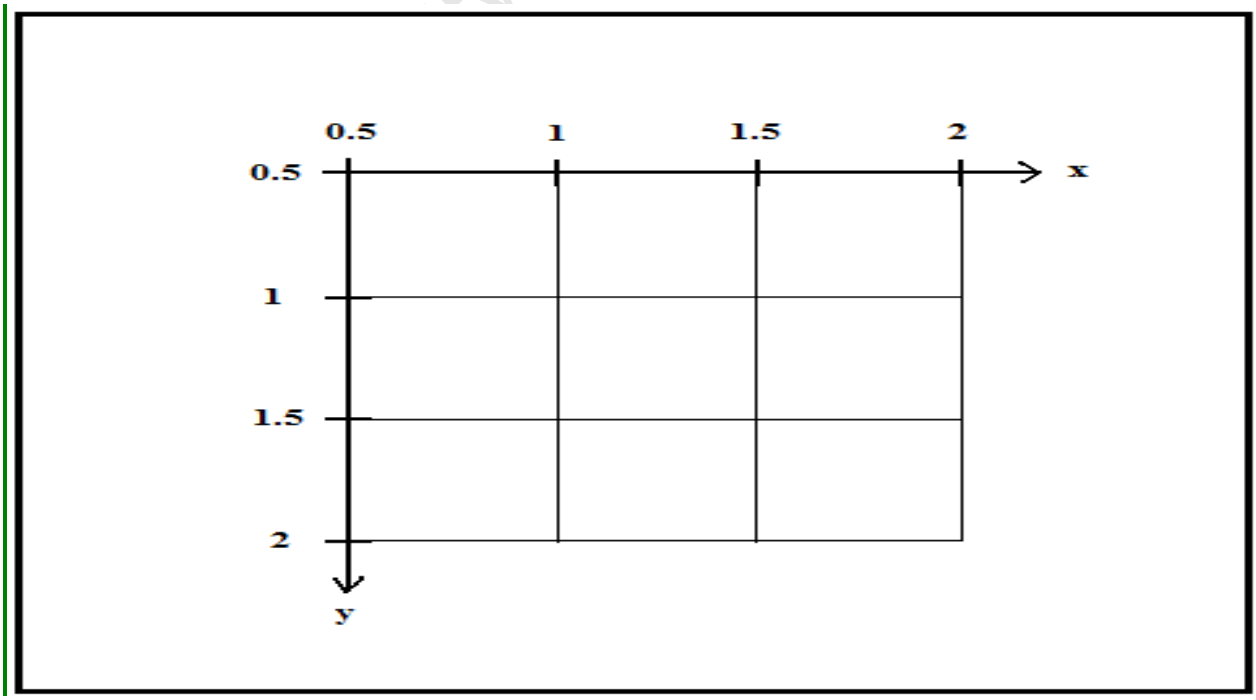


Figure 3. Spatial coordinate system for images in MatLab.

In MatLab the spatial coordinates of the centre point of any pixel are identical to the pixel coordinates for that pixel. There are some important differences, however. In pixel coordinates, the upper-left corner of an image is (1,1), while in spatial coordinates the upper left corner of an image is (0.5,0.5). The spatial coordinate system is continuous. The spatial coordinate system is shown in figure 3. This representation of the world in a coordinate form, allow for the accurate tracking and monitoring of an environment through a computer. These areas can be modelled and represented in the 2D coordinate system. This is the basis of 2D object tracking. An important criteria adopted for tracking in general is the dimensionality of the tracking space. Object tracking algorithms are either classified as 2D or 3D. 2D object tracking deals with the projection in the image plane of the tracked object which in reality moves in 3D. 3D object tracking attempts to track 3D object movement by means of 2D information obtained from a one or more cameras.

This coordinate plot, shows that mathematical models can be used to represent certain aspects and behaviour of a monitored system. The mathematical models can be used to represent characteristics of the system such as the relationship between inputs and outputs, systems response etc. A measurement device, in the form of a digital camera can be utilised to output data signals proportional to variables of interest. The variables of interest in this case are the x , y and z coordinates of the object being tracked. Feedback control can be utilised, so that the output variables can serve as inputs to a system controller. This is the basis of deterministic system models (Maybeck, 1979).

However, the problem with the deterministic system model approach is that the basic computer and camera system do not perform as accurately when implemented. Large amounts of noise (light interference) present in the environment, the frame rate and the distance that light has to travel to the camera, are among the many issues contributing to the misrepresentation of the tracked object in the 3D coordinate system (Maybeck, 1979. Figure 4 indicates the representation of the tracked object in a coordinate system due to errors, noise and disturbances.

Figure 4. Misrepresentation of the location in 3D space

Figure 4 indicates the inaccurate vectorised plot of the centre of mass of the tracked object in blue. The ellipse no longer encloses the person, whose position is indicated by the green arrow, due to noise and system errors. What this implies is that interaction with the real environment of figure 2 and 4, by means of a computerised model of the room, would not be accurate due to inaccuracies present in this virtual model. For example, if the computer had to shoot or throw an object at the person enclosed within the ellipse of figure 2, the object would be thrown to the ellipse in figure 4.

Deterministic system models do not adequately represent a real life system as shown in figure 4. The behaviour of system models does not accurately represent real life models for the following reasons (Oberkampf, 2005):

- No mathematical system model is perfect.

  Models only depict characteristics of interest. Models depict the dominant modes of system response, thus many modes are left unrepresented. Many parameters within

a system are not modelled with the utmost certainty. Thus there are many sources of uncertainty in models. The tracking system does not compensate for the time lag involved in the movement of light from the tracked object to the camera. Due to the speed of light this time lag can be considered negligible, but may become significant if tracking of a single object occurs over an extended period of time.

- Disturbances which can neither be controlled nor modelled deterministically have an effect on dynamic systems. Deterministic systems are thus driven by controlled inputs as well as disturbances. In the case of the tracking system disturbances take the form of light from other external sources, as well as light radiated and reflected from other objects in the surrounding environment.

- Sensors do not provide perfect and complete data about the system.

  Sensors do not provide exact readings of measurements but provide their own distortions and noise. Sensors are almost always noise corrupted. The tracking system used a digital camera as a sensor. The light from the image experiences some distortion due to reflection at the camera lens, refraction within the camera lens, time gap between each frame and electrical system noise added to each pixel of the image.

- Limited amount of examples and measurements.

  In order to mathematically model a system, measurement data and previous models or examples are required. A shortage of these factors could result in uncertainty is system representation.

- Randomness inherent in observed phenomena.

  Every system in nature constitutes a certain amount of randomness that cannot be modelled accurately. This may introduce an amount of uncertainty in measurements.

In order for the tracking system to model the environment accurately, a means has to be obtained to address the uncertainty present in system data. According to Oberkampf (2005) there are two basic types of uncertainty, namely: aleatory and epistemic uncertainty. The two types of uncertainty are described as follows:

i. Aleatory Uncertainty

This is the inherent variation associated with a physical system or the environment. It is also referred to as variability, irreducible uncertainty, stochastic uncertainty and random uncertainty. An example of aleatory uncertainty is variation in atmospheric and lighting conditions.

ii. Epistemic Uncertainty

This is due to the lack of knowledge of quantities, of processes in the system or of the environment. It is also referred to as subjective uncertainty, reducible uncertainty and model form uncertainty.
Examples of epistemic uncertainty are:

- Lack of experimental data to characterise new processes.
- Poor understanding of initiating events theory.

System design approaches have been utilised in order to overcome issues arising from uncertainty in measurements. This inability to perfectly model a system has led to the following approaches to uncertainty.

- Develop system models that account for uncertainty.
- Use a system model which includes incomplete, noise corrupted data from the sensors in order to optimally estimate quantities of interest.
- Design controllers for the system which operate in the presence of noise corrupted data, uncertain system descriptions and disturbances. These controllers have to form estimates of measurement data in the presence of uncertainty.
- Evaluate performance capabilities of estimation and control systems before and after they are built.

To overcome the problem of uncertainty in the image input data to the computerised environment, system models accounting for uncertainty are designed and implemented. These system models contain stabilised control systems, which operate on noise corrupted data, in order to form estimates of the measurement data. The designed system receives uncertain,

noise corrupted inputs and manipulates this data in a manner as to achieve an output similar to that which would be obtained with certain, precise and definite inputs. The system has to allow for the integration of measurement data into system estimates. Thus the system has to be designed in such a manner that it recognizes that measurements are noisy and should be ignored at times in order to have a minimal effect on estimates. This is a complicated procedure, which requires an explanation of basic mathematical concepts and expressions used in estimate techniques. This procedure is elaborated upon in the next chapter.

Video based object tracking belongs to a class of **pass**ive tracking methods. Passive tracking methods rely on signals naturally emitted by the object being tracked (Moeslund and Granum 2001). Passive tracking methods differ from active tracking methods which involve placing sensors or transmitters on the tracked object. Active tacking methods simplify motion capture to signals concerning factors such as object location, orientation or speed which can be transmitted directly to a receiver for processing by a computer (Moeslund, 2000).

Active objects tracking techniques include (Welch and Foxlin, 2002; Burdea and Coiffet, 2003):

- Mechanical trackers that are based on a kinematic structure (either serial or parallel),which is attached to the object (usually the human body) and consists of links interconnected with joints.
- Inertia trackers, which are devices that consist of gyroscopes and accelerators and measure the translational acceleration, as well as the angular velocity of an object.
- Ultrasonic trackers, where transmission and sensing of ultrasonic waves is used. The time taken for a brief ultrasonic pulse to travel from a stationary transmitter placed in the environment to a receiver attached to the moving object is measured and used to identify the object position or orientation.
- Magnetic trackers, are non-contact devices and use the magnetic field produced by a stationary transmitter to measure the position of a receiver placed on the moving object.
- Radio and microwave trackers, where the time-of-flight of the waves from a stationary transmitter to a receiver attached on the object of interest is measured to determine the range of an object.

- Hybrid trackers, which employ more than one of the above position/orientation measurement technologies to track objects more accurately than what a single technology would allow.
- Marker or LED (Light Emitting Diode) Tracking. These are placed on the moving object, with stationary cameras sensing the emitted light. In other applications cameras are mounted on the moving object and sense emitted light from LEDs or markers placed in the environment.

Active object tracking methods can be successfully used to track objects. However, these methods come with many drawbacks. Some of the drawbacks associated with active tracking methods are:

- A well controlled environment is required. The environment has to be such that communication can always occur between transmitters and receivers.
- Tracking devices have to be placed on the object. This is a limitation in situations where online constant monitoring of random objects and people is required.
- The process itself is intrusive as it involves placing devices (sensors and transmitters) on a tracked object or person.

In the rest of this section several criteria are used to provide a classification scheme for the object tracking algorithms.

The output of an object tracking algorithm depends on the application and the representation used to describe the object (Nikolaidis et al. 2009). Thus the output can take the form of the 2D coordinates of the tracked object's centre of mass, the object's 3D position in world coordinates or the contour of the object. As mentioned before the dimensionality of the tracking space is one important criterion adopted for tracking. This dimensionality is either 2D or 3D. 3D Object tracking can be defined as the estimation of the position and orientation of a rigid object in 3D space from video data obtained from one or more video cameras. The location of a rigid object in 3D space is determined by the position of its centre of mass with respect to a world coordinate system. In certain applications determining only the position (i.e. considering the object as a point mass) or only the orientation parameters might of an object may be sufficient. A typical 3D rigid object tracking algorithm involves the initialisation of a 3D geometric model and enrichment of the model with texture information. For each

subsequent frame the model parameter vector (position and orientation) that best describe the object in the current frame is evaluated. Model parameters derived in the previous frame are used to provide a rough estimate of parameters in the current frame. The algorithm attempts to maximise the similarity between the model on the image plane and image content projected onto the image plane (Nikolaidis et al. 2009).

Another important characteristic adopted for tracking is the structure of the object to be tracked. There are three categories into which an objects structure can be classified. These categories are rigid, deformable and articulated. Rigid object tracking deals with tracking the motion of rigid objects, whilst deformable object tracking deals with tracking the motion of deformable objects. On the other hand, articulated object tracking concerns the tracking of objects composed of different rigid objects connected by links or joints. Living beings such as humans, animals and insects exhibit articulated attributes. Thus, with articulated tracking each of the rigid parts experience rigid motion but the overall motion of the object is not rigid (Aggarwal et al. 1998). A major difficulty in articulated object tracking is to deal with inter-frame changes of the moving object. The image shape of a moving object may undergo deformation, since a new aspect of the object may become visible or the actual shape of an object may change. Jang and Choi (2000) proposed the utilization of active models for tracking moving objects. This model based tracking algorithm can adapt itself dynamically to an image sequence, by capturing the changes in the images shape from one time frame to the next. This allows for the tracking of articulated objects. Such adaptation of the active model occurs under the framework of energy minimisation. An energy function is designed to embody structural and spectral attributes of a target. A Kalman filter is than applied to predict motion information. The Kalman tracking algorithm has two main modules: a prediction module and an updating module. The prediction module estimates motion parameters of a target. The estimates obtained are used to limit the possible location areas of a target in successive frames. The updating module accounts for inter-frame changes of the target. The updating module first uses template matching in order to determine the best match for an old model. The best match model is then incrementally transformed until a state of minimal energy is reached. The state of minimal energy, in turn, reveals the updated model for the next frame. Thus the active model incrementally evolves in order to reflect inter-frame changes under the framework of energy minimisation. Precise tracking of the corresponding articulated structures in 3D is necessary in order to extract higher level information (e.g. facial recognition, gesture recognition, etc.) about the behaviour of tracked objects or tracked people. For this reason most of the articulated object tracking algorithms attempt tracking in

3D. Many 2D articulated object tracking algorithms employ a 3D model of the articulated object, as the methodology is similar to that of 3D articulated tracking algorithms. 2D or 3D articulated object tracking can be model free or model based. 2D articulated object tracking refers to recovering the position in the image plane of the rigid parts making up the articulated object.

2D rigid object tracking tries to determine the motion of the projection of one or more rigid objects on to the image plane. This motion is the relative motion between the camera and the observed scene. A basic assumption behind 2D rigid motion tracking, also utilised with the tracking algorithm implementation for this dissertation, is that there is only one, rigid, relative motion between the camera and the observed scene. This assumption thus rules out articulated objects and concentrates on the part with the largest centre of mass in the tracked articulated object/person. Methods for 2D rigid Object tracking can occur in the following categories (Nikolaidis et al. 2009):

- Region-Based methods

  An image region is a set of pixels having the same characteristics. These homogenous pixels can be obtained by means of image segmentation. Image segmentation refers to partitioning video into spatial, temporal, or spatiotemporal regions that are homogeneous in some feature space (Figueroa et al. 2004). For 2D region-based object tracking, a region can be defined as the projection of the tracked object onto the image plane. If the colour of the object to be tracked can be modelled and distinguished from other objects in the frame, then colour information can be very effective in region based object tracking. This can lead to real time object tracking of the order of 20 -30 frames per second . This method is straightforward and easily implementable. Due to ease of implementation this method was used for the 2D tracking in the practical implementation of this dissertation. The method is elaborated in section 3.2 of the next section

- Contour-Based methods

  Contour based object tracking methods involve representing the object using outline contour information and devising an object tracking algorithm to track the outline over time. Contour based methods are more complicated than region based methods where colour can be used to represent an entire region. Contour based methods are however more

robust to partial occlusions (part of tracked object is not visible to the camera) and illumination variations. Contour tracking has found numerous applications in surveillance, medical diagnosis and audiovisual speech recognition. Active contours, also known as "snakes" have been used for object segmentation and tracking. An active contour algorithm dynamically deforms a contour to "lock" onto image features such as lines, edges, boundaries etc. Active contours are affected by factors such as image content.

- Feature Point-Based methods

Feature point-based object tracking is the attempt to recover the motion parameters of a feature point in a video sequence. This involves the parameters associated with the planar translation of a point, as points in 2D space neither rotate nor translate with respect to depth. The procedure is as follows:

Let $A = \{A_0, A_1, \dots, A_{N-1}\}$ denote the N frames of a video sequence. Let $m_i(x_i, y_i), i = 0 \dots N-1$ denote the position of the same feature point in those frames. The aim is to determine a motion vector $d_i(d_{x,i}, d_{y,i})$ that best determines the position of the feature point in the next frame, $m_{i+1}(d_{i+1}, d_{i+1})$, such that
$m_{i+1} = m_1 + d_i$.

Feature point-based tracking is more prone to tracking drift but can be implemented efficiently (Toyama, 1998). Potentially good points for feature tracking are those with distinctive good characteristics such as brightness, contrast and texture. Robustness of the point neighbourhood to illumination variation and viewpoint changes is another desirable characteristic of good feature points.

- Template-Based methods

Template based techniques are similar to region based techniques, in that a template is essentially a model of the image region and therefore of the corresponding object to be tracked. The first step, initialisation, involves selecting the template to be used. Templates are acquired prior to tracking. A template specific to a particular class of objects can be created. Template matching can be defined as the process of searching the target image (i.e. the current frame of the video sequence) to determine the image region that resembles the template, based on similarity or distance measure. Essentially, the template region

should undergo a geometrical transformation that would "place" it onto the target image in such a way as to minimise the distance measure used. The goal of template matching algorithm is to estimate the parameters of such a transformation.

Object tracking algorithms can also be characterised by the model (3D volumetric model, surface model, colour distribution model, geometric model etc.) of the object to be tracked. Depending on the application the tracking algorithm can either be model based or model free. In surveillance applications (Haritaoglu et al. 2001; Isard and MacCormick 2001), 3D geometry models of the object to be tracked are not necessary, because the parameters of interest involve only the presence and the spatial position of humans.

The tracking mode of operation also serves as a characteristic of object tracking methods. The mode of operation is either online or offline. Online tracking algorithms utilise information obtained from pervious frames in order to predict the object location in the current frame. Offline object tracking algorithms utilise information from an entire image sequence stored in a database. Thus offline tracking algorithms can make use of information before (prior) and after (posterior) to the frame of interest. The only disadvantage with offline tracking methods is the increased computational load. In the practical aspect of this dissertation offline tracking methods were also used as the mode of operation.

Before applying a tracking algorithm proper **initialisation** is required. Initialisation can occur online or offline, with the aim of obtaining information about the tracked object, the tracking environment and/or the camera. Initialisation of the camera can be achieved easily by means of offline calibration. Offline calibration enables the determination of intrinsic parameters as well as extrinsic parameters for the camera. Examples of intrinsic parameters are the focal length and radial distortion. An example of an extrinsic parameter for the camera is the external scene geometry. The tracking algorithm implemented in this dissertation obtains the object position in the first frame by means of background subtraction. Here initialisation involves capturing the background image. Determining intrinsic and extrinsic parameters in such a manner requires a fixed camera setup. Recalibration is required if the camera setup changes. Online calibration of the camera is also possible (Faugeras et al. 2000; Malis and Cipolla 2002). Another requirement for tracking initialisation, is the initialisation of the tracking model. For the implementation in this dissertation, the initialisation involved object segmentation in order to determine an x, y coordinate vector of the initial position of the

tracked object in the environment.

