The copyright of this thesis vests in the author. No quotation from it or information derived from it is to be published without full acknowledgement of the source. The thesis is to be used for private study or non-commercial research purposes only.

Published by the University of Cape Town (UCT) in terms of the non-exclusive license granted to UCT by the author.

University of Cape Town

The Control of Semi-Autonomous Robots

Prepared by:
Graeme McPhillips

Supervised by:
Professor J Tapson

Department of Electrical Engineering
University of Cape Town
2004

University of Cape Town

This dissertation is submitted to the University of Cape Town
in fulfilment of the academic requirements
for the Degree of Master of Science in Engineering

31 March 2004

Declaration

I declare that this dissertation is my own work. Where collaboration with other people has taken place, or material generated by other researchers is included, the parties and/or material are indicated in the acknowledgements or references as appropriate.

This work is being submitted for the Master of Science Degree in Electrical Engineering at the University of Cape Town. It has not been submitted to any other university for any other degree or examination.

Signed by candidate

6 JUNE 2004

Graeme McPhillips

Date

Acknowledgements

I wish to thank the following individuals and organisations for their invaluable assistance:

- Professor Jon Tapson for proposing and supervising this research
- De Beers for their generous sponsorship towards this work
- Andrew Sass, Stephen Marais and Bjorn Prenzlow for their continued advice and assistance
- Professor Martin Braae for his advice regarding control theory
- Fred Nichols for all his work on the initial vision system and Jerome Francis for his efforts in trying to successfully implement the vision system
- Martin Beyers for his hard work and dedication to the project
- Chris Wozniac and Malcolm Attfield for their assistance with the project
- The undergraduate thesis students whose work has aided in the progress of this project
- The teams from UP and UND who participated in the 2003 F180 RoboSoccer competition in Cape Town
- Janeen McPhillips for her invaluable support and assistance throughout this thesis.

University of Cape Town

Synopsis

Robotic soccer is an international area of research which involves multiple robots collaborating in an adversarial and dynamic environment. Although many different forms of robotic soccer are played, the University of Cape Town (UCT) chose the RoboCup small-sized robot league, officially known as the F180 RoboSoccer league, as a means of pursuing robotics research within the institution.

The robot soccer game is played between two teams of five robots on a carpeted surface that is 2.8 m long by 2.3 m wide. The robots have their own on-board controllers that execute instructions sent to them from a computer-based artificial intelligence (AI) system. In order for the AI system to keep track of all the robots and the ball (an orange golf ball), a global vision system is utilised. This global vision system uses images captured from either one or multiple digital cameras mounted above the field of play to determine the position and orientation of the team's robots, the position of the other teams' robots and finally the position of the ball.

In the true spirit of competition and furthering research, the rules which govern F180 RoboSoccer league cover only the basic format of the game thereby leaving various aspects of the robots, global vision system and AI design open for development.

Since there was no RoboSoccer research in existence at UCT prior to the inception of this researcher's Masters' thesis the task included both the establishment of this format of robotics research at the institution as well as the actual design and development of the robots and the associated components as outlined below.

Developing a team of robots requires a wide array of knowledge and the research undertaken was accordingly broken into three key components; the design of the robots (which included their related electronics and on-board controller), the design of a vision system and the design of an AI system. The main focus of this author's work was on the design of the robots as well as the overall structuring and integration of the UCT F180 RoboSoccer team. In addition, the areas of the global vision system and AI system that were covered within the scope of this thesis, are also presented.

Prototypes were developed and in the first the main emphasis was placed on the movement of the robot, with the design of the kicking mechanism only occurring subsequent to this. After the first competition in 2002, this first design was abandoned in favour of developing a simpler robot with which to continue development. This simpler robot became the second

University of Cape Town

prototype which, after testing, was refined into the competition robot for 2003.

During this period, the AI and global vision systems were developed by undergraduate thesis students. This research was then incorporated where applicable and, finally, the residual problem areas were again addressed by a collaboration of staff and students.

Whilst the design and implementation of the robots was very successful, the vision system was not successfully implemented before the competition in 2003. Although an autonomous game of soccer was not successfully played in the 2003 competition, the UCT F180 RoboSoccer team had made a great deal of progress towards this goal and, consequently, a strong foundation for future robotic soccer research within UCT has been established.

The work presented within this dissertation is the author's own work except where explicitly stated.

University of Cape Town

Contents

University of Cape Town

University of Cape Town

University of Cape Town

University of Cape Town

University of Cape Town

University of Cape Town

University of Cape Town

University of Cape Town

List of Figures

University of Cape Town

University of Cape Town

University of Cape Town

University of Cape Town

List of Tables

University of Cape Town

Chapter 1

Introduction

1.1 Introduction

RoboCup is an international research and educational initiative that provides a platform through which a team of robots compete in a dynamic environment to achieve a common goal of playing a game of soccer. The University of Cape Town (UCT) has chosen the F180 RoboSoccer league [1] as a means of promoting this research within the institution.

The F180 RoboSoccer is played between two teams of five robots as shown in Figure 1.1. All aspects of the robots and gameplay are governed by the official F180 rules [2].

The robots are limited to a height of 150 mm, an outer diameter of 180 mm, and may incorporate dribbling and / or kicking mechanisms. The game is played on a green carpeted field 2.8 m long by 2.3 m wide with an orange golf ball. A global vision system uses an overhead camera mounted 3 m above the field to track the position and orientation of the robots and this information is then used by the artificial intelligence (AI) to develop gameplay strategies. The AI then sends appropriate commands to the robots via a wireless communications link.

As such, the design and implementation of a successful F180 RoboSoccer team requires the seamless integration of many components, providing an intellectually challenging area of research to stimulate interest at both undergraduate and postgraduate levels.