An important feature of the tracking algorithm, mentioned briefly before, is the ability to handle occlusions. A major concern with object tracking is either occlusion or self occlusion of the tracked object. An object can either be partially or totally occluded. Occlusion occurs where parts of or the entire tracked object is not visible due to obstruction from static (walls, trees etc) or moving objects (people, cars etc). One intelligent method to address occlusions is re -initialisation of the tracking algorithm in frames where occlusions are high enough to cause tracking drift. Re-initialisation implies that the initialisation of the tracking algorithm (e.g object detection for x and y coordinates) occurs periodically in order to detect the object after it has been occluded. An effective method for occlusion handling involves the Structural Kalman filter (Jang et al. 2002). This filter can successfully estimate motion information when partial, total or self occlusion occurs. The Structural Kalman filter does not treat a target as one entity, but rather partitions a target into several sub-regions. The filter then evaluates each sub-region independently, together with its relationship with other sub-regions. The Structural Kalman Filter is thus a composite of two types of Kalman Filters: nth Cell Kalman filter and the Relation Kalman Filters. The Cell Kalman filter is allocated to each sub-region and the Relation Kalman filter is allocated to the connection between two adjacent sub-regions. The Cell Kalman filter forms estimates of motion information for each sub-region, whilst the Relation Kalman filter forms estimates of the relative relationship between two adjacent sub-regions. The final estimate of the occluded sub-region is thus obtained by combining the estimates of the involved Cell and Relation Kalman filters. Thus the Structural Kalman filter supplements the unreliable measurements of an occluded sub-region with the measurements of its adjacent sub-regions. As an object becomes more occluded, the Structural Kalman filter's dependency on relation information increases and the role of the relation filters becomes crucial in determining the a priori estimate of an object. The relational information supplements the unreliable measurements on a partially occluded sub-region, so that an a priori estimate of the next state of the occluded region is obtained using the relational information and actual measurements from the Structural Kalman Filter. When no occlusion occurs, the relation filters become inactive and the Structural Kalman filter operates just like the Kalman filter. Another method of occlusion handling is to segment the object of interest into N parts and to perform tracking on the set of these parts (Gentile et al. 2004). This procedure is performed by estimating N possible transformations and using a voting scheme to select a single transformation representative of the entire object. Performance of all transformation are evaluated at regular intervals in order to enhance the accuracy of the

tracking process. Tracking is thus enabled in the presence of occlusions as only a relatively small part of the objects need to be visible to successfully track the object. For the object tracking algorithm utilised in this research dissertation, no occlusion handling mechanisms are determined, but it is has been left to the Kalman filter to predict object information when total occlusion does occur in frames. Occlusions are ignored and treated as noise. The Kalman filter predicts the position of the occluded regions, based on velocity estimates obtained from measurements prior to total occlusion. This method is robust against short term occlusions. However the approach is inefficient against severe, long term occlusions. In multiple camera systems, self occlusions and occlusions between objects can be handled more efficiently. Multiple camera methods combine information from multiple cameras in order to determine the best / optimal view. When a camera looses information due to occlusion, information for the tracked object can be obtained from other cameras also observing the scene (Utsumi et al. 1998).

Developing an object tracking algorithm capable of estimating object motion information under all possible conditions is a very difficult process. The algorithm has to account for disturbances in the system such as occlusions (all or some parts of object are not visible to the camera), varying/poor lighting conditions, single/ multiple cameras and image projection ambiguities. These difficulties have led to the development of constraints which only focus on certain aspects of the tracking model. Constraints allow for the tackling of specific aspects of a very complex model. The constraints can refer to factors such as the motion of the camera, motion of the tracked object or the appearance of the environment. The constraints can thus be used to form the characterisation of the tracking object's algorithm (Nikolaidis et al. 2009). Thus a tracking algorithm can be characterised by their focus, such as whether focus is on a particular object, the tracking environment, the number of objects and the state of the camera. Motion models identified in frames/images provide adequate information to act as constraints for the tracking process. Frame/image measurements can be utilised to update the tracked object's motion parameters in the motion models.

The process of identifying human models for tracking processes is more complicated due to the occurrence of unexpected events. Different activities undertaken by humans whilst being tracked also create additional tracking models. The greater the number of models which have to be considered, the greater the computational complexity. The increased computational complexity due to the increased number of factors up for consideration have resulted in the incorporation of prediction schemes for object tracking in consecutive frames. The prediction

algorithms utilise information acquired from previous frames and predict the state (e.g. location) of the tracked object in the current frame. The predicted state of the object is then compared with the actual state of the object in the current frame. A straightforward algorithm for prediction is the Kalman filter (Maybeck 1979; Welch and Bishop 1995; Zarchan and Musoff 2001) The Kalman filter also estimates prediction error. The Kalman filter is a linear Bayesian estimator requiring a linear state and measurement models. For non-linear state and measurement models, algorithms can be utilised which provide approximations to state and measurement models. The first approximation algorithm is an extension of the Kalman filter called the Extended Kalman filter (Merven et al. 2003). The Extended Kalman filter uses first order polynomials in order to form approximations. This approximation process of the Extended Kalman filter eventually introduces errors where states and models cannot be represented by first order polynomials. In the event of such occurrences an Unscented Kalman filter is utilised. The Unscented Kalman filter uses higher order polynomials to form approximations to state and measurement models (Julier et al. 1995). In order to make use of the standard Kalman filter, Extended Kalman Filter and the Unscented Kalman Filter, it is required that the average/mean of the total system noise is equal to zero and that the posterior distribution function (pdf) of the system state is zero. The posterior distribution function provides the probability of an event occurring after having taken into account new information (Reliability Engineering Resources, 2006). Thus a new (posterior) distribution is obtained. If these conditions do not exist then it would be impossible for any version of the Kalman filter to form exact or approximate representations of the system and measurement states. If the probability density function of the system noise is not zero, then no implementation of the Kalman filter will work properly. In such cases a Particle Filter can be used (Isard and Blake 1998). A Particle Filter is a sequential Monte Carlo algorithm which forms approximations to the posterior distribution functions of the system state and measurement states.

Other Practical 2D motion estimation algorithms are, Global Motion Estimation, Block Matching, Phase Correlation and Optical Flow via Regularisation. These methods deal with the other approach that can be used for devising tracking algorithms. According to this approach object detection is performed in each frame of a video sequence and afterwards a relationship is determined between objects detected in each frame. These practical motion estimation algorithms are as follows (Konrad 2009):

- Global Motion Estimation

Global Motion Estimation involves compensation for camera motion. Camera movement affects motion of all image points and is thus often an obstacle to solving various video processing problems. To detect motion in video captured by a mobile camera, camera motion must be compensated first (Li et al. 1997). Since camera motion is limited to translation and rotation, and affects all image points, a spatially parametric motion model supported on the whole image is appropriate. The spatially parametric motion models are accurate only for specific image formation, and object motion/surface models .

To handle large velocities and to speed up computations, the method needs to be implemented hierarchically. Thus, an image pyramid is built with spatial prefiltering and sub sampling (usually by 2) applied between each two levels. The computation starts at the top level of the pyramid (lowest resolution). The resulting motion parameters are projected onto a lower level.

Since the global motion model applies to all image points. Thus, points moving independently of the global motion may generate large errors and thus bias an estimate of the global motion parameters. The corresponding pixels are called outliers and, ideally, should be eliminated. This can be achieved by using a robust criterion. For example, a Lorentzian function or a truncated quadratic can be used, but both provide a non zero cost for outliers. This reduces the impact of outliers on the estimation but does not eliminate it completely. The outliers tend to appear at intensity transitions since it is there that any inaccuracy in global motion caused by a local (inconsistent) motion will induce large error in uniform intensity areas undergoing local motion the error remains small. By excluding the outliers from the estimation, the accuracy of computed motion parameters is improved.

- Block Matching

Block matching is the simplest algorithm for the estimation of local motion (pixel motion). It uses a spatially constant and temporally linear motion model over a rectangular region of support. The translational 2D motion is only valid for the orthographic projection and 3D object translation. This model applied locally to a small block of pixels can be quite accurate for a large variety of 3D motions. It has proved accurate enough to serve as the basis for most of the practical motion estimation algorithms used today. Due to its simplicity and regularity (the same operations are performed for each block of the image), block matching can be relatively easily implemented and, therefore, is used today in real-

time encoders for all video compression standards. In video compression, motion vectors d are used to eliminate temporal video redundancy via motion-compensated prediction. Hence, the goal is to achieve as low an amplitude of the prediction error as possible.

- Phase Correlation

  Phase Correlation can be used to overcome the shortcomings of block matching. As discussed above, block matching can precisely estimate local displacement but must examine all possible candidates. This is an exhaustive search. Methods based on the frequency-domain criteria are capable of identifying global motion. By combining the two approaches, phase correlation (Tekalp 1995) is able to exploit advantages of both. Likely candidates are computed using a frequency-domain approach and then they are assigned a spatial location by local block matching. The phase correlation method is basically an efficient maximisation of a correlation based error criterion. The shape of the maxima of the correlation surface is weakly dependent on the image content, and the measurement of their locations is relatively independent of illumination changes. However, rotations and zooms cannot be easily handled.

- Optical Flow via Regularisation

  Optical Flow Via Regularisation uses a translational/linear motion model at each pixel. The method attempts to find continuous functions ν_1 and ν_2, which are implicitly dependent on x, the location of the tracked object. An alternative to the above method is to formulate the problem directly in the discrete domain. By differentiating a discrete cost function, a system of equations can be computed and subsequently solved by Jacobi or Gauss-Seidel relaxation. This is a difficult calculus problem to solve.

3. Methods and Techniques

3.1 Segmentation

An image region is a set of pixels having the same characteristics. These homogenous pixels can be obtained by means of image segmentation.

Video segmentation refers to partitioning video into regions that are homogeneous in some feature space (Tekalp 1995). It is an integral part of many video analysis and coding problems, including, (i) improved motion estimation, (ii) 3D motion and structure estimation with multiple moving objects (Dimitrova et al. 2002), and (iii) video surveillance. These are computer vision applications, where segmentation helps to identify foreground and background objects and occlusion regions.

Video segmentation methods should be considered in the context of the requirements for the application in which they are used. Factors that affect the choice of a specific segmentation method include (Correia and Pereira 2004) the following:

■ *Real-time performance:* If segmentation must be performed in real time then simple algorithms that are fully automatic must be used. One can employ semiautomatic, interactive algorithms for off-line applications such as video indexing (Izquierdo and Ghanbari 2002).

■ *Scene complexity:* The more complex the scene, the more complex the segmentation algorithm required.

3.1.1 Scene Change Detection

Scene change detection is a straightforward segmentation problem since it is one dimensional. Scene change detection methods locate discontinuities in frames across which large differences are observed. These discontinuities are usually a combination of colour and motion (Gargi et al. 2000; Koprinska and Carrato 2001).

Temporal discontinuities may be abrupt (cuts) or gradual (special effects, such as wipes and fades). It is easier to detect cuts than special effects. The simplest approach for detecting

temporal discontinuities is to quantify frame differences in the pixel intensity domain. If a predetermined number of pixels exhibit differences larger than a threshold value, then a cut can be declared. This method is sensitive to presence of noise and camera motion. A slightly more robust approach may be to divide each frame into rectangular blocks, compute statistics of each block such as the mean and variance independently, and then check the count of blocks with changing statistics against a set threshold. Applying low-pass filtering to each frame prior to computing frame differences or block statistics should also improve robustness (Bovak 2009, p143).

An alternative to the above methods is to consider frame histogram differences instead of pixelwise or blockwise frame intensity differences. This method is implemented by computing n-bin colour histograms, $h_k(i), i = 1, \dots, n$ for each frame k. Methods such as the histogram intersection measure and Kolmogorov-Smirnov (Koprinska and Carrato 2001) test can be used to quantify similar or dissimilar histograms. Detection of these special effects requires combination of histogram difference and camera motion estimation. Camera motion can be estimated and frame compensation occurs before computation of these features.

3.1.2 Spatiotemporal Change Detection

Change detection methods segment each frame into two regions. These regions are classified as changed and unchanged regions in the case of a static camera or global (involving camera movement) and local motion (only pixel movement) regions in the case of a moving camera (Aach et al. 1993). This section deals only with the first case, where unchanged regions correspond to the background and changed regions to the foreground. The first case is the method used in the practical aspect of this dissertation for segmentation. The dominant motion segmentation approach (Bergen et al. 1991) offers a solution to the estimation of the camera motion without prior scene segmentation, but in the practical implementation of the dissertation a static camera was used. The spatiotemporal change detection method used in the dissertation is discussed in Spatial Change detection using two frames, in the next section.

3. 1.3 Spatial Change Detection Using Two Frames

The simplest method to detect changes between two registered frames would be to analyse the frame difference (FD) image, which is given by

$$FD_{k,r}(x) = s(x,k) - s(x,r) \quad (1)$$

where

- $x = (x_1, x_2)$
- $s(x,k)$ – represents the intensity value at pixel x in frame k.

FD shows the pixel-by-pixel difference between the current image *k* and the reference image *r*. The reference image *r* may be taken as the previous image k-1, the background image or an image at a fixed time. Methods using a fixed reference frame are called background subtraction methods (Chien et al. 2002). For example, in the practical aspect of the dissertation a room is monitored using a fixed camera. An image of the room when it is empty is used as the fixed reference image. Using a static camera and assuming the illumination remains more or less constant between the frames, computation of the pixel locations where $FD_{k,r}(x)$ differs from zero, indicating regions "changed" due to local motion is possible. To distinguish the nonzero differences that are due to noise from those that are due to local motion, segmentation can be achieved by **thresholding** the FD as:

$$z_{k,r}(x) = \begin{cases} 1 & if \ |FD_{k,r}(x)| > T \\ 0 & otherwise \end{cases} \quad (2)$$

where

- *T* is an appropriate threshold.
- $z_{k,r}(x)$ is called a segmentation label field, which is equal to "1" for changed regions and "0" otherwise. The value of the threshold *T* can be chosen by an optimal threshold determination algorithm. However in the practical aspect of the dissertation it was chosen by trial and error.

Pixelwise thresholding is followed by one or more postprocessing steps to eliminate isolated labels. Postprocessing methods included morphological filtering of the changed and unchanged region masks.

In practice FD image analysis is not satisfactory due to reasons such as misinterpretation of uniform intensity regions and variation in the pixel intensity difference due to motion in the

frame. The pixel intensity problems can be addressed by using a locally normalised FD function (Irani et al. 1994). However, this function was not utilised in the Kalman implementation in this dissertation for the following reasons:

- Motion tracking occurred in a controlled environment with large colour contrasting regions.
- The only motion in the environment was that of the tracked person.

A locally normalised FD/change detection algorithm addressing the shortcomings of FD can be constructed as follows:

(i) Construct a Gaussian pyramid where each frame is represented in multiple resolutions. Start processing at the lowest resolution level.
(ii) For each pixel at the present resolution level, compute the normalized FD. The FD is obtained from Irani et al. (1994) as:

- $$FDN_{k,r}(x) = \frac{\sum_{x\in\aleph}|s(x,k)-s(x,r)||\nabla s(x,r)|}{\sum_{x\in\aleph}|\nabla s(x,r)|^2+c} \quad (3)$$

where

- N denotes a local neighbourhood of the pixel **x**,
- $\nabla s(x,r)$ denotes the gradient of image intensity at pixel **x**
- c is a constant used to avoid numerical instability.

A high normalised difference indicates a moving pixel. For such an event, replace the normalised difference from the previous resolution level for that pixel with the new value. Otherwise, retain the value from the previous resolution level.
(iii) Repeat step ii) for all resolution levels.
(iv) Threshold the normalised motion detection function at the highest resolution level.

3.2 Region-Based Object Tracking in detail

For 2D region-based object tracking, a region can be defined as the projection of the tracked object onto the image plane. If the colour of the object to be tracked can be modelled and distinguished from other objects in the frame, then colour information can be very effective in region based object tracking. This can lead to real time object tracking of the order of 20 -30 frames per second (Nikolaidis et al. 2009).

The major difficulty concerning colour based object tracking is how to provide compensation or robustness to changes in illumination. One approach to address this problem would be to control the illumination conditions in an indoor environment. This was the approach used in this research dissertation. This is however not a true reflection of real world conditions as illumination outdoor or indoor is never constant. Other approaches to address illumination variance are colour correlation (Finlayson et al. 2001) and normalisation of the colour space (Gevers and Smeulders 2000).

Colour modelling can be classified as parametric or non-parametric. Parametric techniques use a single Gaussian or a mixture of Gaussians. This is the technique used for tracking in the practical part of the dissertation, due to simplicity of implementation. Non-parametric techniques , are histogram based such as Bayes skin probability maps. For non-parametric modelling methods , the HS colour spaces (HSV, HIS,HLS, etc) which are related to human perception of colour perform better in methods based on lookup tables.

A basic algorithm that allows efficient colour-based object tracking is chroma-keying (Starck and Hilton 2007). The background is a single colour and the object colours are very different from the background. Simple colour thresholding is then used to separate the object from the background and then track it.

Another approach often used in colour-related region based object tracking is one involving colour histograms (Merven et al. 2003). This approach falls in the category of non-parametric techniques. The colour histogram of the i-th region A_i is denoted by O_i. Initialisation steps are carried out in order to compute the colour histograms of all objects in the scene from a number of frames of video sequence. The computer histograms from initialisation are then stored in a database as reference colour histograms, denoted by O_i^r. In each new frame of a video sequence, a color histogram O_i^t, is calculated for each of the tracked objects. Each calculated

histogram O_i^t is then compared to the reference colour histogram O_i^r of the object in order to determine the best match and find the position of the tracked object in the current frame. Different criteria, such as the histogram intersection measure (Swain and Ballard 1991) which performs bin-by-bin comparison between two histograms and returns a relative match score based on the portion of pixels found within the same bin for each histogram, are used to measure histogram similarity. Reference histograms can however only deal with a fixed colour distribution, and are thus not able to handle changes in illumination. These changes in illumination can lead to tracking drift. A solution to this problem would be to update the bins from the histogram at intervals using information from the frames of the video sequence. This however comes at the expense of increased computational complexity.

A popular region based tracking system employing colour as a tracking mechanism is Pfinder (Wren et al. 1997). This algorithm is capable of tracking a single person in complex environments using a single fixed camera. It uses blob representation, which is coherent image regions where the pixels have similar properties. A feature vector is formed using the spatial coordinates (x,y) and colour components of each pixel in a blob. The initialisation step is used to form a model of the background for the algorithm. As a person enters the scene large changes in the scene are detected and a statistical blob model is constructed. The spatial distribution of each blob in subsequent frames is predicted using Kalman filters. The likelihood that a pixel belongs to a certain blob is evaluated. The likelihood is then used to assign the pixel to a certain blob or the background. Finally the statistical model of each blob is updated.

A popular region-based tracking systems also implemented in this dissertation is based on background subtraction (Aggarwal 1996). This approach concerns building a model of the background rather than the object. The approach used in this dissertation takes the model of the scene to be a single image acquired during the initialisation step. The background is assumed static during the progression of the video sequences. Each new frame in the video sequence is subtracted from the static scene model in order to obtain a segmented foreground, containing moving or new objects. The subtraction itself is not accurate due to noisy measurements obtained by the camera as well as changes in the scene environment. Noisy measurements are due to external influences such as illumination changes, motion caused by wind blowing objects in the environment, moving clouds etc. The problem is more extreme in outdoor environments. In this research a Kalman filter is implemented to provide robust compensation against noisy measurements.

Other approaches to modelling the background involve modelling it by a different 3D Gaussian whose mean and covariance are evaluated from a number of consecutive frames depicting only the background. The model of the background is thus obtained and continuously updated by means of using a simple adaptive filter incorporated within the tracking algorithm. The likelihood that each pixel's colour belongs to the colour distribution of a pixel from the background model is evaluated. If the likelihood is large the pixel is labelled a background pixel and vice versa.

A simple algorithm involving background subtraction is presented by (Fuentesa and Velastin 2006). This method also uses one image as the reference background model and no updates are performed on the background model. When a human or new object enters the scene, foreground pixels are detected in each frame using the luminance contrast between frame and the background image. Pixels are grouped in blobs represented as bounding boxes. Blobs detected in two consecutive frames using their match as an overlapping criterion. Matching matrices are used for the tracking and handling of blob splitting/merging and blob entering/leaving a scene. These matrices carry information regarding the correspondence between blobs in two consecutive frames.

3.3 Bayesian Object Tracking

Many object tracking algorithms perform tracking within a Bayesian framework. The Bayesian framework belongs to a class of state space techniques which estimate the state of the system over discrete steps. These techniques assume that noisy measurements w_k are available at each of these discrete time steps.

Firstly the approach requires a state vector x_k to be devised. This vector contains all the data required to describe the system. For tracking a moving person in two dimensions, the state vector would typically consist of the x and y coordinates of the person's position in each frame, as well as the velocity and acceleration along each coordinate.

Secondly a measurement vector z_k is devised. The measurement vector contains "noisy" object positions obtained via the camera.