It is clear that it is this type of research that will drive the development of autonomous robots in the future. Such robots will be required to accomplish tasks in environments that are not only hazardous, but are constantly changing. Through the use of distributed

University of Cape Town

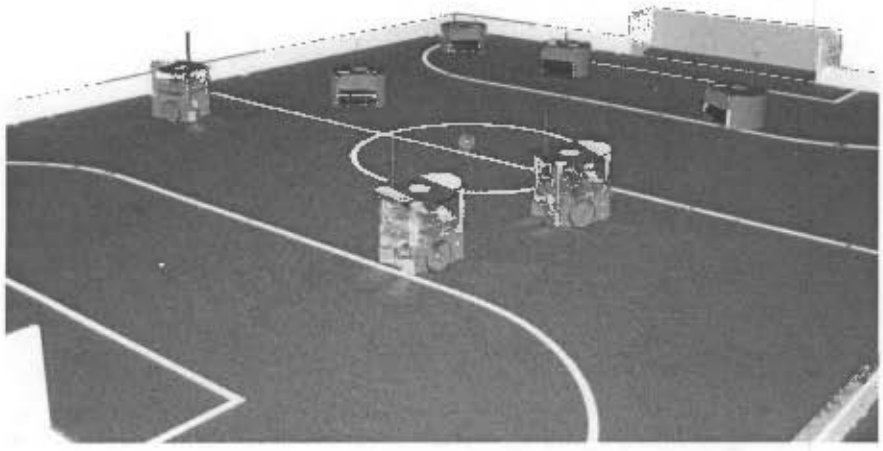


Figure 1.1: A F180 RoboSoccer game
This shows a game in progress between the University of Cape Town (4 robots in the background) and the University of Natal (3 robots in the foreground).

computing, the robots will be able to achieve tasks in the real-time dynamic environments with which they are faced.

Not only will these robots work on individual tasks, but also function as a team working to achieve a common goal. It is only through new technologies and ongoing research that co-operative and distributed behaviour within a dynamic system will become a reality.

1.2 Objectives

The primary objective is the formation of a UCT F180 RoboSoccer Team[1] with which to compete against other institutions in 2002. These include the University of Pretoria and the University of Kwazulu-Natal who are also involved in this research.

To reach this objective, three key components were identified as being pivotal to the success of the project. These were the design and implementation of:

- the robots and their associated components (both mechanical and electronic)
- an AI system
- a vision system

[1] A 'Team' means a set of five robots able to play an autonomous game of soccer.

University of Cape Town

Due to the scope and diverse nature of these components, the basis of this Masters thesis was the design and implementation of the five robots and their associated components. In addition, the overall specifications and integration of all the key components were to be considered. The AI and vision systems were assigned as undergraduate research topics that would be integrated into the system before the competition.

(At the end of 2002, the University of Pretoria hosted the first F180 RoboSoccer competition. Having only had a single semester of research it was regarded more as an opportunity to gain experience from the other teams. Unfortunately, the University of Natal was not ready to compete and withdrew. With only the two teams remaining, and both experiencing problems, it was decided to postpone the competition to the following year. Despite all of this, it was a good learning experience as UCT was introduced to another version of robot soccer, where the use of simple robots was very effective.

The 2003 F180 RoboSoccer competition was held in December and was hosted by UCT at the MTN Science Centre in Canal Walk, and, as such, was the proving ground for the 2003 UCT F180 RoboSoccer team.)

1.3 The 2003 UCT F180 RoboSoccer Team

The final robot design used for the UCT team is shown in Figure 1.2. As can be seen, the robot was semi-circular in shape with a rotating kicker / dribbling mechanisms at the front. The colour circles on top of each robot were used by the vision system to track the robots during autonomous game-play. Also depicted are the RF antennae and on / off switches.

University of Cape Town

1.4 Plan of Development

This dissertation follows the development of the UCT team from the initial stages in 2002 until the 2003 F180 RoboSoccer competition in Cape Town. Initially a background study of suitable concepts was undertaken (Chapter 2) before the design of the first prototype (Chapter 3) that was used in the inaugural South African competition in 2002. Thereafter, drawing on the experience gained in Pretoria, a second prototype was designed and tested (Chapter 4) before the final model was implemented (Chapter 5) in the 2003 competition.

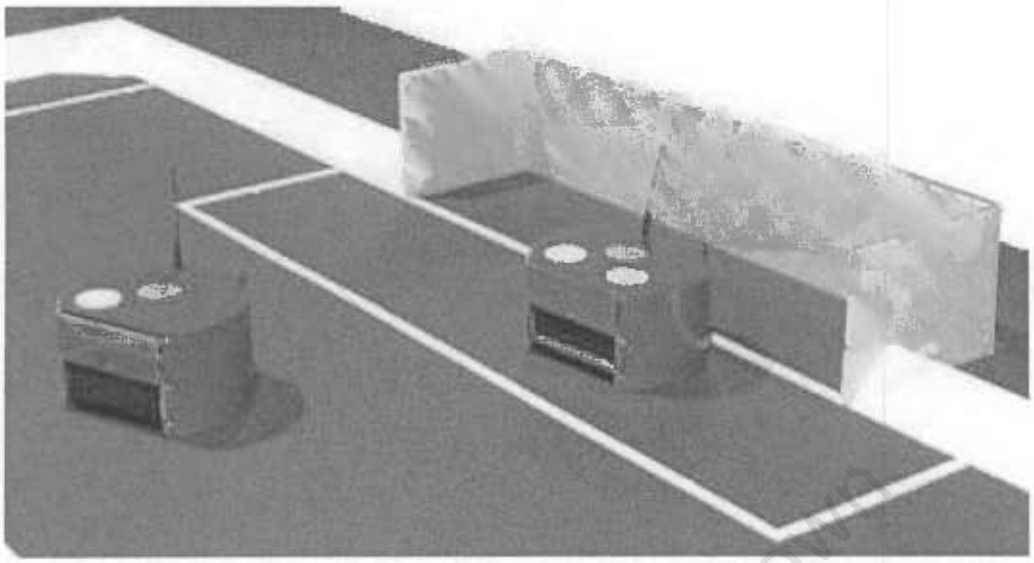


Figure 1.2: Two 2003 UCT F180 RoboSoccer robots
This shows the two robots in front of the yellow goal with their individual markers for the vision system.

An outline of the mechanical design is presented first followed, in turn, by a description of the electronics used (Chapter 6) and an outline of the associated software (Chapter 7). After completion of the robot design, the AI (Chapter 8) and vision systems (Chapter 9) were addressed and the specifications and outlines of these systems are subsequently presented. Then, the control theory that had to be designed and implemented in order to improve the predictability of the robots following trial testing of the integrated systems is described (Chapter 10). The dissertation ends with concluding remarks and recommendations (Chapter 11) for future UCT RoboSoccer research.

University of Cape Town

Chapter 2

Conceptualisation of the Mechanical Design

University of Cape Town

2.1 Design Goals

The underlying goal was to achieve a robot design that was not only agile but also omni-directional, since omni-directional robots had proven themselves as the dominant design in F180 RoboSoccer [3, 4, 5].