Thirdly a system model is required in order to perform estimation and hence describe the evolution of movement of the object state over time. The non-linear representation of the system model is as follows:

- $x_k = f_k(x_{k-1}, w_{k-1}) \ldots\ldots x_k \in R^n$ (4)

where,

- f_k represents a dynamic function mapping the system state x_k and the noise/disturbance state w_k at time k to a new system state x_{k+1} at time $k+1$.
- $x_k \in R^n$ implies x_k is an element of the n dimensional real space R^n. The state variable vector x_k summarises the relevant information from the system at any time and describes how the system changes as a function of time and input (Freeston, 2002). The state variable vector contains information on the tracked objects x, y and z coordinates. It represents the tracked object at time index k .
- w_k represents the system input noise or disturbances in the model. In statistical terms w_k is said to have a normal distribution with a mean distribution of 0. The zero mean is an indication that the average size of the noise is zero.

The linear representation of the system model is as follows:

- $x_k = A_{k-1}x_{k-1} + w_{k-1}$ (5)

where,

- A_k is an n x n state transition matrix relating the state x_{k-1} in the previous step *k-1* to the current state x_k at step index k. A_{k-1} is of size $n \times n$ as it maps the state x_{k-1} which is an element of n dimensional real space R^n.

Fourthly a measurement model is required to link the noisy measurements to the state vector.

The non-linear measurement model is as follows:

- $z_k = h_k(x_k, v_k)$, (6)

where,

- h_k represents a dynamic function mapping the system state x_k and the observation noise/disturbance state v_k at time k to a new measurement state z_k at time k.

The linear measurement model is as follows:

$z_k = H_k x_k + v_k \qquad for\ k = 0 \ldots . N - 1$ (7)

where,

- H_k is a matrix relating the state variable x_k to the measurement variable z_k.
- v_0 is the random measurement or observation disturbance noise. This is the noise due to the measurements having been taken. This noise has a zero mean and is considered to be independent from the effects of the input noise w_k.

Two factors have to be considered in order to perform object tracking in a Bayesian framework. These factors are:

1. Measurement and system models should be available in probabilistic form.
2. In object tracking, an estimate of the object position is required each time a measurement becomes available.

Bayesian object tracking produces a state estimate at each time step k based only on all past measurements z_k up to time k. For Bayesian tracking the assumption is made that the probability density function (pdf) of the state vector $p(x_0|z_0)$ is known ,

where,

- x_0 is the initialized state vector
- z_0 is the set containing no measurements

The probability density function of the state vector $p(x_0|z_0)$ simply tries to determine the probability of state vector x_0, when the probability of the measurement vector z_0 is known.

The goal of Bayes Theorem is thus to obtain the posterior pdf $p(x_k|z_k)$ at time step k (Statistical Glossary),

where

- x_k is the state vector at present time step k
- z_k is the set of measurements at present time step k.

Assume that equation 4, the system model, is defined by the probability density function $p(x_k|x_{k-1})$, with process noise w_k

where,

- $p(x_k|x_{k-1})$ implies x_k is the state vector at present time step k provided that the state vector x_{k-1} is known at the previous time step of $k-1$.

Also assume equation 4, the measurement model/likelihood function, is defined by $p(z_k|x_k)$ with measurement noise v_k.

where,

- $p(z_k|x_k)$ implies, the probability of the set of measurements z_k at time step k , provided the state vector x_k is known for time step k

The state estimation process consists of two steps. The first step is prediction (time-update) and the second is update (corrector). During the prediction step, the posterior pdf at time step $k-1$, $p(x_{k-1}|z_{k-1})$, is propogated forward in time , using the system model $p(x_k|x_{k-1})$ (Arulampalam et al. 2002), in order to obtain the prior pdf $p(x_k|z_{k-1})$:

- $p(x_k|z_{k-1}) = \int p(x_k|x_{k-1})p(x_{k-1}|z_{k-1})dx_{k-1}$ (8)

Thus the prior pdf $p(x_k|z_{k-1})$ is obtained at time step k.

where,

- $p(x_k|z_{k-1})$ implies, the probability of system state x_k is at time step k , provided the measurement vector z_k is known for time step $k-1$

The second step, update, modifies the prior state by using the latest measurement data available obtained from the prediction step. The desired posterior pdf $p(x_k|z_k)$ is obtained using Bayes' theorem:

- $p(x_k|z_{k-1}) = \frac{p(z_k|x_k)p(x_k|z_{k-1})}{p(z_k|z_{k-1})}$ (9)

where ,

- $p(z_k|z_{k-1})$ in the denominator is used for normalisation and calculated as follows (Arulampalam et al. 2002):
  $p(z_k|z_{k-1}) = \int p(z_k|x_k)p(x_k|z_{k-1})dx_k$ (10)

Solutions to equations 9 and 10 can be obtained by means of the standard Kalman filter or by approximations using an extended Kalman filter (EKF) or particle filter (PF) (Moral, 1996)

3.2.1 Kalman Filters and Extended Kalman Filters

The Kalman filter is a Bayesian filter and is the best possible estimator provided that the following conditions hold:

- Function f and h in equations 4 and 6 are linear and known
- The distribution of the measurement and process noises are Gaussian. The Gaussian noise has a normal probability distribution, implying that it can be described in terms of its mean and variance. The Gaussian noise must have a zero mean value and a standard deviation σ.
- The posterior pdf is Gaussian

3.2.1.1 The Kalman Filter

The Kalman filter is a recursive solution to a discrete data filtering problem. It provides an efficient computational means to the state of a process. This is achieved by minimising the mean of the squared error (Welch and Bishop 2001).

Linear stochastic difference equations describe both the system model (evolution of the object state over time) and the measurement model $z_k \in R^m$:

- $x_{k+1} = A_k x_k + B_k u_k + w_k$ (11)
- $z_k = H_k x_k + v_k$ (12)

where in equation 11

- B_k is an $n \times l$ matrix relating the optimal control or input $u_k = \mu_k(x_k)$ to the state x_k.
- A controller $u_k = \mu_k(x_k)$ acts on the uncertain input data x_k in order to produce more accurate system input data at x_{k+1}. This is shown in figure 5 on the next page.

Equations 11 and 12 are depicted in figure 5 on the next page.

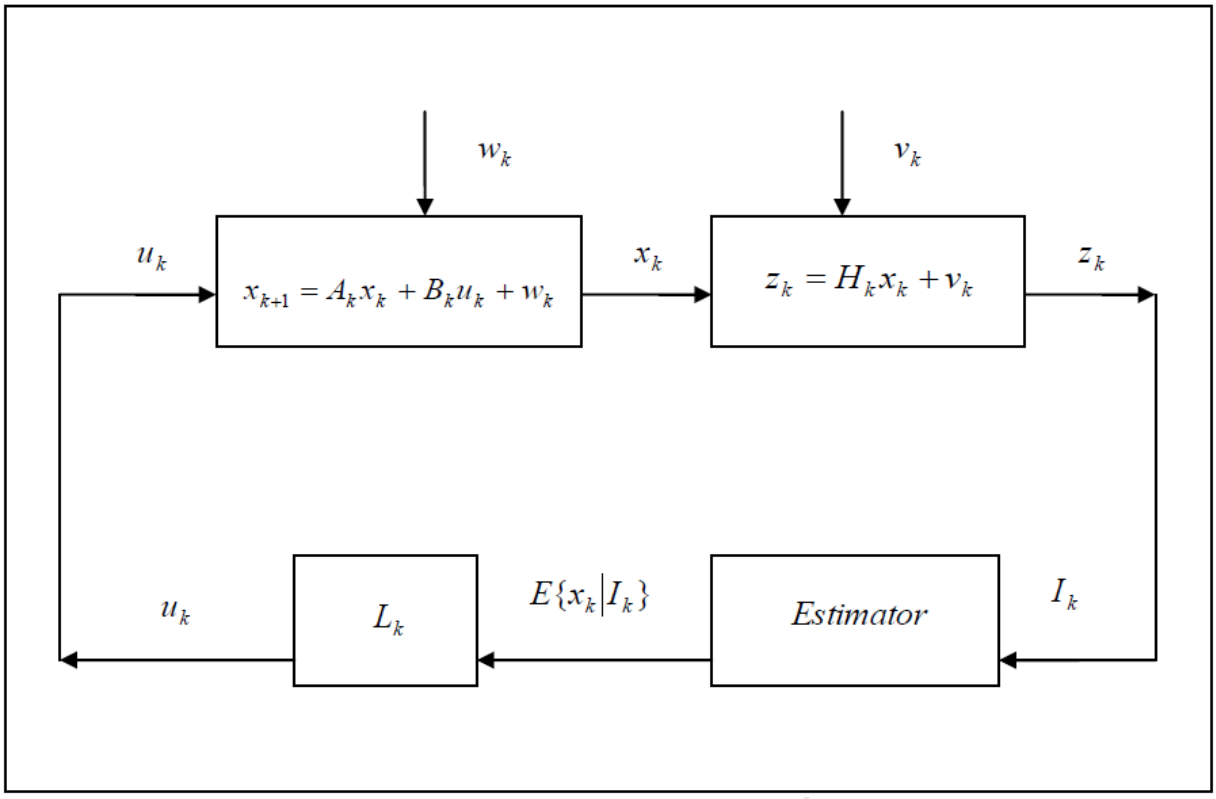


Figure 5. Imperfect State Representation of Tracking System

For the linear Kalman filter implementation w_k and v_k have the following relationship and properties:

- Independent- implies that they do not effect each other
- White – implies that the mean of each type of noise is zero
- Normal probability distributions - implies that each of these two types of noise can be described in terms of their mean and variance. The mean is considered to be zero for these two types of noise.

  Thus the probability distribution for the input noise w_k can be approximated as:

  $p(w_k) \approx N(0, Q)$

  with,

- 0 being the mean value of the noise
- The input noise covariance matrix Q.

The Input Noise Covariance matrix Q_k is a measure of the magnitude of noise present in the input data. *The* expression for the input noise covariance matrix Q_k is:

- $$Q = \begin{bmatrix} \sigma_x & \sigma_{xy} & \sigma_{xx'} & \sigma_{xy'} \\ \sigma_{xy} & \sigma_y & \sigma_{yx'} & \sigma_{yy'} \\ \sigma_{x'x} & \sigma_{x'y} & \sigma_{x'} & \sigma_{x'y'} \\ \sigma_{y'x} & \sigma_{y'y} & \sigma_{y'x'} & \sigma_{y'} \end{bmatrix}$$

The nth diagonal element of the covariance matrix is a measure of the variance of the nth element of the system state x_k from its means. The off diagonal elements are the cross covariance of the variables of state vector x_k . For tracking purposes the Input Noise Covariance matrix Q_k is simplified to only express the variation of the state variables from the state average or mean. By only considering the diagonal elements, the Input Noise Covariance matrix Q_k is simplified to the following matrix expression:

- $$Q = \begin{bmatrix} \sigma_x & 0 & 0 & 0 \\ 0 & \sigma_y & 0 & 0 \\ 0 & 0 & \sigma_{x'} & 0 \\ 0 & 0 & 0 & \sigma_{y'} \end{bmatrix}$$

The variables of interest are identified by the subscript of the standard deviation symbol σ.

The probability distribution for the measurement noise v_k can be approximated as:

$$p(v_k) \approx N(0, R)$$

with,

- 0 being the mean value of the noise
- Variance of the noise given by the measurement noise covariance matrix R .

In practice Q, R, A, B and H may change with each time index k.

3.2.2 Computational Implementation of the Kalman Filter

In order to represent the Kalman filter in a computational environment two terms have to be introduced and elaborated upon, namely "*a priori*" and "*a posteriori*". The Kalman filter operates in a two step predictor-corrector manner. During the predictor step, the current estimate along with an estimate of the error covariance matrix are propagated forward in time. The corrector state incorporates a new measurement in order to modify the propagated current state and error covariance estimates. The *a priori* state estimate at time index *k* (based on all past measurement prior to step k) is denoted $\hat{x}_k^- \in R^{n_x}$,

where

- $\hat{x}_k^-$ is an element of the n dimensional real space R^{n_x}

A priori knowledge refers to the prior knowledge of the system. The a priori state estimate thus refers to prior knowledge of state x_k, before an estimate was formed. This is the probability estimate of the state before receiving new information (Statistical Glossary).

A posteriori estimate is a revised probability estimate that takes into account new information from measurements that have been taken (Statistical Glossary)

The a posteriori state estimate at time index k (as soon as measurement z_k becomes available) is defined as:

$\hat{x}_k \in R^{n_x}$

where,

- $\hat{x}_k$ is an element of the n dimensional real space R

The information obtained from the a priori and *a posteriori* state estimates, allow *a priori* and *a posteriori* error state estimates to be determined.

The a priori state estimate error $\hat{e}_k$ can be obtained by subtracting the a priori state estimate from the actual state x_k .

- $\hat{e}_k \equiv x_k - \hat{x}_k^-$ (13)

The *a posteriori* state estimate error e_k can be obtained by subtracting the *a posteriori* state estimate $\hat{x}_k$ from the actual state estimate x_k.

- $e_k \equiv x_k - \hat{x}_k$ (14)

An important Kalman filter parameter is the error covariance matrix P . The error is defined as the difference between the actual state variables and the estimated state variables. The error covariance matrix P is thus a measure of the uncertainty present in the estimate. The square root of the diagonal elements in P give the error present in the variable estimates.

The square root of the variances gives the standard deviation σ of the error present in those variables.

The a priori error covariance matrix P_k^- gives the uncertainty present in the a priori state

estimates P_k^- (based on all past measurement prior to step k) is determined as follows:

- $P_k^- = E[e_k^- e_k^{-T}]$ (15)

The *a posteriori* error covariance matrix P_k gives the uncertainty present in the *a posteriori* state estimates. P_k is determined as follows:

- $P_k = E[e_k e_k^T]$ (16)

In both equation (15) and (16), the error covariance matrix is obtained by finding the expectation, E , of the error " e" multiplied by its transpose "e^T". The expectation E is defined as the mean or average of a result (Pinkney, 2003).

The *a priori* error covariance matrix P_k^- has an important role in the functioning of the

Kalman filter. It is used in the calculation of the Kalman gain factor K_k for each time index k. K_k is chosen as an $n\ x\ m$ matrix,

where,

- n represents the n dimensional real space of $x_{k+1} \in R^n$
- m represents the m dimensional real space of $z \in R^m$

The Kalman gain factor K_k is chosen in a manner that allows for the minimisation of the *a posteriori* error covariance matrix P_k . The gain factor K_k that minimises the a posteriori error covariance is given by (Maybeck 1979; Brown 1992; Jacobs 1993):

- $K_k = \frac{P_k^- H^T}{HP_k^- H^T + R}$ (17)

where,

- K_k is the Kalman gain factor
- P_k^- is the a priori error covariance matrix
- H is the measurement prediction
- R is the measurement noise covariance

Knowing the Kalman gain factor K_k and the measurement prediction $H\hat{x}_k^-$, an a posteriori state estimate $\hat{x}_k$ can be obtained. This is a linear combination of an a priori state estimate $\hat{x}_k^-$ and a weighed difference between an actual measurement z_k and a measurement prediction $H\hat{x}_k^-$. This linear combination is shown in equation 18:

- $\hat{x}_k = \hat{x}_k^- + K(z_k - H\hat{x}_k^-)$ (18)

In equation 18 the term $(z_k - H\hat{x}_k^-)$ is called the residual. The residual indicated the difference between the predicted measurement $H\hat{x}_k^-$ and the actual measurement z_k.

Where the residual is equal to zero, it implies that the predicted and actual measurements are exactly the same. Thus for a zero residual there are no errors in the measurements that have been taken.

Minimising the result for the gain factor K_k of equation 17 is very significant in that it shows: The minimising results for the gain K_k indicate that as the measurement covariance R approaches zero the values of the actual measurement z_k become more reliable than the predicted measurement $H\hat{x}_k^-$. As the a prior estimate error covariance P_k^- approaches zero, the values of the predicted measurement $H\hat{x}_k^-$ become more reliable than the values of the actual measurement Z_k.

3.2.3 The Computer Algorithm for the Linear Kalman Filter

The Kalman filter operates by estimating the state at some point in time and then obtains feedback measurements containing noise. The feedback measurements are then incorporated into the state estimates in order to form an update. The Kalman filter thus falls into two categories:

i. Time update equations (Predictor Equations)

ii. Measurement update equations (Corrector Equations)

Time update equations project forward in time, the current statex_k and error covarianceP_k, in order to obtain an *a priori* estimate for the next state. For this reason time update equations are referred to as predictor equations.

The discrete Kalman filter time update equations are:

- $\hat{x}_{k+1}^- = A_k\hat{x}_k + B_k u_k$ (19)
- $P_{k+1}^- = AP_kA^T + Q$ (20)

Measurement update equations incorporate a new measurement via feedback into an *a priori* estimate in order to achieve an *a posteriori* estimate. For this reason measurement update equations are called corrector equations.

The discrete Kalman filter measurement update equations are shown below:

- $K_k = \frac{P_k^- H^T}{HP_k^- H^T + R}$ (21)
- $\hat{x}_k = \hat{x}_k^- + K_k(z_k - H\hat{x}_k^-)$ (22)
- $P_k = (I - K_k H)P^-$ (23)

Each time and measurement update allows the previous *a posteriori* estimate at index k+1 to be used in the prediction of the new *a priori* estimate at index k . The advantage of the Kalman filter is that it conditions the current estimate on all past values of the measurements. This process is described in figure 6 on the next page.

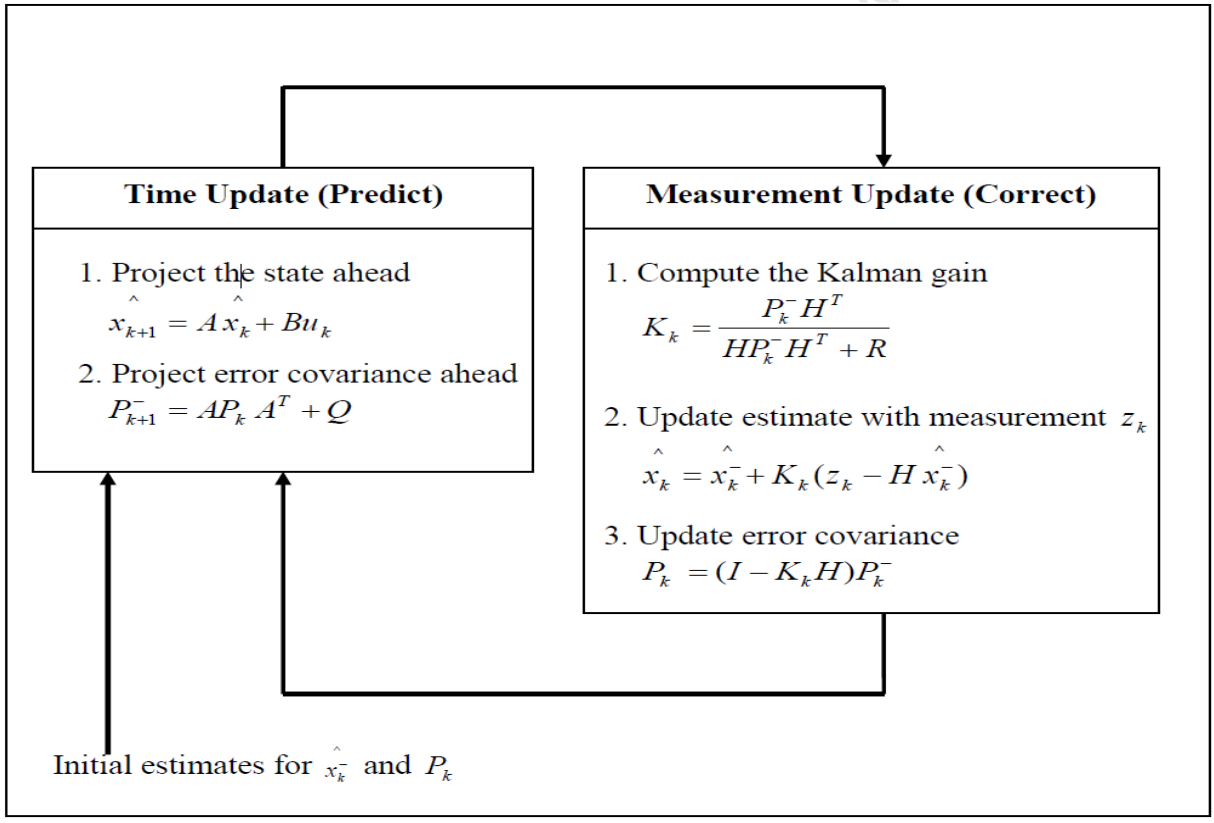


Figure 6. Operation of the Discrete Linear Kalman Filter

Figure 6 shows that an initial state estimate $\hat{x}_k^-$ and error covariance matrix P_k are placed into the Kalman filter at time index k. The initial values at time index k, are then used to determine an *a priori* state $\hat{x}_{k+1}^-$ estimate and an *a priori* error covariance matrix P_{k+1}^- estimate for time index k+1. This occurs for the time update or predictor part of the Kalman filter.

The a priori state estimate $\hat{x}_{k+1}^{-}$ and an a priori error covariance matrix estimate P_{k+1}^{-} for time index $k+1$are then updated with measurements passed into the Kalman filter. This is the corrector part of the Kalman filter. The corrector part of the Kalman filter computes the Kalman gain factor K_k which it then uses to update estimates, with the measurement obtained from the measurement vector z_k. The Kalman gain factor $K_k$and the measurement vector z_k are then used to determine a state estimate x_k and error covariance matrix P_k estimate which takes into account measurement data. The new state x_k estimate and the new error covariance matrix P_k estimate, are now the new inputs for the Kalman filter at a new time index k . This data is then used to determine an a priori state $\hat{x}_{k+1}^{-}$ estimate and an a priori error covariance matrix P_{k+1}^{-} estimate for time index $k+1$. This cycle continues depending on the number of iterations required.

3.2.4 Kalman Filter Parameter Selection and Tuning

The measurement noise covariance R is measured prior to the operation of the filter. This allows offline samples of the measurements to be obtained in order to determine the variance of the measurement noise.

Obtaining the process noise covariance Q is more complicated as no means exists to observe the process being estimated. The selection of Q allows the manipulation or injection of uncertainty into the process.

Advanced methods achieve tuning of R and Q though the implementation of another Kalman filter in a process referred to as System identification.

3.2.5 Effects of R, P, Q and K on Kalman Tuning

The measurement covariance matrix *R* determines the amount of data usage from the sample. When *R* is high there is too much noise in the measurement and thus the Kalman filter does not follow the measurement data that closely as that measurement is inaccurate.

The effect of the input covariance matrix *Q* is that it determines the performance of the

Kalman filter estimate, by giving an indication of the accuracy of the state estimate x . Thus for small values of the input covariance matrix Q , the Kalman filter utilises less data from the measurement as it is certain of its estimate.

The error covariance matrix P is reduced by accurate measurement data. Errors in the estimate are reduced, if measurement data exists to reinforce the accuracy of the estimate. The reduction of the P is however restricted by the input covariance matrix Q , which is added at each step in the computation of P , as shown by equation 22. Equation 23, shows that R and P are included into the Kalman filter through the gain K . The gain K is the determining factor, in how much the innovation is used to correct the estimate of the Kalman. The innovation is the difference between the actual measurement and the measurement obtained from the model. The Kalman gain K is directly proportional to the error covariance matrix P and inversely proportional to the measurement noise covariance matrix R. Thus if R is large compared to P , the gain would be small and so would the innovation. If P is large compared to R , the gain K would be larger and hence the innovation. Thus the certainty of the estimate for a large P is not that accurate, and requires adjustment to the estimate from the Kalman filter. For large P the Kalman filter thus incorporates more data from the measurement.

The input covariance matrix Q contributes to the uncertainty of the model due to its inclusion in each step of the calculation of the error covariance matrix P as indicated by equation 22. The Kalman filter output for large values of Q tracks the output more closely, then for smaller values. However large values of Q introduce a lot of noise into the final output measurements. Thus a compromise has to be established between tracking performance and noise in the output.

3.2.6 Extended Kalman Filter

For a number of object tracking and computer vision problems the state and measurement models might not be linear. For such situations the Kalman filter cannot be utilised unless a means exists to linearises both the state and measurement models. Thus a Kalman filter has to be employed which linearises about the current mean and covariance. This is known as the Extended Kalman filter (Zarchan and Musoff 2001). Let w_k and v_k represent the process and measurement noise respectively. The system model describing the evolution of the state over time can be given as:

- $x_k = f(x_{k-1}, u_{k-1}, w_{k-1})$ (24)

The measurement model is given by z ∈ R^m:

- $z_k = h(x_k, v_k)$ (25)

Both the measurement and state model are nonlinear. The difference in the formulation of the state and measurement vectors for the Extended Kalman filter is that f and h are nonlinear.

The process and measurement noise are not known, yet the state and measurement vector can still be approximated as:

- $\tilde{x}_k = f(\hat{x}_k, u_{k-1}, o)$ (26)
- $\tilde{z}_k = h(\tilde{x}_k, 0)$, (27)

where

- $\tilde{x}_k$ is the a posteriori estimate of the state from a previous time step

Following the linearisation and derivation process identified by Welch and Bishop (1995), the equations for the first step (prediction) are:

- $\tilde{x}_k = f(\hat{x}_k, u_{k-1}, o)$ (28)
- $P_k^- = A_{k-1} P_{k-1} A_{k-1}^T + W_{k-1} Q_{k-1} W_{k-1}^T$ (29)

where

- A_k is the Jacobian matrix of the partial derivatives of $f(.)$ with respect to x_k
- W_k is Jacobian matrix of the partial derivatives of $f(.)$ with respect to w_k
- $Q_k$is the process noise covariance

The second step (update) equations are:

- $K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1}$ (30)
- $\hat{x}_k = \tilde{x}_k + K_k(z_k - h_k(\tilde{x}_k, 0))$, (31)
- $P_k = (I - K_k H_k) P_k^-$ (32)

where

- H_k is the Jacobian matrix of the partial derivatives of $h(.)$ with respect to x_k
- V_k is Jacobian matrix of the partial derivatives of $h(.)$ with respect to v_k
- R_k is the measurement noise covariance

The Extended Kalman filter uses first order polynomials and as a consequence eventually introduces errors. This problem can be overcome with an implementation of the Unscented Kalman Filter, which is an improvement of the Extended Kalman Filter. The Unscented Kalman Filter makes use of third and higher order terms of the Taylor series.

3.2.7 Particle Filters

If the posterior pdf is not Gaussian, Kalman filters will not perform adequately. For these cases Particle Filters are utilised. A Particle Filter is a sequential Monte Carlo method that can be used for object tracking within a Bayesian framework. The Particle filter represents the probability distribution of alternative solutions as a set of samples (i.e. particles), each of which carries a weight. Estimates of the posterior distribution are calculated based on these samples and their associated weights. As the number of samples grows, the filter approaches the optimal Bayesian estimate (Arulampalam et al. 2002). Ideally sampling would be done on the posterior distribution, but due to its complexity this is not the case. Thus sampling techniques such as factored sampling is carried out on the prior pdf. Factored sampling is a random sampling technique assigning a weight to each random sample. The weighed set then becomes the approximation to the posterior density. Another sampling technique, importance sampling, does not sample from the prior pdf, but from another density function that can "drive" the selection of samples towards areas of posterior pdf. Hence, the posterior pdf can be described more accurately. The main advantage of Particle Filters are that they allow the fusion/merging of information from different sources (Perez et al. 2004).

4. The Tracking System

4.1 Implementation of the Kalman Filter as a Tracker

Person tracking in real time can find applications in surveillance and monitoring systems (Merven et al., 2003). In this case the tracking problem is approached in a probabilistic manner. The implementation takes the form of a Kalman Filter, which acts as a state estimator and whose estimation property is utilised as a tracker. A block based motion – compensated prediction approach is utilised. The procedure involves block matching techniques, where a macro block has to be determined in a reference frame. The reference frame can either be a past reference frame to allow for future estimation or a future reference frame to allow backward estimation (Karna et al., 2007)).

This example involves tracking a single person in 3D space with a camera, with the tracking results projected onto an x , y coordinate system.

4.2 Tracking Methodology

4.2.1 Equipment

- Camera

  A Canon PowerShot A620, 7.1 megapixel digital camera was used as the tracking mechanism. The tracking resolution of the camera when in video/movie mode is 640/480. The camera outputs 3.7 frames per second.

- PC

  Camera is linked to a Dell Latitude D620 notebook. The notebook runs Windows XP, has a 2.0 GHz dual core processor and 2 Gigabytes physical RAM.

- Software

  The software used to create the computational environment was Matlab 7.1.

4.2.2 State Space Representation

The three dimensional view of the room containing the camera is shown in figure 7. The space marked in yellow is the area within the cameras view.

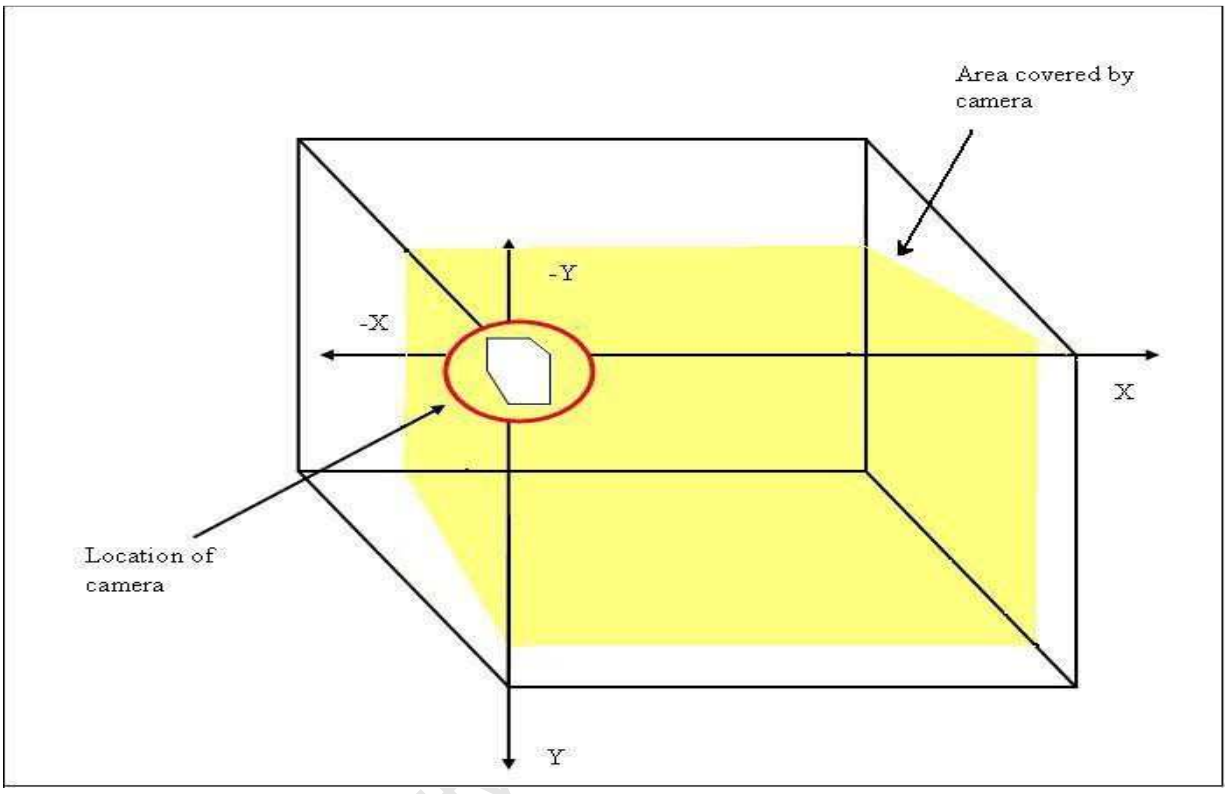


Figure 7. Projection to image space.

Figure 7 shows x and y coordinates of the front plane as this will be used to represent tracking in a two dimensional view. The three dimensional image of figure 7 is converted to a two dimensional representation, as shown in figure 8 on the next page.

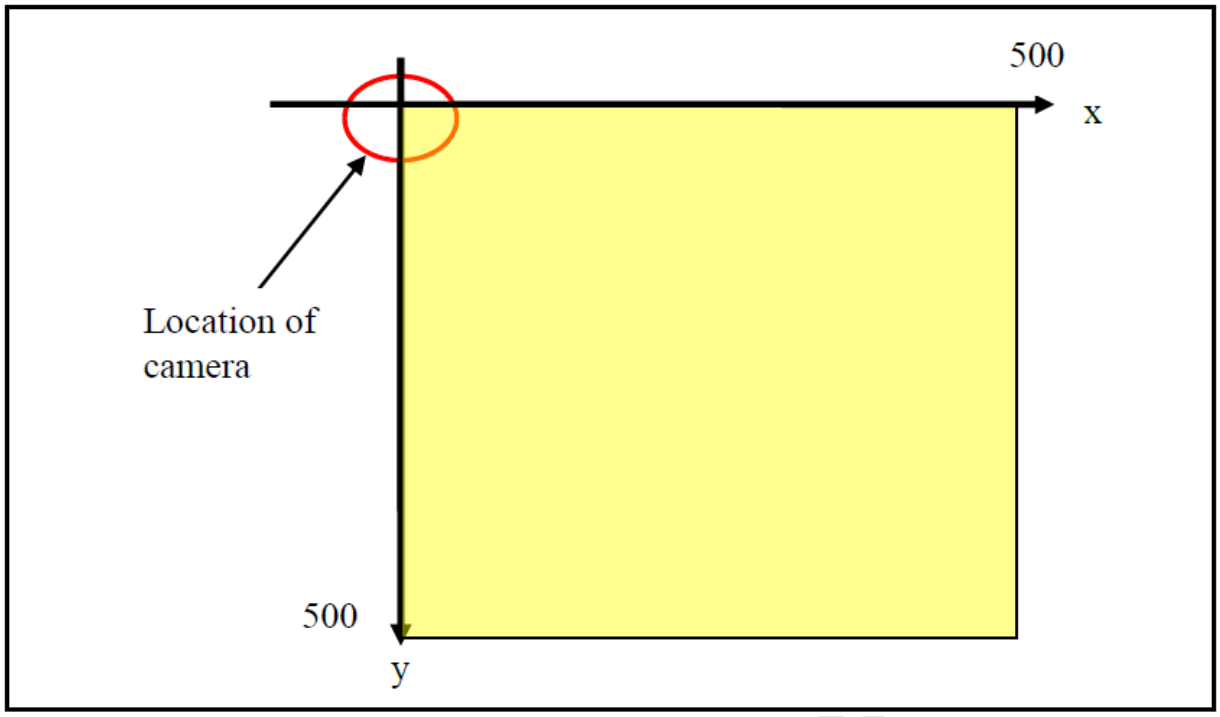


Figure 8. The 2 dimensional plot of the room as seen by the camera

Figure 8 shows a two dimensional plot of the area seen by the camera through the front plane of figure 7. It is a two dimensional plot of the front yellow plane of figure 7. The two dimensional figure represents a frame or picture taken by the camera. The limits of the two dimensional spatial coordinate plane in the MatLab computational environment for each axis are:

- X axis (0-500)
- Y axis (0-500)

4.2.3 Method Overview

The Kalman filter tracking method first involves segmentation in order to identify the moving object. The Kalman filter is then initialised through the parameter selection. The location of the foreign object which has to be tracked is then predicted and the filter carries out a series of update equations in order to determine an accurate location. Finally the object's location is projected into image space.

4.2.3.1 Segmentation

In the analysis of the objects in images it is essential that we can distinguish between the objects of interest and "the rest." This latter group is also referred to as the background. The techniques that are used to find the objects of interest are usually referred to as segmentation techniques - segmenting the foreground from background (Chien et al. 2002).

Segmentation works by first obtaining an image of the background to be monitored. This is the reference image. Once a new object enters the monitored background, the object is added to the reference image. In order to identify the tracked object which occurs in the foreground, the reference background image is subtracted from the background image containing the tracked object. This separates the moving tracked object from the rest of the image.

4.2.3.2 Determining the parameters

The initial conditions and parameters are determined for the Kalman filter in order to initialise the tracking system. Thus initial conditions are determined for the time step Δt ,
the input distance B.u , and the state transition matrix A . An initial approximation also has to be determined for the state vector x^{approx} and measurement vector z^{approx}.

4.2.3.3 Prediction

The recursive operation of the Kalman filter involves the repeating cycle of correction and prediction using equations 11 and 12 (Welch, 2009). When measurement z^k becomes available at time step k, one predicts the a priori state $\hat{x}_k^-$ and error covariance P_k^- using the Kalman time update equations 19 and 20:

- $\hat{x}_{k+1} = A\hat{x}_k + Bu_k$ (19)
- $P_{k+1}^- = AP_kA^T + Q$ (20)

4.2.3.4 Update

After Kalman time measurement update or prediction occurs , the gain matrix $K \in R^{n\times m}$ is then computed using equation 20 as:

- $K_k = \frac{P_k^- H^T}{HP_k^- H^T + R}$ (21)

The a priori state $\hat{x}_k^-$ and error covariance P_k^- are then corrected in order to obtain the a posteriori state $\hat{x}_k$ and error covariance $P_k$as stated in equations 22 and 23 as:

- $\hat{x}_k = \hat{x}_k^-(z_k + H\hat{x}_k^-)$ (22)
- $P_k = (I - K_k H)P_k^-$ (23)

4.2.3.5 Projection to image space

The path taken by the tracked object is shown as a 2 dimensional plot on an (x, y) axis. The introduction into the implementation of a tracking mechanism in figure 2 and figure 7 deal with a 3 dimensional plot. A 2 dimensional spatial coordinate plot is however used in this implementation to facilitate ease of computation.

4.3 Segmentation to Identify the Foreground

A central problem called segmentation discussed in section 3.1, is to distinguish objects from the background (Asano et. al, 1996). Segmentation subdivides an image into its constitute regions or objects (Yang, 2004). According to Sonka et al. (2008) as cited by Moravčík (2009), the main goal of segmentation is to divide an image into parts that have a strong correlation with objects or areas of the real world contained within the image. In image processing useful pixels within the image are separated from the rest. The result of image segmentation is a set of segments that collectively cover the entire image or set of contours extracted from the image.

Threshold and edge detection are two of the most common image segmentation techniques (Moravčík, 2009). The method utilised in this research involves fixed threshold detection discussed in detail in section 3.1. MatLab provides functions to perform this segmentation. In brightness threshold all the pixels brighter than a specified brightness are taken as 1 and the rest are left 0. However for threshold segmentation to be truly effective sufficient contrast between objects and the background is required (Sonka, 1992). Thresholding can thus be thought of as a transformation of an image into a binary image g(i, j). The function involved in this transformation was discussed in section 3.1.3 and is as follows:

$$g(i,j) = 1 \quad for \quad f(i,j) \geq T$$
$$= 0 \quad for \quad f(i,j) < T \tag{26}$$

where;

- T is a user specified threshold
- g(i, j) = 1 for the segmented foreground image regions.
- $g(i,\ j)$ = 0 for background regions.

The central question in thresholding then becomes, how does one choose the threshold? There is no universal procedure for threshold selection that is guaranteed to work on all images; there are a variety of alternatives (Tekalp, 2009). In this research "Fixed Threshold" is utilised. This method entails using a threshold that is chosen independently of the data. Thus a constant threshold on a scale from 0 to 255 can be selected in such a way to keep the amount of falsely classified pixels to a minimum.

For this research an image of a room without any moving objects is taken. This image shall be used as the “background” image. This background image is shown in figure 9.

Figure 9. The background Image

Figure 9 displays the monitored room without any disturbances or changes. This is thus used as the reference or background image and is the constant area to be monitored. The foreground images would thus be all other images. The foreground images will then be compared to the background image. In figure 10, a foreground image is shown containing a person. This image can be thought of as being the background image with a person added
to it.