In the initial design stages, it was decided to establish a baseline performance benchmark of a linear velocity of $1.0\,ms^{-1}$ and acceleration of $1.5\,ms^{-2}$. These benchmarks were decided upon, in part, on the basis of earlier work that had been done by this researcher [6]. In addition, these benchmark values corresponded well with other RoboSoccer literature [4].

The other fundamental design goal was to incorporate a kicking mechanism into the robot as this would greatly increase the chance of scoring since, provided the robot was in possession of the ball and had a clear shot on goal, a direct shot could be taken without needing to dribble the ball into the goal.

2.2 Conceptual Ideas

Since there was no previous research into omni-directional robots by either the existing F180 RoboSoccer research group or other research groups within the University, it became necessary to do a background study on omni-directional robots.

A review of the current literature that discussed some of the international F180 RoboSoccer teams revealed that the early designs were simple two-wheeled robots that employed skid pads at the front and rear to provide stability [7, 8] and, as such, were based on the Turtle robot design [9]. The later designs incorporated Stanford Wheels which were used in different configurations in order to provide omni-directional movement [4, 3]. The principle of the Stanford Wheel was also used by Cornell University for their omni-directional robots [5]. Key features of each of these designs are discussed in the following sections.

2.2.1 Turtle robot

The Turtle robot, although not omni-directional, was considered for its simplicity and known ability within F180 RoboSoccer [7, 8].

The general layout for the Turtle robot is shown in Figure 2.1 and, as can be seen, the Drive Systems are mounted directly opposite each other with the wheel axis running through the centre of gravity (COG). Skid pads are placed at both the front and rear of the robot to maintain stability.

Due to the nature of the design, the Turtle robot cannot move laterally[1], but is able to change direction by either spinning on the spot or whilst moving forwards or backwards. This directional change is achieved through driving the wheels at different speeds and, due to this, the Turtle robot achieves a high degree of manoeuvrability.

2.2.2 Pluto mobile robot

In attempting to overcome the inability of the Turtle robot design to move laterally, one option considered was to rotate the drive systems to face the required direction of motion. The wheels could then again be driven at different speeds in order to introduce rotation while moving. However, for this to occur, the drive systems would need to be continuously rotated to keep them facing the required direction.

This design option forms the concept behind the Pluto mobile robot [9]. The actual layout for the Pluto mobile robot is shown in Figure 2.2. Since two wheels per wheel assembly are used, a differential is required to ensure minimal slippage of the wheels when the assemblies are rotated. Each wheel assembly can be rotated independently of the others.

[1]By not having the ability to move laterally, the robot looses its ability to face a desired direction for means of stopping or kicking the ball while moving sideways.

University of Cape Town

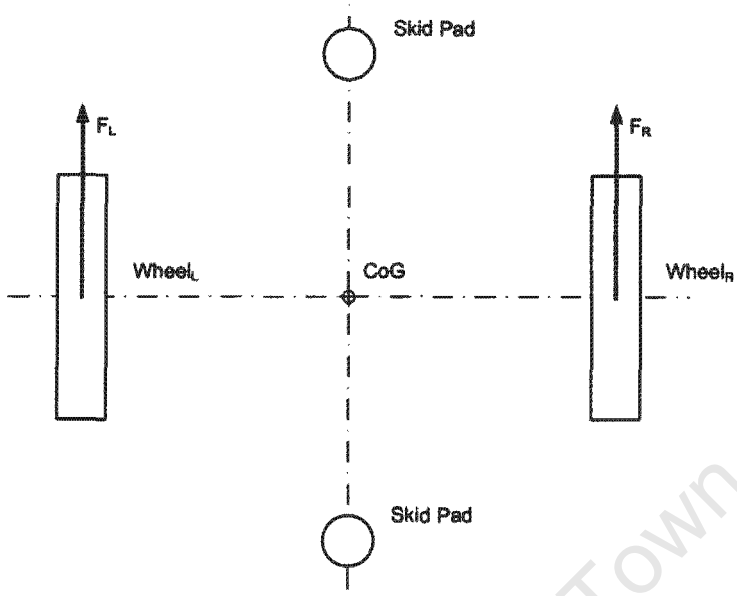


Figure 2.1: Turtle robot layout
This shows the position of the two wheels and their force vectors as well as the skid pads and the centre of gravity in the middle of the robot.

As there are three wheel assemblies, the speed of each assembly needs to be controlled in relation to the steering if the robot is to achieve a required direction of motion and rotation. The kinematics of this design therefore proved too complex for two eight-bit microprocessors to control the robot in real time.

As such, the concept behind the Pluto mobile robot would have needed to be simplified if it were to be considered for the UCT F180 RoboSoccer robots.

University of Cape Town

2.2.3 Stanford Wheel

The success of the Stanford Wheel in omni-directional robots stems from its capability of sideways motion [9].

The Stanford Wheel consists of a circular hub surrounded by rollers which run perpendicular to the hub (Figure 2.3). This translates into two primary modes of motion: rotation of the hub about its axis without the rollers spinning, and translation along the direction of the axis of the hub with the rollers spinning and the hub stationary about its axis. Motion in any other direction is achieved through a combination of these two primary modes.

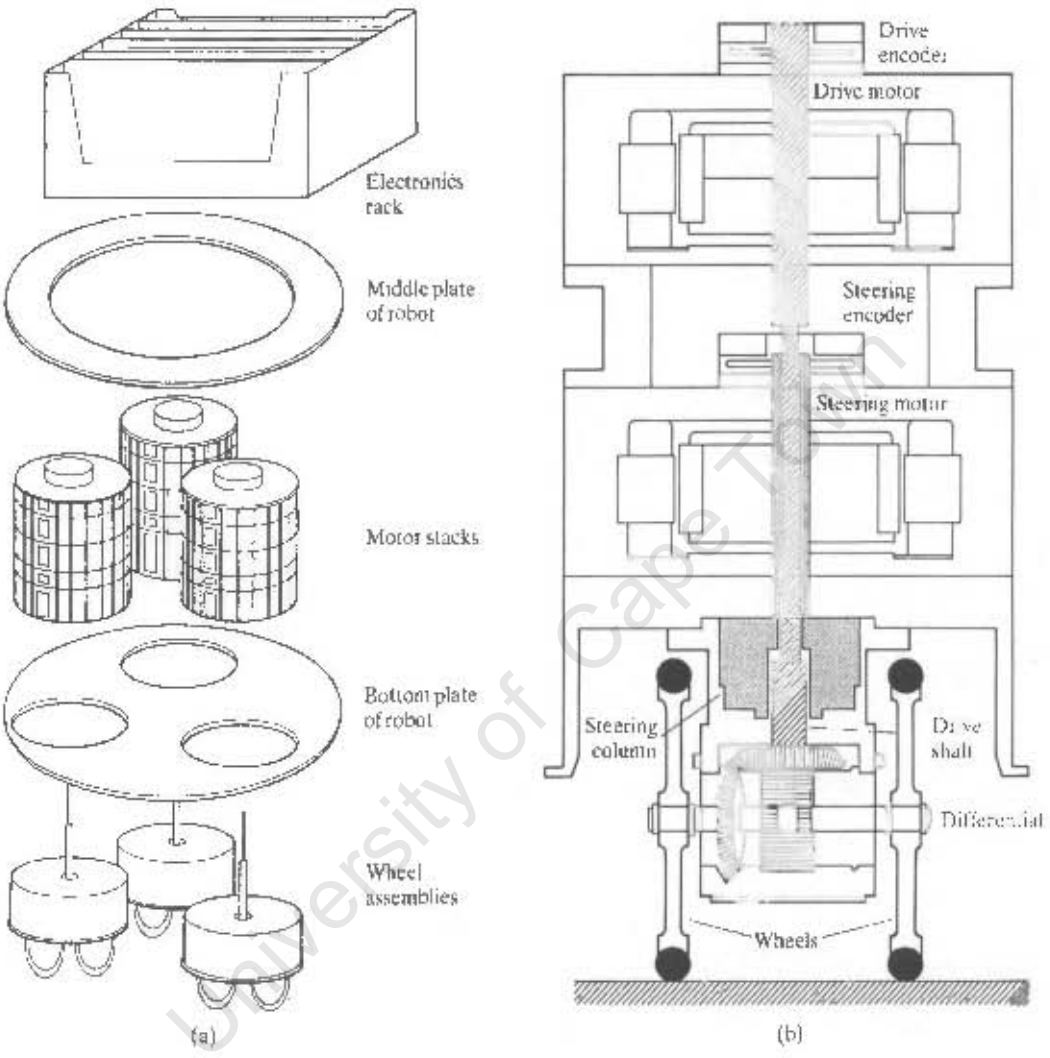


Figure 2.2: Illustration showing the Pluto mobile robot as extracted from 'Introduction to Robotics' Figure 2.6 [9]
This shows the overall layout of the Pluto mobile robot in (a) with the detail of the wheel assembly and motor stack depicted in (b). The details to note were the use of a differential and a central drive shaft which ran through the steering column. Both drive and steering motors were located on the drive shaft axis.

University of Cape Town

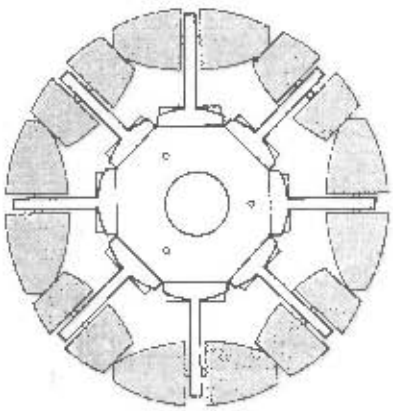


Figure 2.3: Illustration showing the Stanford Wheel as extracted from 'Introduction to Robotics' Figure 2.9 [9]
The Stanford Wheel is shown from the side view with the hub in the centre and the rollers around the circumference. The rollers are curved to ensure a constant wheel diameter.

The simplest layout consists of three Stanford Wheels mounted 120 degrees apart on a circumference (Figure 2.4). Although this layout overcomes the slight variation in diameter of the wheels (the gaps between the rollers) it does make the robots more susceptible to becoming handicapped should any of the rollers jam.

As the wheels are driven, each wheel applies a tangential force f_n to the ground. The resultant force vector f (Figure 2.4) determines the direction of motion of the robot. The resultant motion vector can then be broken-up into a rotational vector and a translational vector.

The three Stanford Wheel layout was used by the RooBots [3] in 2001 and 2002 whilst Cornell [5] used a three wheel layout in 2001 which was then changed to a four wheel layout for 2002 and 2003. The Stanford Wheel was also a common choice amongst other F180 teams at the 2003 RoboCup tournament, as evidenced in video footage of the games [10].

2.2.4 Illanator Wheel

The last of the concepts that were considered was the Illanator Wheel [9]. The Illanator Wheel is similar in concept to the Stanford Wheel but, in contrast, has rollers not perpendicular but rather at forty-five degrees to the direction of rotation of the wheel (Figure 2.5).

University of Cape Town

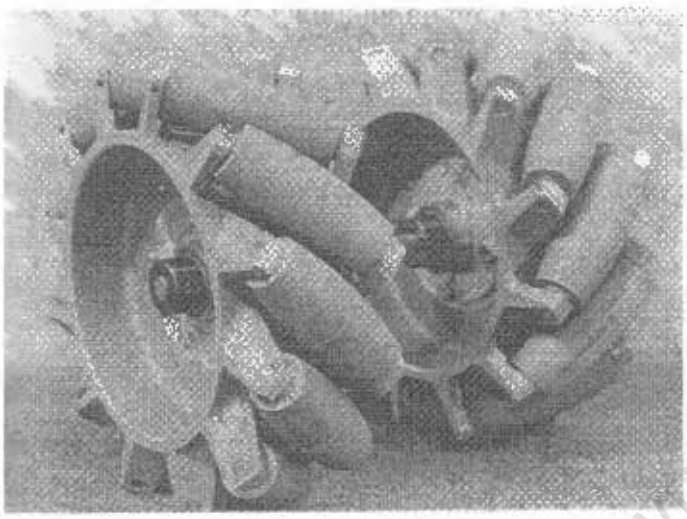


Figure 2.5: Illustration showing the Illanator Wheel as extracted from 'Introduction to Robotics' Figure 2.10 [9]
Both the left and right hand Illanator Wheels are shown in the illustration. Note how the rollers are placed at a forty five degree angle around the circumference of the wheel.

that the wheels would have needed to be designed and manufactured as the wheels were seemingly unavailable off the shelf.