Figure 10. New moving object in the background

Having obtained the background, image thresholding is applied to all new images of the room in order to achieve segmentation, with the aim of separating foreign objects from the background image of the room. Using MatLab segmentation is achieved by subtracting the indexed foreground image from the indexed background image. A MatLab indexed image consists of two arrays, namely an image matrix and a colour map. The colourmap is an ordered set of values that represent the colours in the image. For each image pixel, the image matrix contains a value that is an index into the colormap. The colourmap is an m-by-3 matrix of class double. Each row of the colourmap matrix specifies the red, green, and blue (RGB) values for a single colour (Mathworks, 1998): colour = [R G B].

In this implementation with MatLab, the background image variable is Imback and the foreground image variable is Imwork. These image variables are utilised in the code shown in the Appendix.

Thus for the background image:

```
red = Imback(:,:,1);
green = Imback(:,:,2);
blue = Imback(:,:,3);
```

For the foreground image:

```
red = Imwork(:,:,1);
green = Imwork(:,:,2);
blue = Imwork(:,:,3);
```

To achieve threshold segmentation equation 26 is utilised. In this implementation with MatLab for equation 26;

$$f(i,j) = |abs(Imwork(:,:,1) - Imback(:,:,1) > 37)|$$

```
|(abs(Imwork(:,:,2) - Imback(:, :,2)) > 37)
|(abs(Imwork(:,:,3) - Imback(:, :,3)) > 37)
```

where;

- *abs*() is a MatLab function which returns a real integer
- $T = 37$ is a threshold T value of 37
- $g(i, j) = 1$ for all values exceeding $T = 37$
- $g(i, j) = 0$ for any values less than or equal to $T = 37$

After numerous implementations, trial and error resulted in a threshold T value of 37 being chosen. T =37 was chosen as it is the maximum limit in this implementation which allows the object to be displayed during segmentation without major distortions in the segmented image. For threshold T values greater than 37, the segmented image is mainly dark and it is impossible to see the tracked object of interest. For threshold T values less than 37, there are too many distortions in the segmented image and it is impossible to see the tracked object. For a Threshold T value of 37, the object of interest is shown in the segmented image in figure 11.

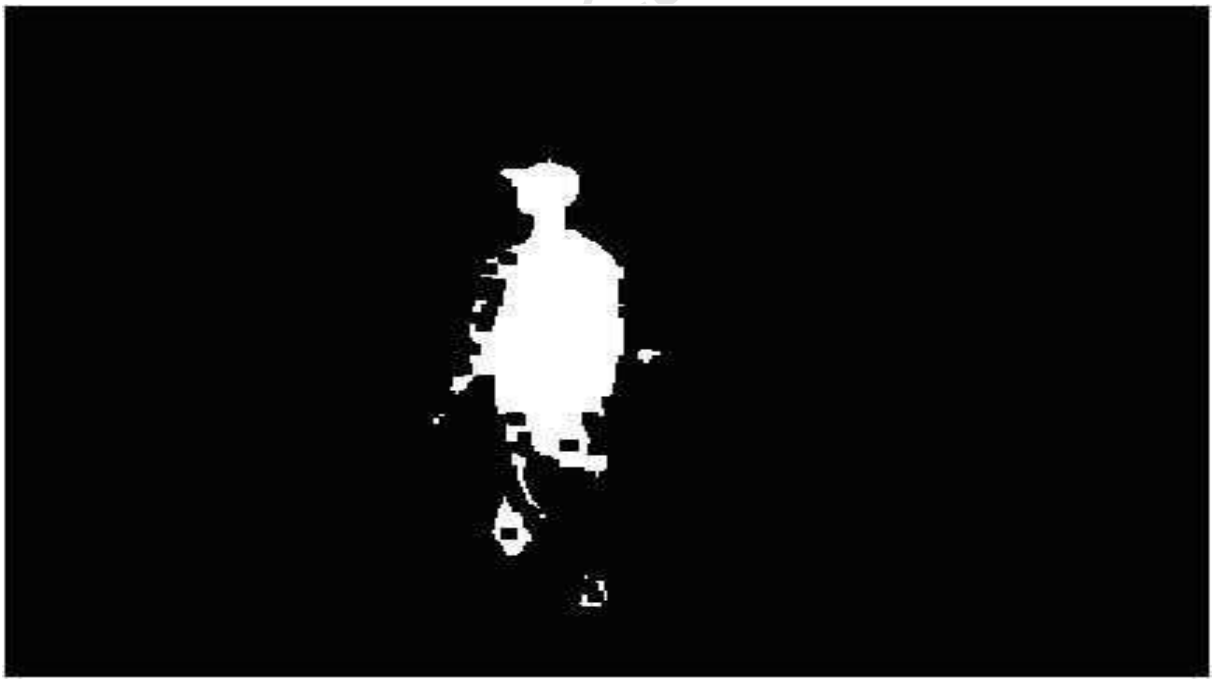


Figure 11. Segmented image for T = 37

Figure 11 is an image of the highlighted foreground pixels obtained as a result of subtracting the background image with an object to be tracked from a background image without that

object. The result of this action is displayed in figure 11. The shape of the person is highlighted whilst the rest of the background becomes black.

The commented MatLab code used to achieve segmentation is shown in Appendix A.

4.4 Determining the Centre of Mass of a Tracked Object

Segmentation allows the foreground pixels of the tracked person to be identified. The central distribution of the foreground pixels is then used as the centre of mass. Figure 12 shows the centre of mass of the person being tracked in the room. Segmentation allowed the foreground pixels to be identified. The central distribution of the foreground pixels then forms the centre of mass.

Figure 12. True position of centre of mass of person

The centres of mass of the foreground pixels are enclosed within the green circle indicated in figure 12.

The x and y coordinates of the centre of mass are found from the method called "segmentation" in the code of Appendix A (Karna et al., 2007).

The centre of mass of the moving person is now used in an observer capacity in order to

determine the measurement vectors for the Kalman filter. The centre of mass used in an observer capacity is indicated in figure 13. The figure shows the observer or true centre of mass within the green circle. From the observer measurements of the position are taken and passed in to the measurement vector.

Figure 13. Estimated position of centre of mass of person

The red circle indicates the position estimates of the Kalman filter after measurements are received from the observer.

4.5 Input Calculations

4.5.1 The Frame Rate

The actual frame rate for the 640 x 480 megapixel movie clip was 15 frames per second. The main factor to be considered in this implementation is that it is not real time (online) tracking. A camera was used to record the person's movement through the room. The time period it took to record the movement was 32.49 seconds. In order to apply the Kalman MatLab algorithm on the movie clip, the clip had to be split into distinct, consecutive image frames. 121 different snapshots were then taken to represent 121 different frames of the person's movement. 121 is an arbitrary number and only came about as 121 snapshots accurately displayed the movement of the person through the room. These picture frames were then

uploaded into MatLab. The frame rate is quite low due to the utilisation of only 121 frames for the considered time period. As the duration of the actual movie was 32.49 seconds and as only 121 image frames were uploaded, the MatLab frame rate per second was determined as:

$$\frac{frames}{second} = \frac{121}{32.49} = 3.724$$

Thus merely to illustrate the operation of Kalman Tracking, the calculated frame rate using the 121 snapshots, is quite low.

4.5.2 The Change in Time Δt

The frame rate represents the number of frames or pictures per second. Thus a picture is taken every $\frac{1}{3.724} = 0.2685$ seconds.

4.5.3 The Input Distance B. u

The assumption is made that the rate of change in the x direction is the same as the rate of change in the y direction. After segmentation, observation of the movement of the centre of mass of the moving body allows the input distance to be computed. Uninterrupted movement of the person through the room starts at frame 11 and continues until frame 23, when the camera was stopped. Thus to determine the x location of the centre of mass the MatLab spatial coordinate system is used. In the spatial/pixel coordinate system a pixel is treated as a discrete unit, uniquely identified by a single coordinate pair, (x, y) . By considering the x value for the centre of mass MatLab coordinate pair, only movement in the x direction is considered and allows the computation of the rate of change
of movement per frame as:

$$v_x(t) = \frac{(Frame11 \;\; Xposition) - (Frame12 \;\; Xposition)}{\Delta t}$$

$$v_x(t) = \frac{374.4236 - 366.4458}{\Delta t 0.2685} = 29.795$$

In this implementation, the computation of rate of change in movement per frame assumes that the person's velocity or speed is constant. Thus the input distance B. u is computed as

follows:

$B.u = |\Delta t|v(t)$

$B.u = 0.2685 \times 29.795 = 7.9778$

4.5.4 Prediction in World View

The system model describes the x and y position of the person being tracked, as well as the velocity in a 4-D state vector. The 4 – D state vector is as follows:

$$x = \begin{bmatrix} x \\ z \\ x^{approx} \\ z^{approx} \end{bmatrix} \quad (27)$$

where,

- z^{approx} represents the initial approximation to the measurement vector.
- x^{approx} represents the initial approximation to the state vector.

The uncertainty in the state vector is represented by the input covariance matrix Q as:

- $$Q = \begin{bmatrix} \sigma_x & 0 & 0 & 0 \\ 0 & \sigma_y & 0 & 0 \\ 0 & 0 & \sigma_{x'} & 0 \\ 0 & 0 & 0 & \sigma_{y'} \end{bmatrix} \quad (28)$$

where,

- σ represents the square root of the variance or standard deviation of each element in the state vector

For each processed frame from the camera, changes in the tracking system state x develop in the following manner for each time index Δt:

$$x(t + \Delta t) = A(\Delta t)x(t) + Bu(\Delta t) + |\Delta t|v(t) \quad (29)$$

where,

- $|\Delta t|$ represents the modulus of the change in time. The modulus is the positive value for the change in time Δt .

The change in state x_{k+1} is the same as the state space representation of equation 5, with the only difference being that the expression for the input noise w is missing. This will be included in the calculations by the Input Covariance Matrix Q. The expression for the change in state x_{k+1} is:

$$x_{k+1} = A(\Delta t)x_k + Bu_k\Delta t \tag{30}$$

where

- $Bu(\Delta t)$ is the input distance
- $A = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

  A is the matrix relating the state in the previous step to the future step. For the tracking system A is characterized by the given matrix elements. Placing A into equation 29 gives the expression for each element of the state x_{k+1} as:

$$\begin{bmatrix} x_{k+1} \\ z_{k+1} \\ x_{k+1}^{approx} \\ z_{k+1}^{approx} \end{bmatrix} = \begin{bmatrix} x_k + \Delta t(x_k^{approx}) \\ z_k + \Delta t(z_k^{initialapprox}) \\ x_k^{approx} \\ z_k^{approx} \end{bmatrix} + Bu_k\Delta t \tag{31}$$

where

- Δt is the time period that elapses between frames.

The observation vector $z(t)$ is still defined as:

$$z(t) = H\big(x(t)\big) + v(t) \tag{32}$$

The change in time allow for determining the expression for the Error Covariance Matrices $\mathrm{P}^{\mathrm{aposteriori}}$ for each change in time $\mathrm{t}-\Delta\mathrm{t}$. This is possible having obtained an initial estimate of the state vector at time $\mathrm{t}-\Delta\mathrm{t}$ in the Kalman filter:

- $\mathrm{x}^{\mathrm{approx}}(\mathrm{t}-\Delta\mathrm{t}|\mathrm{t}-\Delta\mathrm{t})$

where,

- $x^{approx}(t-\Delta t|t-\Delta t)$ implies the estimate of x^{approx} at time $t-\Delta t$ knowing only the state information $t-\Delta t$. The state estimate $t-\Delta t$ is the *a priori* state estimate $x_{t-\Delta t}^{a\ priori}$

The uncertainty $\mathrm{P}(\mathrm{t}-\Delta\mathrm{t}|\mathrm{t}-\Delta\mathrm{t})$ of the prediction for the state estimate $\mathrm{x}^{\mathrm{approx}}$ at that same time of $\mathrm{t}-\Delta\mathrm{t}$ is given as:

$$\mathrm{P}(\mathrm{t}-\Delta\mathrm{t}|\mathrm{t}-\Delta\mathrm{t})$$

The predicted state or a posteriori state $\mathrm{x}_{\mathrm{t}}^{\mathrm{aposterior}}$ after initial conditions at time t for the new image frame is given as:

$$\mathrm{x}_{\mathrm{t}}^{\mathrm{aposterior}}(\mathrm{t}|\mathrm{t}-\Delta\mathrm{t})=\mathrm{A}(\Delta\mathrm{t})\mathrm{x}^{\mathrm{approx}}(\mathrm{t}-\Delta\mathrm{t}|\mathrm{t}-\Delta\mathrm{t})+\mathrm{B}(\Delta\mathrm{t})\mathrm{u}(\mathrm{t}-\Delta\mathrm{t}) \quad (33)$$

where,

- $\hat{x}_t(t|t-\Delta t)$ represents the state $\hat{x}_t$ at time index t due to calculations which happened at time index $t-\Delta t$.

The uncertainty P_{t} for the predicted state at time index t for a new image frame is given as:

$$\mathrm{P}_{\mathrm{t}}(\mathrm{t}|\mathrm{t}-\Delta\mathrm{t})=\mathrm{A}(\Delta\mathrm{t})\mathrm{P}(\mathrm{t}-\Delta\mathrm{t}|\mathrm{t}-\Delta\mathrm{T})\mathrm{A}^{\mathrm{T}}(\Delta\mathrm{t})+\mathrm{Q}(\Delta\mathrm{t}) \quad (34)$$

where,

- $P_t(t|t-\Delta t)$ represents the state $\hat{x}_t$ at time index t due to calculations which happened at time index $t-\Delta t$.

The MatLab code incorporating the above theory can be found in Appendix A.

5. Tracking Simulations Results and Discussions

The tracking simulation occurred in the room of figure 12. One person was monitored and tracked by the camera as he proceeded through the room.

The aim of the simulation is to compare the real movement of the person's centre of mass on the x coordinate axis to the Kalman estimate of the centre of mass' x coordinate for frames 10 to 20.

Different values were considered for the input covariance matrix Q . These values are namely; Q = 0; 1; 10; 50 and 100. The aim of the different values for the Input Covariance Matrix Q is to determine the value that allows for optimal performance of the Kalman Tracker. This will enable the determination of tracking performance with and without the
Kalman Tracker.

A random measurement noise covariance matrix R was created using a MatLab random number generator function.

The input parameters to the Kalman filter were determined as follows:

The real state estimate x_{k+1} obtained for the step index $k+1$ is:

- $x_{k-1} = \begin{bmatrix} x \\ z \\ x^{approx} \\ z^{initial} \end{bmatrix} = \begin{bmatrix} 367 \\ 274 \\ 0 \\ 0 \end{bmatrix}$

where;

- $x = 367$ is the x axis spatial coordinate when the person enters the room
- $z = 274$ is the Kalman measurement when the person enters the room

The change in time Δt or the step index k, as determined in section 4.5.2 is:

- $\Delta t = 0.2685$

The Change in the state matrix A due to the effect of changes in time Δt is:

- $A = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0.2685 & 0 \\ 0 & 1 & 0 & 0.2685 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

The input distance $B.u = |\Delta t|v(t)$ is:

- $B.u = \begin{bmatrix} 0 \\ 0 \\ 0 \\ |\Delta t|v(t) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 8 \end{bmatrix}$

The input covariance matrix Q is:

- $Q = \begin{bmatrix} \sigma_x & 0 & 0 & 0 \\ 0 & \sigma_y & 0 & 0 \\ 0 & 0 & \sigma_{x'} & 0 \\ 0 & 0 & 0 & \sigma_{y'} \end{bmatrix}$

The zero off diagonal elements show that there is no correlation between the various elements of the state covariance matrix. Thus the noise added to each element of the state vector is independent of the noise added to the other elements of the state vector.

The measurement covariance matrix R is:

- $R = \begin{bmatrix} 0.2877 & 1.1909 \\ 1.1909 & 1.1465 \end{bmatrix}$

MatLab was used to generate random white Gaussian noise to represent uncertainties and noise in the model measurement data. This allowed the above representation of the measurement covariance matrix R . This value for R is used for all simulation purposes in this implementation, as white Gaussian noise with zero means is required for proper operation of the Kalman filter.

The measurement covariance matrix R determines the amount of information from measurements to be used.

The measurement transformation matrix H used is:

- $H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$

5.1 Tracking Results for various values of Q

5.1.1 For Input Covariance Matrix Value Q = 0

For input covariance matrix Q = 0:

- $Q = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$

This input covariance matrix indicates that there is no input noise as the whole matrix is zero. The measurement covariance matrix R determined with MatLab random noise generator is:

- $R = \begin{bmatrix} 0.2877 & 1.1909 \\ 1.1909 & 1.1465 \end{bmatrix}$

5.1.1.1 Effects of R and Q on Estimate Data

The *a priori* state estimate $\hat{x}_k^-$ obtained for the step index $k = 1$ is:

$$\hat{x}_k^- = 1.0 * 10^3 * \begin{bmatrix} 0.2526 \\ 2.3689 \\ 0.0121 \\ 0.8200 \end{bmatrix}$$

This estimate vector shows an small estimate for its second element, z of approximately 2369. This is too extreme a value for the measurement value z . This would place all estimates of the state out of the expected range of 0 – 500 (as indicated in figure 10). This already indicates that the estimates by the Kalman filter are inaccurate for this value of the Input Covariance Matrix Q .

The Kalman gain obtained from the simulation code is as follows:

$$K = \begin{bmatrix} 0.0365 & 0 \\ 0 & 0.0365 \\ 0.0019 & 0 \\ 0 & 0.0019 \end{bmatrix}$$

The Kalman gain determines how much of the difference between the actual measurement $_k$ and the model measurement is used to correct the state estimate $\hat{x}_k^-$. The Kalman gain is small possibly due to the large amounts of measurement noise in the model. This reflects that the certainty of the measurement is small compared to the certainty of the current state model. Thus few measurement data is included and minimal adjustment to the estimate state $\hat{x}_k^-$ occurs. As the a priori state estimate $\hat{x}_k^-$ is already inaccurate and as the Kalman gain is too small, no significant changes occur to the state estimate $x_k^{a\ priori}$.

Figure 14 show that for values of the input covariance matrix Q which equal zero (Q = 0), tracking by this filter won't be very accurate. Figure 14 indicates the person's movement in the x direction for various frames compared to the position estimates by the Kalman filter for those same frames.

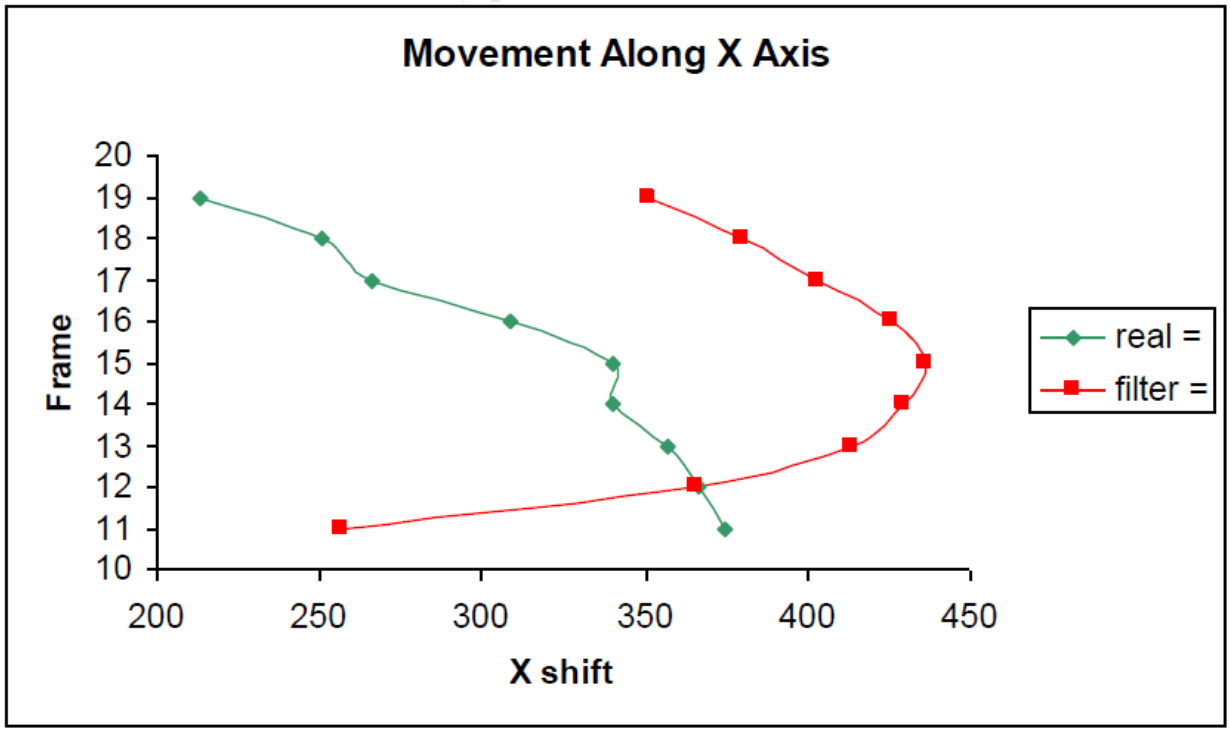


Figure 14. Movement in the X direction from frames 11 to 19, for Q = 0.

Figure 14 is a plot of the movement in the X direction from frame 11 to frame 20. The plot shows that with the input parameters provided it is impossible for estimated state of the filter to reach the true state.

5.1.2 For Input Covariance Matrix value Q = 1

For input covariance matrix Q = 1:

- $Q = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

This input covariance matrix has been selected to indicate a variance or input noise to the state vector elements of magnitude 1. This is shown by the diagonal elements of the covariance matrix which correspond to the elements of the state vector x .

The measurement covariance matrix R determined with MatLab random noise generator is:

- $R = \begin{bmatrix} 0.2877 & 1.1909 \\ 1.1909 & 1.1465 \end{bmatrix}$

5.1.2.1 Interpretation of Results

The *a priori* state estimate $\hat{x}_k^-$ obtained for the step index $k = 1$ is:

- $\hat{x}_k^- = \begin{bmatrix} 599.8066 \\ 148.3184 \\ 228.1603 \\ 88.5542 \end{bmatrix}$

This *a priori* state vector $\hat{x}_k^-$ shows reasonable values for the x coordinate state (first element of the *a priori* state vector $\hat{x}_k^-$). The real values are close to the expected range of 0 – 500 (which are the required values specified by figure 8).

The Kalman gain obtained from the simulation code is as follows:

- $K = \begin{bmatrix} 1.4773 & -0.6126 \\ -0.6126 & 1.0355 \\ 1.4039 & -0.6727 \\ -0.6727 & 0.9188 \end{bmatrix}$

For the value $Q = 1$, the Kalman gain is again small, and reflects that the certainty of the measurement is small compared to the certainty of the current state model. Again little measurement data is included and minimal adjustment to the estimate state $\hat{x}_k^-$ occurs. As the *a priori* state estimate $\hat{x}_k^-$ is already inaccurate and as the Kalman gain is too small, no significant changes occur to the state estimate $\hat{x}_k^-$.

The state estimate $\hat{x}_k^-$ is plotted against the real movement of the person through the room in figure 15.

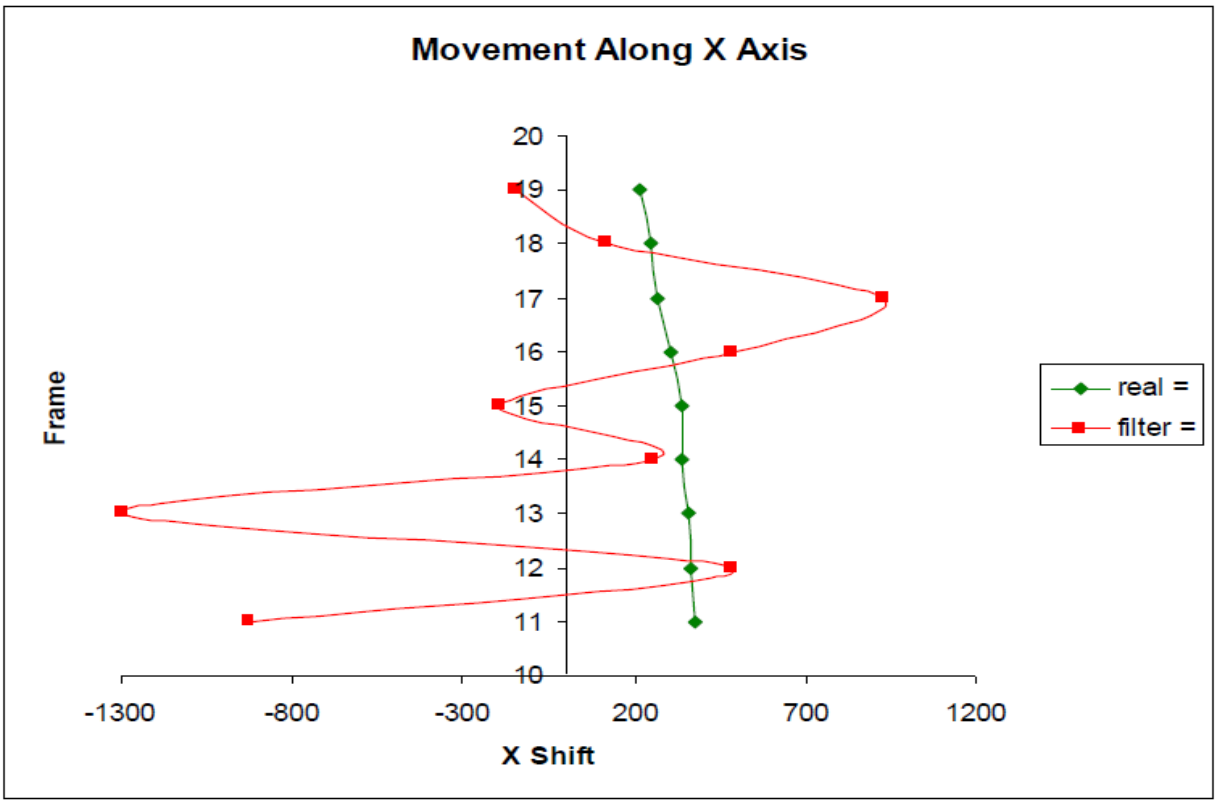


Figure 15. Movement in the X direction from frames 11 to 19 for Q = 1.

The plot shows that with the input parameters provided it is impossible for estimated state of the filter to reach the real state. The presence of the large amount of measurement noise R and

the small input covariance matrix Q has resulted in the Kalman filter following measurements less closely as it remains certain of its estimate, of state positions.

5.1.3 For Input Covariance Matrix value Q = 10

For input covariance matrix $\mathrm{Q} = 10$:

- $Q = \begin{bmatrix} 10 & 0 & 0 & 0 \\ 0 & 10 & 0 & 0 \\ 0 & 0 & 10 & 0 \\ 0 & 0 & 0 & 10 \end{bmatrix}$

This input covariance matrix has been selected to indicate a variance or input noise to the state vector elements of magnitude 10. This is shown by the diagonal elements of the covariance matrix which correspond to the elements of the state vector x .

The measurement covariance matrix R determined with MatLab random noise generator is:

- $R = \begin{bmatrix} 0.2877 & 1.1909 \\ 1.1909 & 1.1465 \end{bmatrix}$

5.1.3.1 Interpretation of Results

The *a priori* state estimate $\hat{x}_k^-$ obtained for the step $k-1$ is:

- $\hat{x}_k^- = \begin{bmatrix} 371.2280 \\ 278.5303 \\ 33.5477 \\ 202.2853 \end{bmatrix}$

The a priori state vector $\hat{\mathrm{x}}_\mathrm{k}^-$ shows reasonable values for x coordinate state (first element of the a priori state vector $\hat{\mathrm{x}}_\mathrm{k}^-$). The estimated values are close to the real expected values ranging between 0 and 500 (expected values are indicated in figure 8).

The *a priori* error covariance P_k^- matrix obtained is shown on the next page. The matrix is an indication of the amount of error present in state estimation. It represents the error between the real state vector and the estimated a priori state vector $\hat{\mathrm{x}}_\mathrm{k}^-$.

- $P_k^- = \begin{bmatrix} 13.3580 & 1.6698 & 11.6210 & 1.1871 \\ 1.6698 & 14.5621 & 1.1871 & 12.4771 \\ 11.6210 & 1.1871 & 52.7167 & 0.9202 \\ 11.1871 & 12.4771 & 0.9202 & 53.3803 \end{bmatrix}$

The Kalman gain obtained from the simulation code is as follows:

- $K = \begin{bmatrix} 0.9946 & -0.0748 \\ -0.0748 & 0.9406 \\ 0.8690 & -0.0827 \\ -0.0827 & 0.8093 \end{bmatrix}$

The Kalman gain K is proportional to the error covariance matrix P_k^- and inversely proportional to the measurement error covariance matrix R. In this case the a priori error covariance matrix P_k^- is large compared to R, the measurement covariance matrix. Thus the gain K tends towards the error covariance matrix P_k^-. This reflects that the certainty of the measurement is large compared to the certainty of the current state model and hence, significant adjustment to the a priori state estimate $\hat{x}_k^-$ occurs for each step index k.

Ideally the error covariance matrix P_k^- reduces to zero to indicate the accuracy of the model. The value obtained for the a posteriori error covariance matrix P_k is closer to zero than for the a priori error covariance matrix P_k^- due to a slightly bigger gain, which effects the determination of the a posteriori error covariance matrix as shown by equation 20. The *a posteriori* error covariance matrix P_k obtained is:

- $P_k = \begin{bmatrix} 0.1970 & 1.0987 & 0.1515 & 0.9401 \\ 1.0987 & 0.9893 & 0.9401 & 0.8294 \\ 0.1515 & 0.9401 & 42.7167 & 0.9202 \\ 0.9401 & 0.8294 & 0.9202 & 43.3803 \end{bmatrix}$

The result of the new gain on the filter state estimate x is shown in figure16.

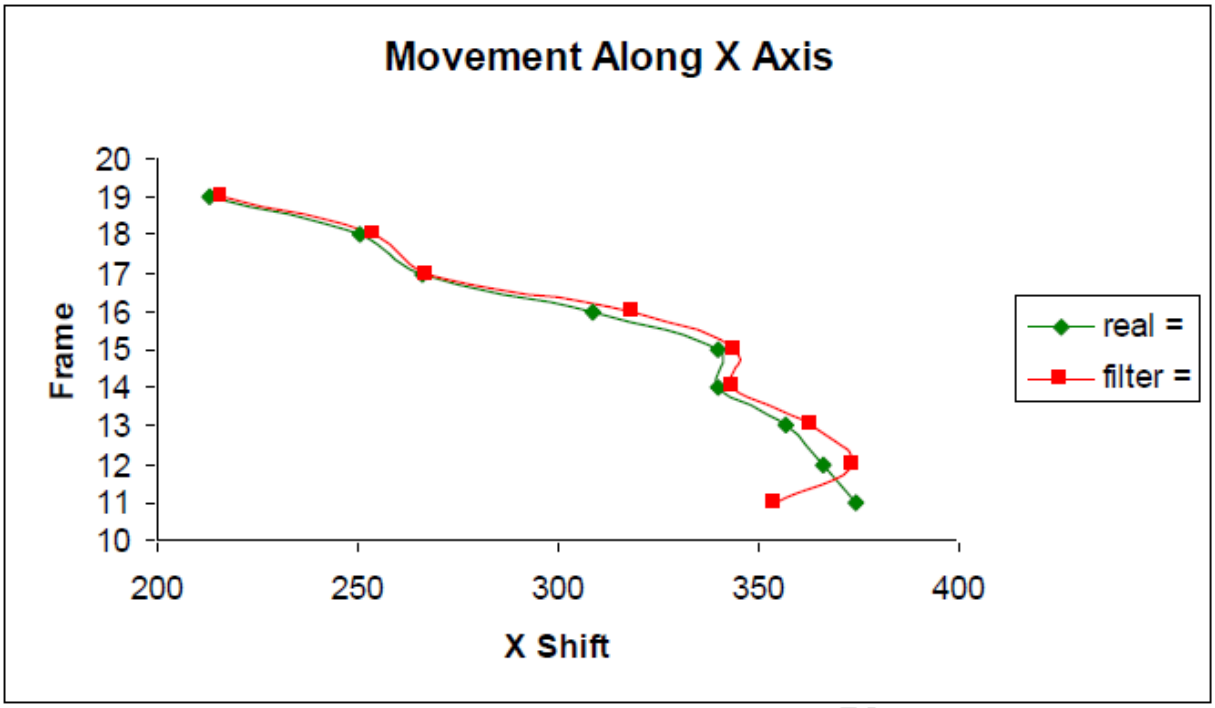


Figure 16. Movement in the X direction from frames 11 to 19, for Q = 10.

Figure 16 shows that with the input parameter Q = 10 , it is possible for the estimated state of the Kalman filter to reach the values of the real state. The Kalman filter follows measurements more closely allowing it to update each state estimate more accurately. Figure 16 also shows a slightly poor performance by the Kalman filter when there is a sudden change in direction, as occur at Frame 11 and 12.

5.1.4 For Input Covariance Matrix value Q = 50

For input covariance matrix Q = 50:

- $Q = \begin{bmatrix} 50 & 0 & 0 & 0 \\ 0 & 50 & 0 & 0 \\ 0 & 0 & 50 & 0 \\ 0 & 0 & 0 & 50 \end{bmatrix}$

This input covariance matrix has been selected to indicate a variance or input noise to the state vector elements of magnitude 50. This is shown by the diagonal elements of the covariance matrix which correspond to the elements of the state vector x .

The measurement covariance matrix R determined with MatLab random noise generator is:

- $R = \begin{bmatrix} 0.2877 & 1.1909 \\ 1.1909 & 1.1465 \end{bmatrix}$

5.1.4.1 Interpretation of Results

The *a priori* state estimate $\hat{x}_k^-$ obtained for the step index $k-1$ is:

- $\hat{x}_k^- = \begin{bmatrix} 386.0588 \\ 275.1856 \\ 31.0279 \\ 199.5950 \end{bmatrix}$

The values of the *a priori* state estimate $\hat{x}_k^-$ are very close to those for the position of the real person at that time instant namely;

- $x_{k-1} = \begin{bmatrix} x \\ z \\ x^{approx} \\ z^{approx} \end{bmatrix} = \begin{bmatrix} 367 \\ 274 \\ 0 \\ 0 \end{bmatrix}$

The accuracy of the Kalman filter is good at this chosen value for the input covariance matrix Q, as it follows the measurement data more closely due to large amounts of uncertainty present in the input. This is proven by the a priori error covariance matrix P_k^- which indicates large amounts of noise present in the state estimation. The a priori error covariance matrix P_k^- represents the error between the real state vector and the estimated a priori state vector $\hat{x}_k^-$.

The *a priori* error covariance matrix for this value of P_k^- is:

- $P_k^- = \begin{bmatrix} 65.7521 & 1.7842 & 57.4486 & 1.2843 \\ 1.7842 & 67.0388 & 1.2843 & 58.3748 \\ 57.4486 & 1.2843 & 263.1132 & 1.0029 \\ 1.2843 & 58.3748 & 1.0029 & 263.8365 \end{bmatrix}$

Kalman filter incorporates more measurements as it is very uncertain about its position as indicated by the large Input Covariance Noise Matrix Q. The Kalman filter follows the error covariance matrix P_k^-, and this reflects that the certainty of the measurement is large compared to the certainty of the current state model. Thus, significant adjustment to the a priori state estimate $\hat{x}_k^-$ occurs, for each time index k.

The plot of the Kalman estimate compared to the real location of the person is shown in figure 17. The Plot shows that the Kalman filter is more accurate when it reinforces its estimate with more measurement data. This value $Q = 50$ also allows the Kalman filter to be more accurate and responsive to sudden changes in direction as indicated on the plot at frame 11 and 12.

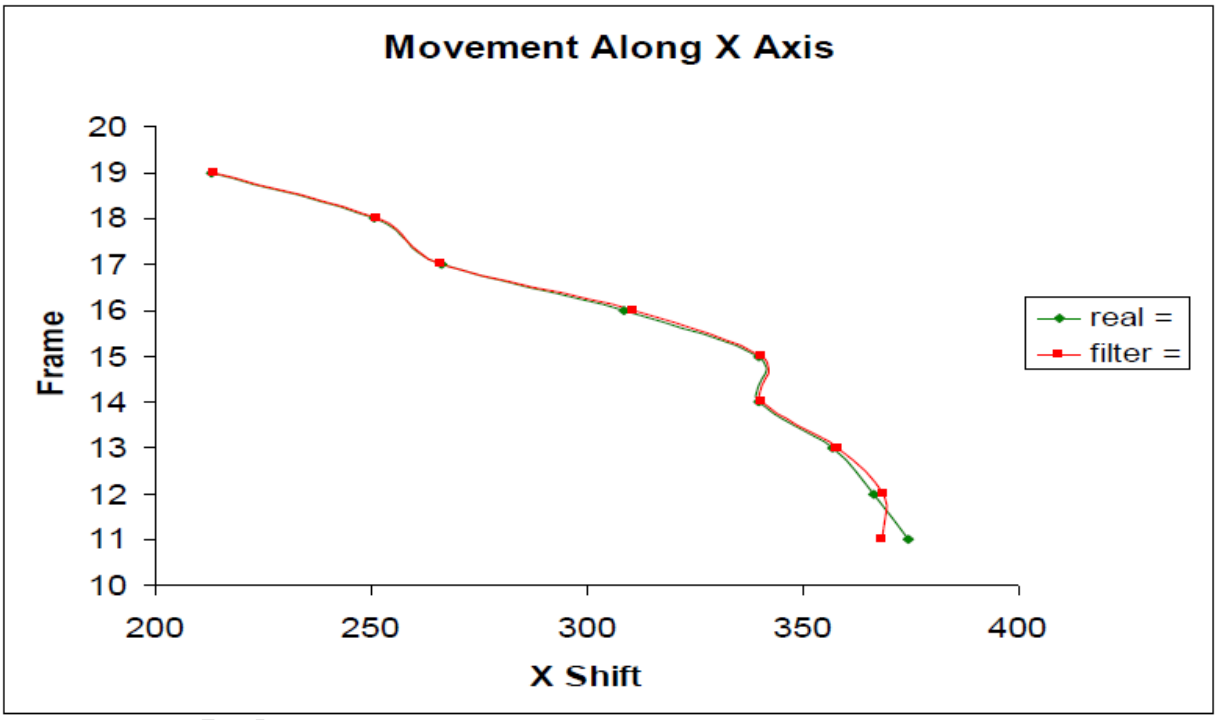


Figure 17. Movement in the X direction from frames 11 to 19, for Q = 50.

Figure 17 shows that the Kalman tracking performance is adequate but takes a bit long to recover to the correct position estimates once a sudden change in direction occurs as occurred at frame 11. At frame 12 the position estimate was still off by a small margin

5.1.5 For Input Covariance Matrix value Q = 100

For input covariance matrix $Q = 100$:

- $Q = \begin{bmatrix} 100 & 0 & 0 & 0 \\ 0 & 100 & 0 & 0 \\ 0 & 0 & 100 & 0 \\ 0 & 0 & 0 & 100 \end{bmatrix}$

This input covariance matrix has been selected to indicate a variance or input noise to the state vector elements of magnitude 100. This is shown by the diagonal elements of the covariance matrix which correspond to the elements of the state vector x.

The measurement covariance matrix R determined with MatLab random noise generator is:

- $R = \begin{bmatrix} 0.2877 & 1.1909 \\ 1.1909 & 1.1465 \end{bmatrix}$

5.1.5.1 Interpretation of Results

The *a priori* state estimate $\hat{x}_k^-$ obtained for the step index $k-1$ is:

- $\hat{x}_k^- = \begin{bmatrix} 367.6224 \\ 274.6981 \\ 30.6789 \\ 199.1998 \end{bmatrix}$

The values of the *a priori* state estimate $\hat{x}_k^-$ are very close to those for the position of the real person at that time instant namely;

- $x_{k-1} = \begin{bmatrix} x \\ z \\ x^{approx} \\ z^{approx} \end{bmatrix} = \begin{bmatrix} 367 \\ 274 \\ 0 \\ 0 \end{bmatrix}$

From all the values of Q used, this provides the closest estimate of the location of the person being tracked.