Since the choice of an Illanator Wheeled robot would have required the design and manufacture of both a left and right wheel (as these were also unavailable off the shelf), it was decided that this option would not be pursued.

In the consideration of the Pluto mobile robot, although overly complex, it was felt that if only two-wheel assemblies were used in place of the three in the standard design, the kinematics could be greatly simplified. Skid pads could then be used to ensure that the robot would not topple over when all the wheels were on the same axis. Providing a small differential could be found, the Pluto mobile robot concept would be feasible design option.

In line with the design goals, the Turtle robot was not considered as it was not omni-directional.

In order to make a final decision, the Stanford Wheel and the Pluto mobile robots were analysed and compared in terms of their:

- Availability of components
- Ease of manufacture

University of Cape Town

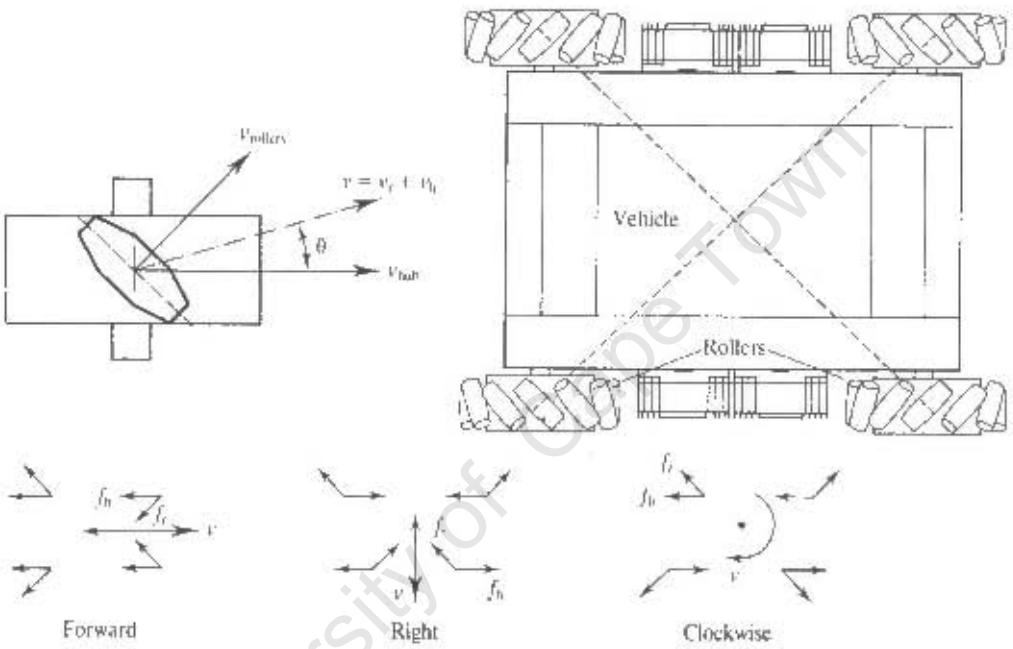


Figure 2.6: Illustration showing the layout and force relationship when using four Illanator Wheels as extracted from 'Introduction to Robotics' Figure 2.12 [9]
This shows the layout of a four-wheeled vehicle using the Illanator Wheels, as well as the resultant velocity vectors (v) when certain forces are created. The resultant vectors shown are for forward, sidewards and rotational movement.

University of Cape Town

2.3.1 Availability of Components

As each design had a key component which would be required, research was done in order to locate these two components.

When sourcing requirements for the Pluto robot a small differential was found at a local radio-controlled model store. Along with the differential, there was a reasonable collection of plastic gears and motors available from within the store that could be physically chosen as required. As most of the parts were small and reasonably priced, a collection of gears was purchased in order to begin the modelling process.

Initially, since a source for the Stanford Wheel could not be found, there was no other choice but to attempt to manufacture the wheels. Whilst in the process of testing Prototype Two (Chapter 4) a design was found that used Transwheels[2] (which were similar in design to Stanford Wheels) for a goalkeeper robot design [4]. Although too late for implementation in the UCT F180 RoboSoccer 2003 campaign, a set of Transwheels was purchased and a development model built (Figure 2.7). This model then became the focus of an undergraduate thesis project [11] with the aim that it might still prove beneficial in the design of future robots.

2.3.2 Ease of Manufacture

If the Stanford Wheel robot was to be considered, it had become apparent that the wheels would have to be designed and manufactured within UCT. Following consultation with the Design lecturers in Mechanical Engineering, it was decided that the Stanford Wheels as shown in Figure 2.3 would be too complex to design manufacture within UCT.

This then left the Pluto mobile robot as the only remaining option as parts would be available locally and the manufacture of the remaining parts could be done within UCT.

2.4 Concluding Remarks

As a result, despite the greater complexity of the Pluto mobile robots, this design was still chosen to form the basis of the UCT F180 RoboSoccer team and a prototype was

[2]Transwheels are a registered trademark of the Kornylak Corporation: www.kornylak.com

University of Cape Town

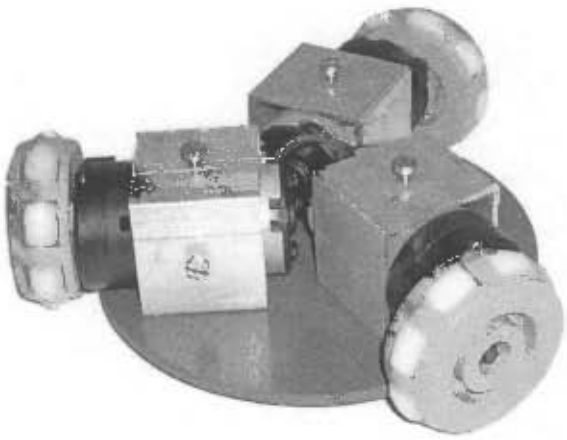


Figure 2.7: Transwheel prototype model
This is the first prototype design using the Transwheels. The wheels were mounted directly to DC geared motors which were, in turn, held in place by square aluminium tubing fastened to a plastic disc. As such, this robot formed the basis for an undergraduate thesis on omni-directional robots using Stanford Wheels.

subsequently built. This decision being based on the availability, design and manufacturing constraints discussed above.

University of Cape Town

Chapter 3

Prototype One: Design and Evaluation

3.1 Introduction

During the feasibility study, an assortment of gears was purchased in order to determine whether or not the concept behind the Pluto-mobile robot design would be practical for F180 RoboSoccer. Also purchased at this time was the differential which was vital to the design. As all the components thus far had been chosen from those used in radio-controlled (RC) model cars, it was felt that the DC motors used by such cars would provide the necessary power to ensure that the robot would be both fast and agile.

Having collected most of the required parts, a "Bottom-up Approach" was adopted in which the main emphasis was placed on the wheel assemblies. The initial design of this prototype had independent direction control for each wheel assembly as depicted in Figure 3.1. However, the dynamics of this design proved too complex given the available development time before the 2002 competition and therefore a simpler design, which used a single motor to control the direction of both wheel assemblies, was finally implemented . This design is illustrated in Figure 3.2.

Since emphasis had been placed on the wheel assemblies, the kicker mechanism was only designed after the completion of the drive system and chassis. As a result, the size of the kicker was quite restricted as is evident in Figure 3.2. In this prototype the open space at the back of the robot was left vacant for the batteries and electronics. Not shown in Figure 3.2 are the skid pads used to maintain the balance of the robot when all the wheels were on the same axis.

University of Cape Town

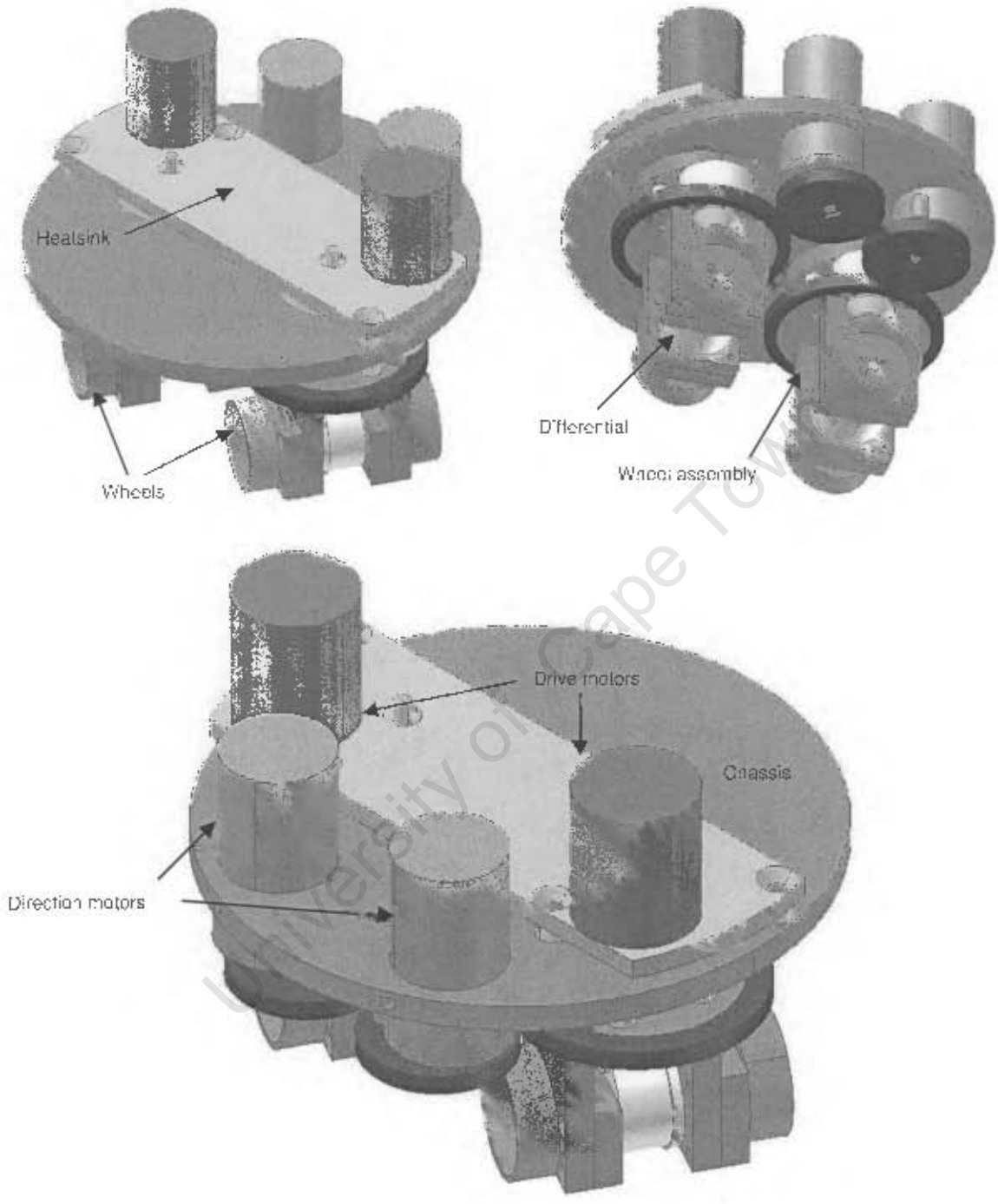


Figure 3.1: Prototype One with independent direction control wheel assemblies
This shows the initial design where both wheel assemblies were independently direction controlled allowing for greater freedom of movement. Note the use of a heat sink for cooling the high current drive motors.