The Kalman filter incorporates more measurements as it is very uncertain about its position due to the large amount of input noise provided by the input covariance matrix Q . The average values of the a priori error covariance matrix P_k^- are also larger than those of the noise

covariance matrix R , causing the Kalman filter to follow measurements more closely due to the uncertainty of its position in state estimation. The results of the Kalman filter tracking the x coordinate of the state vector x are shown in figure 18.

The Kalman gain K is proportional to the error covariance matrix P_k^- and inversely proportional to the measurement error covariance matrix R. in this case the a priori error covariance matrix P_k^- is large compared to R, the measurement covariance matrix. The gain tends towards the a priori error covariance matrix P_k^-, and is thus a reflection that the certainty of the measurement is greater than the certainty of the current state model. Thus significant adjustment to the a priori state estimate $\hat{x}_k^-$ occurs, for each step index k.

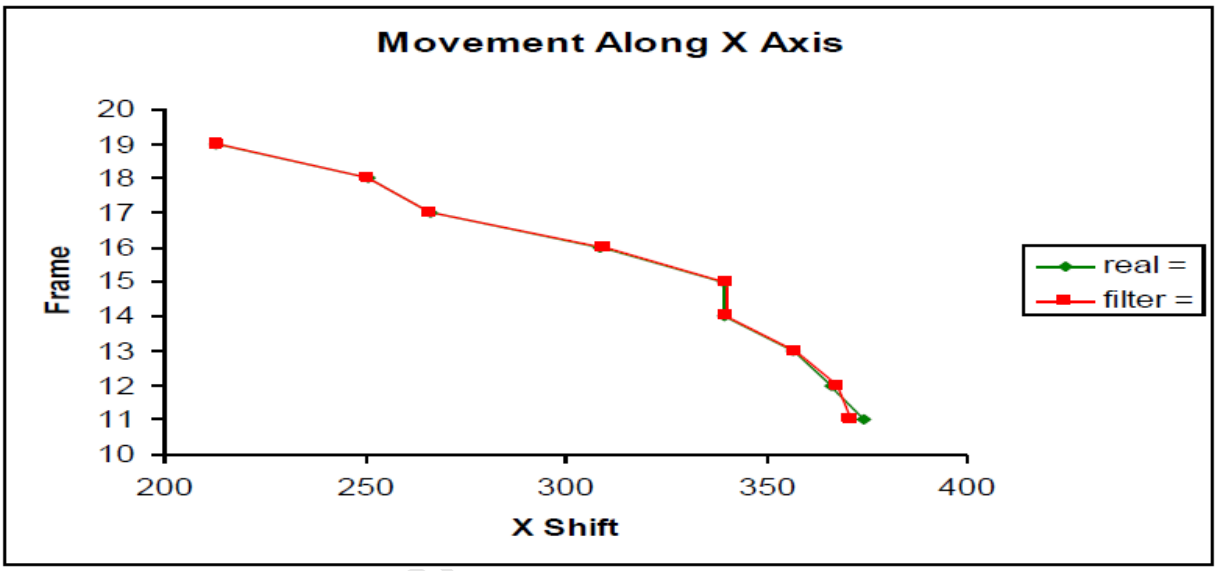


Figure 18. Movement in the X direction from frames 11 to 19, for Q = 100.

Figure 18 is a plot of the movement in the X direction from frame 11 to frame 19. The plot shows that with the input parameters it is possible for the estimated state of the Kalman filter to reach the values of the real state. The Kalman filter follows measurements more closely allowing it to update each state estimate. Figure 18 also shows a better performance by the Kalman filter for Q = 100 then for Q =50 when there is a sudden change in direction, as occurs at Frame 11 and 12.

The values obtained for the estimated state for Q =100 do not vary much from the values obtained for the estimated state for Q =50. The only difference is the slight improvement where a sudden change in direction occurs as at Frame 11.

5.2 Sources of Error in Measurements

- Model linearisation errors
- Lighting conditions. Different lighting conditions have an impact on the accuracy of the vision information obtained. The accuracy is affected by shadows and reflections from windows and mirrors, which can cause a difference in foreground pixels obtained during segmentation.
- The filter overshoots in cases where the object cannot be seen. An example of this occurrence is when the tracked object moves behind a wall. In this implementation it occurs for frames 11 and 12.

6. Conclusion

In conclusion the three questions which directed the research will be answered. These three major questions were: First, how accurate are rigid 2D object tracking methods when used for articulated (composed of rigid parts attached by links or joints) object tracking? Second, how does the tracking algorithm perform when treating occlusions (tracked object's visibility to the camera is blocked by another object) as noise? Third, Is it possible to manipulate the Kalman input parameters in order to achieve more accurate tracking?

6.1 How accurate are rigid 2D object tracking methods when used for articulated (composed of rigid parts attached by links or joints) object tracking?

The MatLab code allowed the implementation of a Kalman filter which tracks the x and y spatial coordinates of an object's centre of mass.

The x and y coordinate plot introduced in figure 8 is shown in figure 19 to demonstrate the x and y spatial coordinate shift of the person moving through the room.

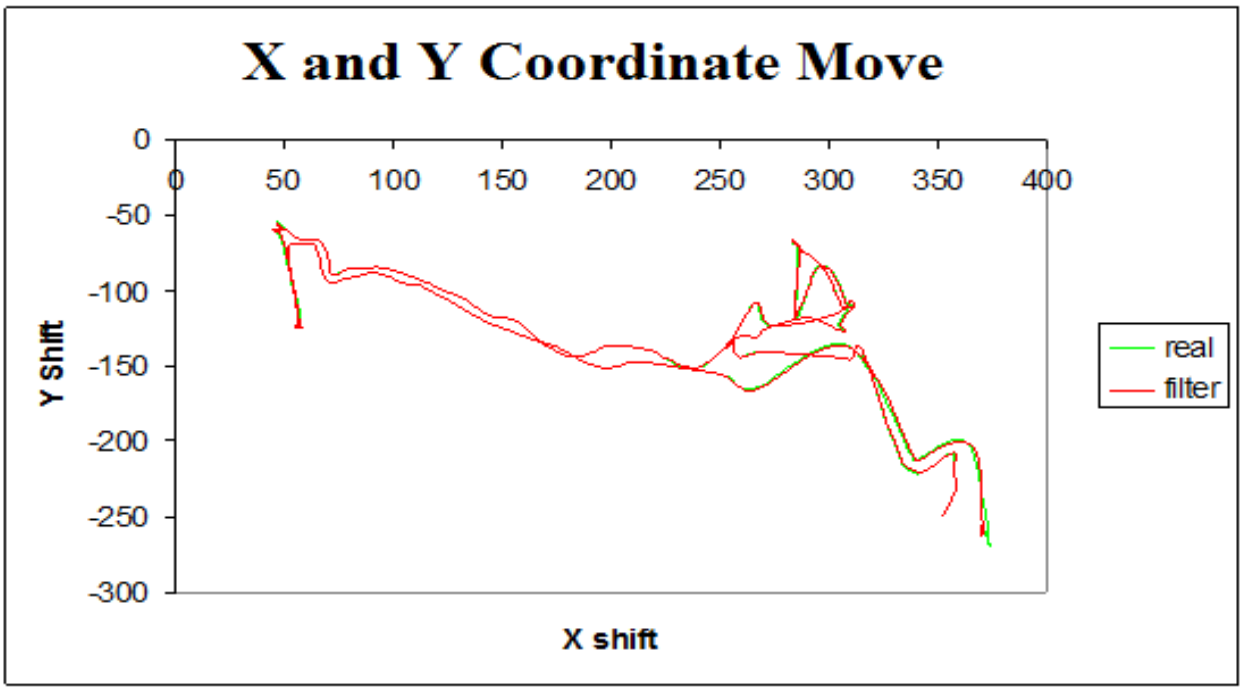


Figure 19. X and Y coordinate plot for movement through the room using Q = 100 .

In figure 19 the green line represents the true spatial coordinate location and movement of the

tracked person's centre of mass. The red line in figure 19 represents the spatial coordinate location of the tracked person's centre of mass as estimated by the Kalman filter. Figure 19 shows that the Kalman filter is very accurate in estimating the position of the centre of the tracked person. Thus following the x and y spatial coordinates of the tracked object give an accurate indication of the object in a real three dimensional environment.

The best position estimates were obtained for Input Noise Covariance Q = 100. For Q = 100, an estimate for the x spatial coordinate at frame 11 shown in figure 20 was obtained as x = 367.6224. The actual value for the x coordinate at frame 11 was x =367. This represents an accuracy of 99.83 % for the position estimate.

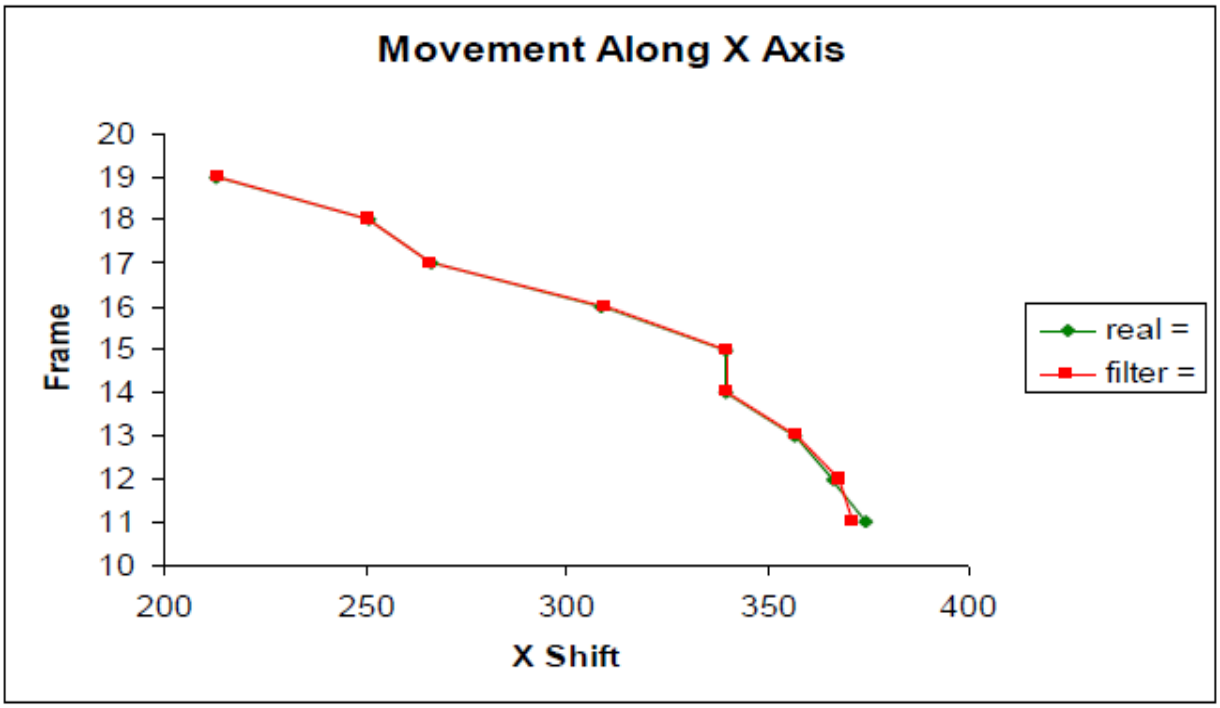


Figure 20. X coordinate for real movement vs. Kalman estimates

Using the same value for the Input Noise Covariance Q = 100, produces an estimate of the z measurement vector as $z = 274.6981$ at frame 11 shown in figure 20. The actual value of the z measurement vector at frame 11 was $z = 274$. This represents an accuracy of 99.75% for the measurement estimate.

6.2 How does the tracking algorithm perform when treating occlusions (tracked object's visibility to the camera is obstructed by another object) as noise??

In figure 21 an occlusion occurs at frame 93 when the tracked person moves behind a wall for about 0.5370 seconds. Figure 21 provides an indication of the Kaman tracking and prediction ability in the presence of total occlusion (visibility is totally obstructed).

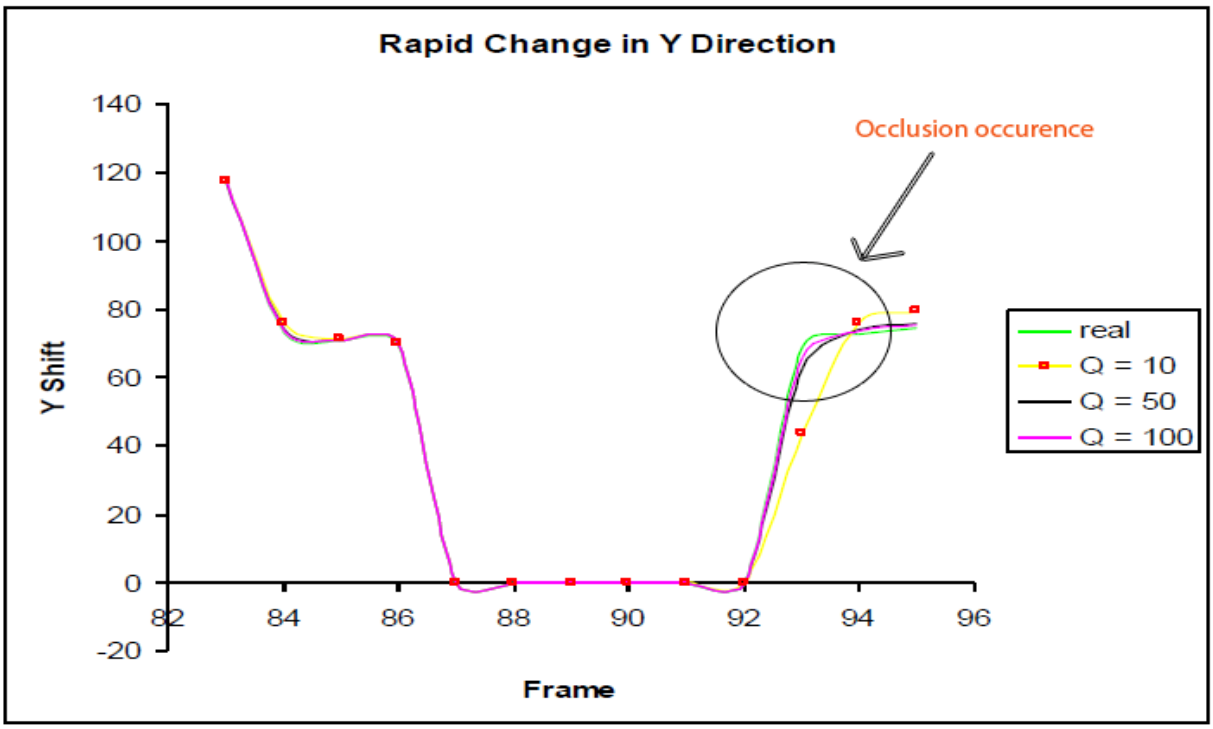


Figure 21. Occlusion handling of the tracking algorithm

The tracking algorithm achieves superior performance for Input Noise Covariance Matrx values $Q = 50$ and for $Q = 100$. These position estimates indicated by the purple and black lines are almost similar to the actual person's position. Poorer performance is obtained when using a smaller value for the Input Noise Covariance Matrix as indicated by the performance of $Q = 10$ in Figure 21. The results show that the Kalman filter performs satisfactorily when treating occlusions as noise for short intervals.

6.3 Is it possible to manipulate the Kalman input parameters in order to achieve more accurate tracking?

The Kalman filter was implemented to achieve tracking for various values of the input covariance matrix Q . The values of Q utilised were: Q = 0 , Q = 1 , Q = 10 , Q = 50 and Q = 100. The experiments show that the three values of the input covariance matrix Q , namely Q = 10, Q = 50 and Q = 100 are considered for best performance of the Kalman filter for the purpose of tracking. The tracking results have shown that the values of Q = 50 and Q = 100 provide the best estimation of the state $\hat{x}_k$, with Q = 100 providing the best estimation. Results show that the tracking ability and state estimation capability of the Kalman filter is directly proportional to Q. All three values of Q = 10; 50 and 100, provide sufficient tracking ability and state estimation in the x direction. Further simulations allowed a plot of these values for the y coordinate axis. The plot of the y axis is shown in figure 22. This plot proves the superior performance for Q = 50 and Q = 100.

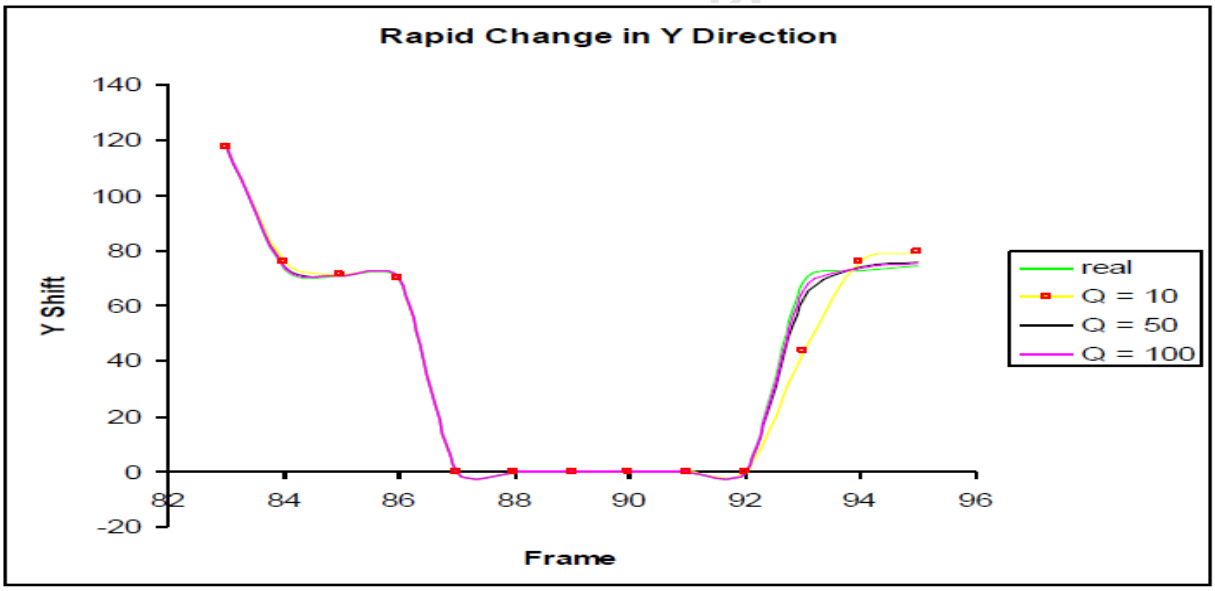


Figure 22. Change in the y coordinate axis for movement through the room

The performance in the y axis does not vary much from performance in the x axis. Best performance occurs firstly for Q =100 and followed closely by Q = 50. For Q = 10 an adequate Kalman response is also obtained.

6.4 Implications of the Findings

The algorithm used in this dissertation provides a basic understanding of the use of the Kalman filter in state estimation. A state space tracking mechanism making use of a Kalman filter is presented.

Tracking results indicate that tracking algorithms designed for rigid object tracking, can be used to track the centre of mass of an articulated object. The results achieved show that with an accurate state model and the formulation of prior assumptions, motion of the tracked object can be estimated, thus enabling the Kalman filter to be used as an efficient tracking mechanism. Off-line applications such as animating autonomous human characters in animated movies and video games can benefit from offline Kalman tracking. The data obtained from tracking a person or character off-line can be used to accurately drive a character in a video game or animated movie. Another advantage off-accurate offline tracking, is that human operators can search archived video for specific patterns without actually viewing the video. This would be possible by performing off-line motion tracking on the archived video in order to obtain tracked motion patterns which coincide with specific motion patterns stored in a database.