University of Cape Town

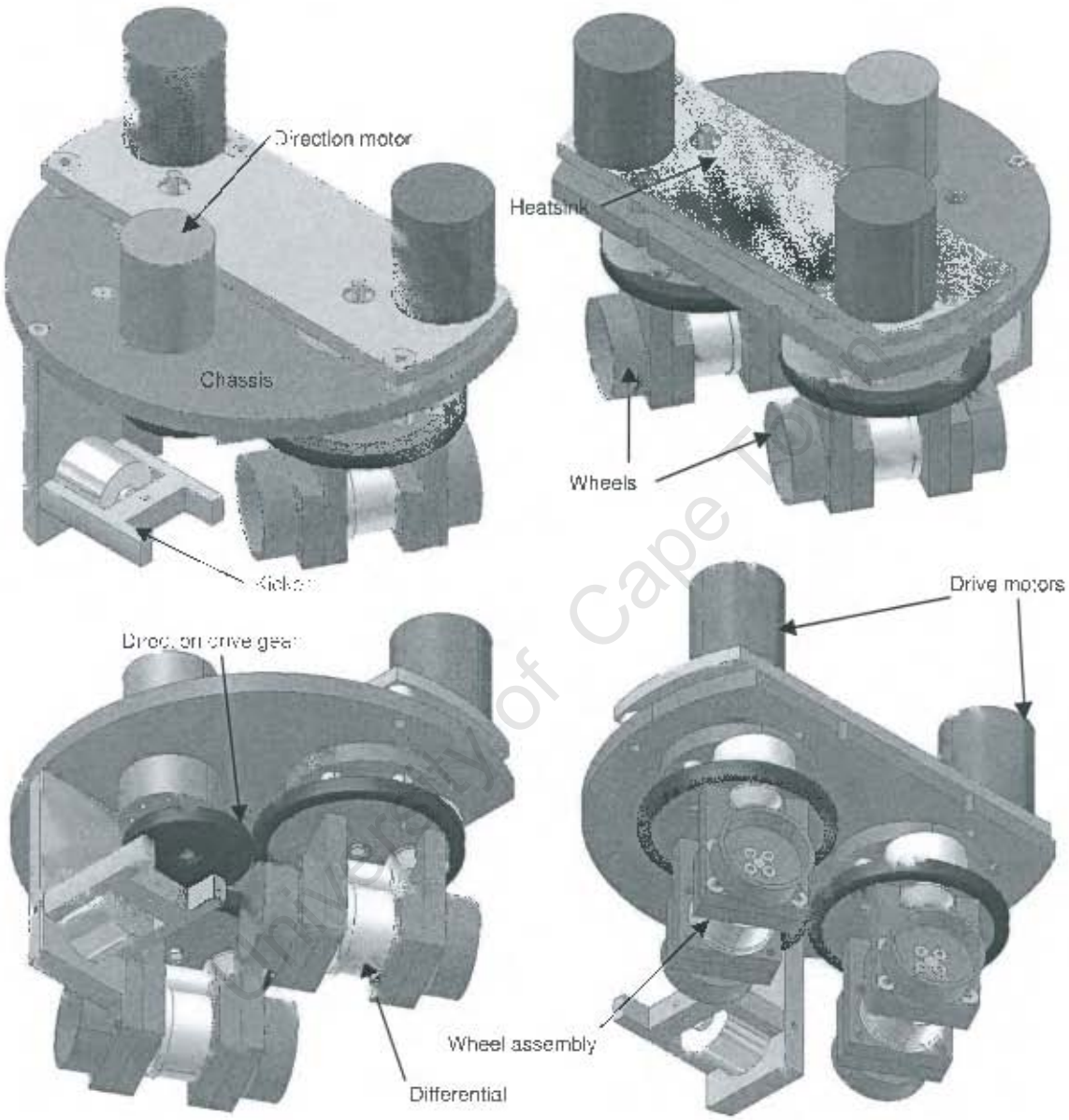


Figure 3.2: Prototype One with joint direction control wheel assemblies
This shows the simplified design where a single motor controls the direction of both wheel assemblies simultaneously. The kicker as implemented for 2002 is also visible.

University of Cape Town

3.2 Design

3.2.1 Choice of components

Once the components for the wheel assemblies had been selected, the direction control and kicker components needed to be chosen. Finally, the power requirements were known which allowed appropriate batteries to be selected.

Motor selection

In order for the direction of the wheel assemblies to be controlled a large plastic gear was incorporated and, as a result, a high torque low speed motor or geared motor would be required to drive this gear. Having had success with the use of a 16 V DC geared motor (Figure 3.3) in a previous project [6], this option was chosen for the drive systems.

Figure 3.3: 16 V DC geared motor
This shows the motor that was used to control the direction of the wheel assemblies.

Taking the space requirements into account and considering that there was only a small selection of DC motors available from which to choose, a high torque 9V DC motor (Figure 3.4) was chosen for the kicker mechanism.

University of Cape Town

Figure 3.4: $9\,V$ DC motor
This shows the motor that was used for the kicker.

Battery selection

In order to meet the power requirements of the two RC $7.2\,V$ DC motors, NiCad batteries were chosen for the drive system. As each RoboSoccer match would consist of two $10\,min$ halves, the batteries would only be required to last these $10\,min$. Panasonic manufacture a low resistance $1.2\,V$ $1700\,mAHr$ NiCad Battery (shown in Figure 3.5) which met these power requirements.

Figure 3.5: Panasonic $1.2\,V$ $1700\,mAHr$ NiCad Battery
This shows the batteries which were used to make up the battery pack to supply power to the drive motors.

Since the $1700\,mAHr$ NiCad Batteries would be required to deliver high currents for the drive motors, it was decided that the electronics would require separate batteries. As these power requirements were substantially lower, a Panasonic $1.2V$ $1000mAHr$ NiCad Battery (Figure 3.6) was chosen.

University of Cape Town

Figure 3.6: Panasonic 1.2 V 1000 $mAHr$ NiCad Battery
This shows the batteries which were used to make up the battery pack for the electronics.

University of Cape Town

Sensor selection

In this design, sensors were required for speed measurement on both wheel assemblies as well as direction control. Given that there were drive shafts on which to mount masks for optical encoders and that there were gears which could be used as masks, the choice of using optical interrupt sensors was logical. As masks were readily available from old computer mice, these were used for the speed measurement. The Fairchild H21A1 optical interrupter switch was chosen to complete the sensor.

As the gears were too thick for the H21A1 optical interrupter switch, the infra-red transmitter and photo-transistor were removed from the plastic casing and mounted on veroboard.

3.2.2 Spatial considerations

The top view of the spatial layout is shown in Figure 3.7. The chief considerations were the outside diameter of the robot (governed by the rules [2]) which in turn gave the maximum permissible diameter of the wheel assemblies. Given that the maximum diameter of the wheel assemblies was 90 mm, the gears, wheels and structure had to fit within this constraint. The positioning of the wheels and direction gears is also depicted in Figure 3.7.

The overall height of the robot also had to be considered but, as this was governed by the choice of gears for the required gearing, it is dealt with in the next sub-section.