The findings also show that for the tracking purpose, best results were obtained for Q = 100. However, the tracking performance does not differ much from that for Q = 50. The simulations also proved that performance increases for larger values of Q, as more measurement data is incorporated into the Kalman position estimates. The findings show that with increasing values of Q, more measurement data is incorporated into Kalman position estimates. Thus, with the availability of more measurement data, the Kalman filter can perform more accurate position estimates. The tracking experimentation show that Q = 50 can be used as an input parameter to this Kalman filter as performance does not vary significantly for large increases of Q beyond this value.

Finally tracking results show that the Kalman tracking algorithm can be used to provide position estimates where the tracked object is partially or totally occluded. Thus position estimates can be obtained when the data input is uncertain for a short period of time.

6.5 Limitations of the Study and Suggestions for Further Research

Due to the scope of this research there are several limitations that should be considered for

future research.

The following issues were not entirely covered in my research:

- The tracking algorithm was implemented offline on a video recording. Online tracking was not performed.
- Encountered difficulties processing the entire movie clip with the Kalman tracking algorithm. The movie had to be split into consecutive image frames. The image frames were then uploaded to MatLab.
- Only white Gaussian noise was considered and thus tracking algorithms were restricted to the Kalman filter. Algorithms, such as the Particle filter, were not implemented to research cases when the input or system noise is not Gaussian.
- Encountered difficulties accounting for latency and person's movement after a tracker cycle has started. The tracker cycle involves sampling, calculating an estimate and finally producing a result. Holloway arrived at a rule of thumb of 1 ms of latency corresponds to 1 mm of misregistration in estimates (Holloway, 1995).
- Tracking algorithm utilised is restricted to tracking of one object or person. Difficulties were encountered in attempting to extend the algorithm to multiple object or person tracking.
- Tracking algorithm utilised requires a constant background. If background changes problems are incurred in performing segmentation.
- Tracking simulations were only carried out using a single camera. Multi camera tracking systems are in available which would enable greater accuracy and the tracking of multiple entities (Aggarwal and Cai 1996; Bergen et al. 1991; Merven et al. 2003).

Appendix I

The code for detection the motion in the background is in the file called, “motion.m”. The code for separating the moving object from the background is in the file called, “segmentation.m”. The code to make use of the Kalman filter is in the file called, “kalman.m”

Code (Karna et al. 2007)

motion.m

```
%detects the motion of a body
clear,clc
% compute the background image
%Imzero creates a 3 dimensional zero matrix, similar in dimension to the
%frame or pictures being read
Imzero = zeros(288,384,3);

%iterate over the first 5 images as they contain only  the background

for i = 1:5
% create an array to read in the images
Im{i} = double(iXcoordead(['Pics/',int2str(i),'.jpg']));

%Transform the array to a mtrix by adding it to the zero matrix
Imzero = Im{i}+Imzero;
end

%The Imback is the result of the average of all 5 background
%images

Imback = Imzero/5;

%create a matrix f similar size as Imback
[XCOORD,YCOORD,Dim] = size(Imback);
```

```
% loop over all images
for i = 1 : 121
  % load image into an array
  Im = (iXcoordead(['Pics/',int2str(i), '.jpg']));

  %

  imshow(Im)

  Imwork = double(Im);
```

segmentation.m

```
% extracts the center (cc,cr) and radius of the largest blob
function [foremm,xcentre,ycentre,radius,flag]=segmentation(Imwork,Imback,index)%

  xcentre  =  0;
  ycentre  =  0;
  radius   =   0;
  flag = 0;
  [Xrange,Yrange,Dim] = size(Imback);

  % subtract background & select pixels with a big difference
  fore = zeros(XCOORD,YCOORD);          %image   subtracktion
  fore = (abs(Imwork(:,:,1)-Imback(:,:,1)) > 37) ...
     | (abs(Imwork(:,:,2) - Imback(:,:,2)) > 37) ...
    | (abs(Imwork(:,:,3) - Imback(:,:,3)) > 37);

  % Morphology Operation  erode to remove small noise. this method performs
  % erosison on a binary image using the structuring element ones(1). This
  %  removes  the  zeros  from  the  image.

  foremm = bwmorph(fore,'erode',2); %2 time



  % select largest object
  labeled = bwlabel(foremm,4);
  stats = regionprops(labeled,['basic']);%basic mohem nist
```

```
[N,W] = size(stats);
if N < 1
return
end

% do bubble sort (large to small) on regions in case there are more than
% 1 large area or bubble

id = zeros(N);
for i = 1 : N
id(i) = i;
end
for i = 1 : N-1
 for j = i+1 : N
  if stats(i).Area < stats(j).Area
   tmp = stats(i);
   stats(i) = stats(j);
   stats(j) = tmp;
   tmp = id(i);
   id(i) = id(j);
  id(j) = tmp;
  end
 end
end

% make sure that there is at least 1 big region
if stats(1).Area < 100
 return
end
selected = (labeled==id(1));

% get center of mass and radius of largest area
centroid = stats(1).Centroid;
radius = sqrt(stats(1).Area/pi);
xcentre = centroid(1);%x coordinate of centre of mass
ycentre = centroid(2);%y coordinate of centre of mass
flag = 1;
return
```

kalman.m

```
clear,clc
% compute the background image
%Imzero creates a 3 dimensional zero matrix, similar in dimension to the
%frame or pictures being read
Imzero = zeros(288,384,3);

%iterate over the first 5 images as they contain only  the background

for i = 1:5

% create an array to read in the images

Im{i} = double(iXcoordead(['Pics/',int2str(i),'.jpg']));

%Transform the array to a mtrix by adding it to the zero matrix

Imzero = Im{i}+Imzero;

end

%The Imback is the result of the average of all 5 background
%images

Imback = Imzero/5;
[Xrange,Yrange,Dim] = size(Imback);

%creating random white Gaussian noise
var1 = randn;
var2 = randn;
var3 = randn;

% Kalman filter initialization
% R = [[ abs(var1), abs(var3)]',[ abs(var3), abs(var2)]'];
R = [[0.2877, 1.1909]',[1.1909, 1.1465]']; % 2 x 2 input measurement noise covariance matrix
H=[[1,0]',[0,1]',[0,0]',[0,0]']; % 2 x 4 initial matrix to relate the state vector to the measurement vector
Q= 100*eye(4); % 4 x 4 input covariance matrix
```

```
P = 100*eye(4); % 4 x 4 initial error covariance matrix
dt= 0.2685 %
A=[[1,0,0,0]',[0,1,0,0]',[dt,0,1,0]',[0,dt,0,1]'];%State transition matrix
change = 8 %pixels change/time step
Bu = [0,0,0,change]';
kfinit=0;
x=zeros(100,4);

% loop over all images
for i = 1 : 121
 % load image
 Im = (iXcoordead(['Pics/',int2str(i), '.jpg']));
 %imshow(Im)
 imshow(Im)
 Imwork = double(Im);

 %perform segmentation and extracr the moving object
 [foremm,xcentre(i),ycentre(i),radius,flag] = segmentation(Imwork,Imback,i);
 if flag==0
  continue
 end

 hold on
  for c = -1*radius: radius/20 : 1*radius
   r = sqrt(radius^2-c^2);
   plot(xcentre(i)+c,ycentre(i)+r,'g.')
   plot(xcentre(i)+c,ycentre(i)-r,'g.')
  end

  % Kalman update

 if kfinit==0

  xapriori = [Xrange/2,Yrange/2,0,0]'

 else

  xapriori=A*x(i-1,:)' + Bu
```

```
  end

  kfinit=1;
  Papriori = A*P*A' + Q
  K = Paprioiri*H'*inv(H*Papriori*H'+R)
  x(i,:) = (xapriori + K*([xcentre(i),ycentre(i)]' - H*xaprioiri))'; P

  = (eye(4)-K*H)*Paprioiri

  hold on

    for c = -1*radius: radius/20 : 1*radius r =
      sqrt(radius^2-c^2);
      plot(x(i,1)+c,x(i,2)+r,'r.')
      plot(x(i,1)+c,x(i,2)-r,'r.')

    end pause(0.1)

end
```

References:

Aach, T., Kaup, A. And Mester, R. (1993) Statistical model-based change detection in moving video. *Signal Processing*, 31(2), pp. 165-180

Aggarwal, J.K., Cai, Q., Liao, W. and Sabata, B. (1998) Nonrigid motion analysis: Articulated and elastic motion. *Computer Visual Image Understanding*, 70(2), pp. 142-156

Aggarwal, J. And Cai, Q. (1996) Tracking human motion using multiple cameras. In *Proceeding of the 13th International Conference on Pattern Recognition (ICPR96), Vol. 3*, Vienna, August 1996., pp. 68-72

Arulampalam, S., Maskell, S., Gordon, N. and Clapp, T. (2002) A Tutorial on particle filters for on-line non-linear/non-Gaussian Bayesian tracking. *IEEE Transactions on Signal Processing*, 50(2), pp. 174-188

Asano, T., Chen, D.Z., Katoh, N., Tokuyama, T (1996). *Polynomial-time solutions to image segmentation*. In: Proceedings of the seventh annual SIAM-ACM Conference on Discrete Algorithms, Atlanta, January 1996. New York: Association for Computing Machinery, p104-11

Bergen, J.R., Burt, P.J., Hanna, K., Hingorani, R., Jeanne, P. and Peleg, S. (1991) Dynamic multiple-motion computation. In Y.A. Feldman and A. Bruckstein. (eds.) *Artificial Intelligence and Computer Vision*, Elsevier, Holland , pp. 147-156

Bertsekas, D.P. (1971) *Control of Uncertain Systems with a Set-Membership Description of the Uncertainty*. Published Thesis (PhD), Massachusetts Institute of Technology.

Burdea, G.H. and Coiffet, P. (2003) *Virtual Reality Technology, 2nd ed*. New York: Wiley-Interscience.

Chien, S.Y., Ma, S.Y. and Chen, L.G. (2002) Efficient moving object segmentation algorithm using background registration technique. *IEEE Transactions on Circuits and Systems for Video Technology*, 12(7), pp. 577-586

Correia, P.L. and Pereira, F. (2004) Classification of video segmentation application scenarios. *IEEE Transactions on Circuits and Systems for Video Technology*, 14(5), pp. 735-741

Dimitrova, N., Zang, H., Shahraray, B., Sezan, I., Huang, T. and Zakhor, A. (2002) Applications of video content analysis and retrieval. *IEEE Transactions on Multimedia*. 9, pp. 42-55

Faugeras, O., Quan, L., and Strum, P. (2000) Self-calibration of a 1D projective camera and its application to the self-calibration of a 2D projective camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10), pp. 1179-1185

Finlayson, G., Hordley, S. and Hubel, P. (2001) Color by correlation: A Simple, unifying framework for colour constancy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(11), pp. 1209-1221

Figueroa, P., Leite, N., Barros, R.M.L., Cohen, I. and Medioni, G. (2004). Tracking soccer players using the graph representation. In: Proceeding of the 17th International Conference of Pattern Recognition (ICPR2004), Vol. IV, Cambridge, August 2004, pp. 787-790.

Freeston, L. (2002) *Application of the Kalman Filter Algorithm to Robot Localisation and World Modelling*. Published Thesis (Bsc), University of Newcastle, Australia.

Fuentesa, L.M. and Velastin, S.A. (2006) People tracking in surveillance applications. *Image and Vision Computing*, 24(11), pp. 1165-1171

Gargi, U., Kasturi, R. and Strayer, S.H. (2001) Performance characterisation of video-shot change detection methods. *IEEE Transaction on Circuits and Systems for Video Technology*, 10, pp. 1-13

Gavrila, D.M. (1999) The visual analysis of human movement: A survey. *Computer Vision and Image Understanding*, 73(1), pp. 82-98.

Gentile, C., Camps, O. and Sznaier, M. (2004) Segmentation for robust tracking in the presence of severe occlusion. *IEEE Transactions on Image Processing*, 13(2), pp. 166-178

Gevers, T. and Smeulders, A.W.M. (2000) Pictoseek: Combining color and shape invariant features for image retrieval. *IEEE Transactions on Image Processing*, 9(1), pp. 102-119

Haritaoglu, R., Cutler, R., Harwoodc, D., and Davis, L.C. (2001) Backpack: Detection of people carrying objects using silhouettes. *Computer Vision and Image Understanding*, 81(3), pp. 385-397

Holloway, R. L. (1995). *Registration errors in augmented reality systems.* Ph.D. thesis, University of North Carolina at Chapel Hill.

Irani, M., Rousso, B. And Peleg, S. (1994) Computing occluding and transparent motions. *International Journal of Computer Vision*. 12(1), pp. 5-16

Isard, M. and MacCormick, J. (2001) BraMBLe: A Bayesian multiple-blob tracker. In: IEEE *Internation Conference on Computer Vision (ICCV2001*), Vol. 2. Vancouver: , pp. 34-41

Isard, M. and Blake, A. (1998) Condensation-conditional density propagation for visual tracking. *International Journal of Computer Vision*. 29(1), pp. 5-28

Izquierdo, E. and Ghanbari, M. (2002) Key components for an advanced segmentation system. *IEEE Transactions on Multimedia*, 4(1), pp. 97-113

Jang, D. and Choi, H. (2000) Active models for tracking moving objects. *Pattern Recognition*, 33, pp. 1135 – 1146

Jang, D., Jang, S. and Choi, H. (2002) 2D human body tracking with Structural Kalman filter. *Pattern Recognition*, 35, pp. 2041 - 2049

Julier, S.J., Ulmann, J.K., and Durrant-Whyte, H. (1995) A new approach for filtering nonlinear systems. In: *American Control Conference*, Orlando, April 1995. pp. 1628-1632

Kalman, R (1960) A new approach to linear filtering and prediction problems. In: *Transaction of the ASME - Journal of Basic Engineering*, 82 (Series D), March 1960. pp. 35-45

Karna A.K., Budhwar, D and Sandhu, S. (2007) *2D Target Tracking Using Kalman Filter*. Published Thesis (Bsc), National Instituite of Technology, India.

Konrad, J. (2009) Motion Detection and Estimation. In: Bovik, A.L (2009) (ed) *The Essential Guide to Video Processing*. San Diego: Elsevier Inc., pp. 31-68

Koprinska, I. And Carrato, S. (2001) Temporal video segmentation: A survey. *Signal Processing: Image Communication*, 16(5), pp. 477-500

Li, H., Sun, H. and Derin editors, H. (1997) *Video Compression for Multimedia computing-statistically Based and Biologically Inspired Techniques.* New York: Kluwer Academic Publishers

Malis, E. and Cipolla, R. (2002) Camera self-calibration from unknown planar structures enforcing the mulyiview constraints between collineations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(9), pp. 1268-1272

MATHWORKS (1998) *Matlab Image Processing Toolbox* [WWW] Mathworks Available from: http://www-visl.technion.ac.il/~re/anatlab/html/2- ImageTypes/matlabip.htm [Accessed 14/01/2010]

Maybeck, P.S. (1979) *Stochastic Models, Estimation and Control*. New York: Academic Press.

Meiss, J (2007) *Dynamical Systems* [WWW] Meiss, J. Available from: http://www.scholarpedia.org/article/Dynamical_systems [Accessed 09/01/2009]

Merven, B, Nicolls, F, and de Jager, G. Multi-camera person tracking using an extended kalman filter. In *Fifteenth Annual Symposium of the Pattern Recognition Association of South Africa,* 2003

Moeslund, T.B. and Granum, E. (2001) A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3), pp. 231-268

Moeslund T.B. (2000) Interacting with a virtual world through motion capture. In L. Qvortrup (ed) *Interaction in Virtual Inhabited 3D Worlds*, Chap. 11. Berlin/New York: Springer-Verlag,

Moral, P. (1996) Non-linear filtering: Interacting particle solution. *Markov Processes and Related Fields, 2*(4), pp. 555-580

Moravčík, T (2009) Image Segmentation in Programming Environment MATLAB. In: *XI International PhD Workshop, OWD 2009, Conference Archives PTETiS*. Wisla, Poland, 17/Oct/2009-20/Oct/2009.Warsaw: pp. 284-289. 26(ISBN:83-922242-5-6)

Oberkampf, W.L. (2005) Uncertainty Quantification Using Evidence Theory, from *Advanced Simulation & Computing Workshop Error Estimation, Uncertainty Quantification and Reliability in Numerical Simulations*. Stanford University, New Mexico on August 22.

Perez, P., Vermaak, J. And Blake, A. (2004) Data fusion for virtual tracking with particles. *Proceeding of the IEEE*, 92(3), pp. 495-513

Pinkney, M (2003) Expectation [WWW] Mathsrevision.net. Available from: http://www.mathsrevision.net/alevel/statistics/expectation.php/ [Accessed 02/11/2008]

RELIABILITY ENGINEERING RESOURCE WEB SITE (2006). *Bayesian Statistics* [WWW] Reliability Engineering Resources. Available from: http://www.weibull.com/LifeDataWeb/bayesian_statistics.htm [Accessed 30/01/2011].

Šonka, M. and Hlaváč, V. (1992) *Počítačové videní.* Praha: Grada Publishing

Sonka, M., Hlavac, V. and Boyle, R. (2008).: *Image Processing, Analysis and Machine Vision*. 3d ed. Toronto: Thomson Engineering.

STATISTICAL GLOSSARY (n.d.) *Apriori Probability* [WWW] Statistical

Glossary.Available rom: http://www2.statistics.com/resources/glossary/a/aprioriprob.php [Accessed 02/11/2008]

STATISTICAL GLOSSARY (n.d.) *Posterior Probability* [WWW] Statistical Glossary. Available from: http://www2.statistics.com/resources/glossary/p/postprob.php [Accessed 02/11/2008]

Swain, M.J. and Ballard, D.H. (1991) Color indexing. *International Journal of Computer Vision, 7*(1), pp. 11-32

Tekalp, A. (1995) *Digital Video Processing*. Upper Saddle River, NJ: Prentice Hall PTR

Tekalp, A.M. (2009) Video Segmentation. In A.L. Bovik (eds) *The Essential Guide to Video Processing*. San Diego: Elsevier Inc., pp. 141-173

Toyama, K. (1998) *Prolegomena for robust face tracking*. Tech. Rep. MSR-TR-98-65, Microsoft Research.

Welch, G. and Bishop, G. (1995) An introduction to the Kalman filter. Tech. Rep. 95-041, University of North Carolina at Chapel Hill, Department of Computer Science.

Welch, G. (2009) HISTORY: The use of the Kalman Filter for human motion tracking in virtual reality. In: Slater, M. and Weisenberger, J. (eds*.) Presence: Teleoperators and Virtual Environments.* Vol. 18, Number I. Massachusetts Institute of Technology, pp. 72 - 91.

WELCH, G. and BISHOP, G. (2001) *An Introduction to the Kalman Filter*, from SIGGRAPH. University of North Carolina, Los Angeles on 12 August.

Welch, G and Foxlin, E. (2002) Motion tracking: No silver bullet, but a respectable arsenal. *IEEE Computer Graphs and Applications*, 22(6), pp. 24-38

Welch, G et al. (2007) *Measurement Sample Time Optimization for Human Motion Tracking/Capture Systems*. In: *Proceedings of Trends and Issues in Tracking for Virtual Environments, Workshop at the IEEE Virtual Reality 2007 Conference*, Charlotte, NC USA, March 2007.

Wren, C., Azabayejani, A., Darrel, T. and Pentland, A. (1997) Real-time tracking of the human body. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(7), pp.780-785

Utsumi, A., Mori, J., Ohya, J. and Yachida, M. (1998) Multiple-human tracking using multiple cameras. In: *International Conference on Automatic Face and Gesture Recognition (AFGR98), Nara, April 1998*. pp. 498-503

Williams, R. (1962) *Linear State Space Control Systems*. Hoboken, New Jersey: Wiley. C2007

Yang, X. (2004) *Image Segmentation*. In: IP seminar for Fall semester in 2004

Yilmaz, A., Javed, O., and Shah, M. (2006) Object tracking: A survey. *ACM Computing Surveys (CSUR)*, 38(4)

Zarchan, P. and Musoff, H. (2001) Fundamentals of Kalman filtering: A Practical Approach. Virginia: American Institute of Aeronautics and Astronautics.