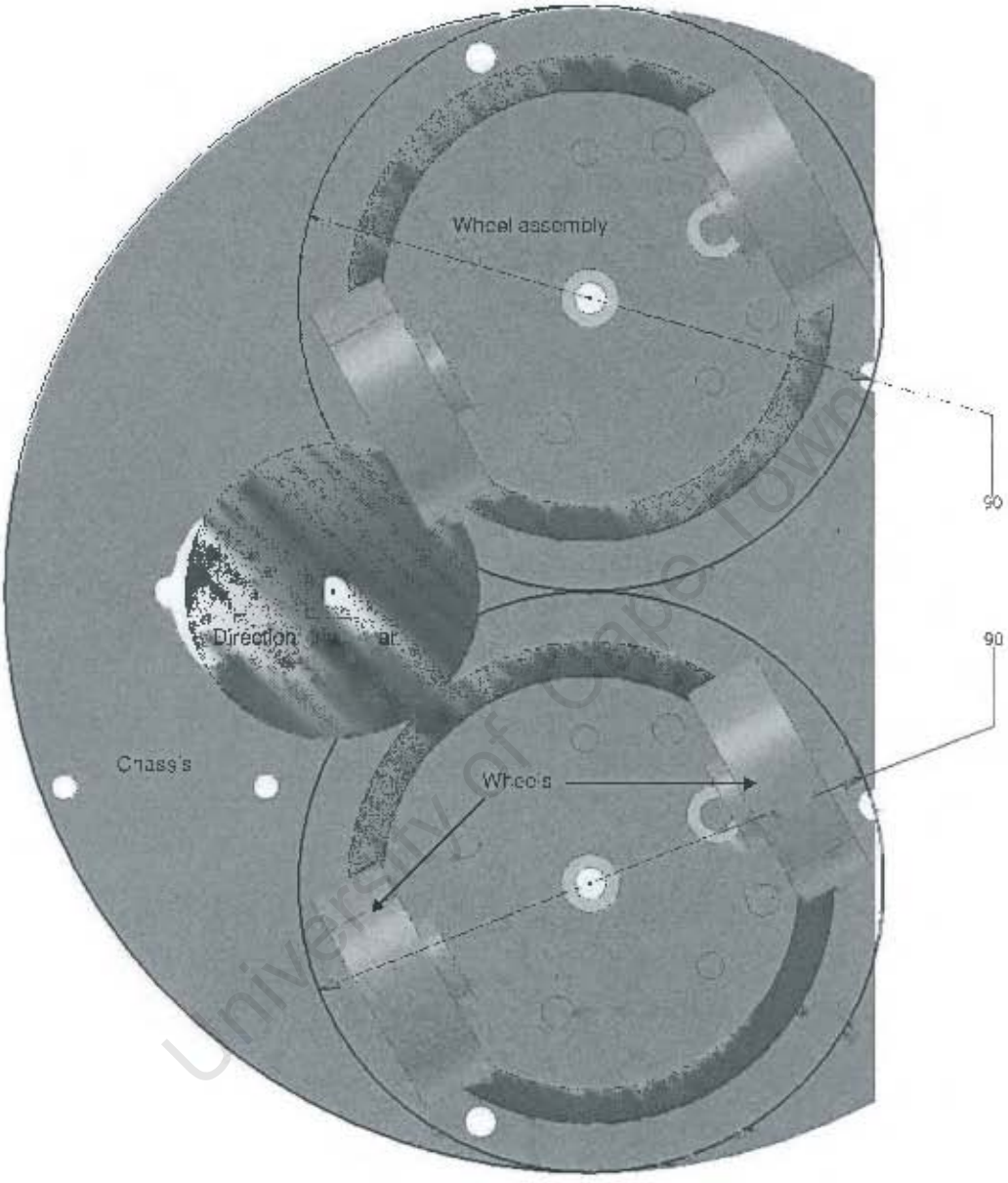


Figure 3.7: Top view of the spatial layout of the robot

This shows the spatial layout of the wheel assemblies and direction drive gear (red). The constraint for the wheel assemblies ($\phi 90$ mm) is shown in blue. All parts of the wheel assembly had to be within this constraint as the assemblies rotated about their respective axes.

University of Cape Town

3.2.3 Gearing requirements and wheel assembly design

The 7.2 V DC motor chosen together with the selection of the gears and differential was rated to 19000 rpm (7.2 V) and, as such needed to be geared down. Having laid out the gears as depicted in Figure 3.8, the final gear-ratio was $20:1$ which translated to a top speed of $2\,ms^{-1}$.

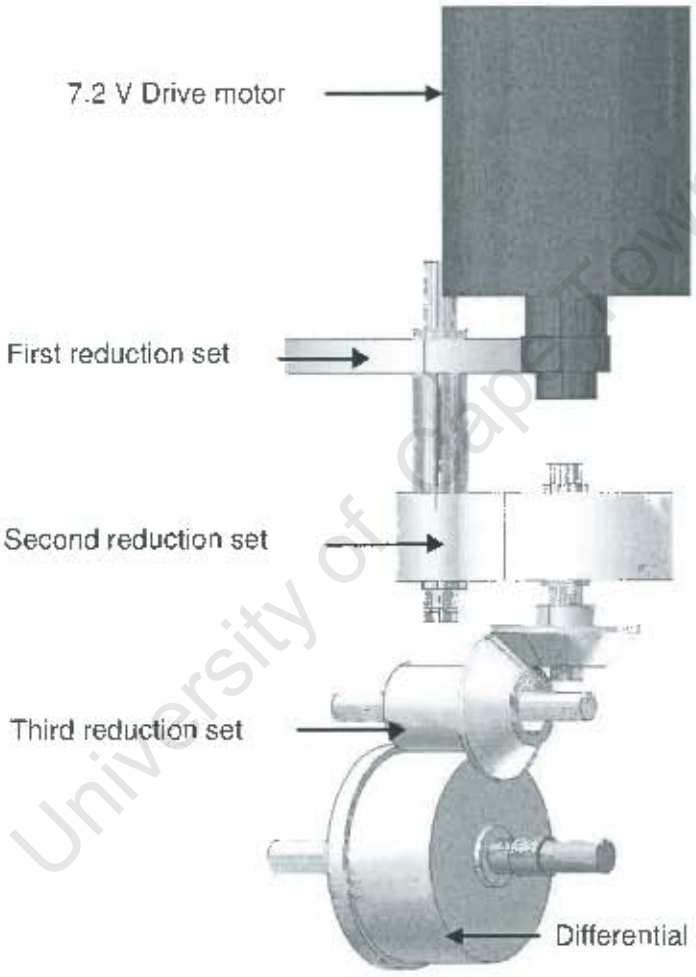


Figure 3.8: Side view of the gearing layout
This shows the overall gearing of Prototype One. The motor and first reduction set were housed within the chassis, whilst the remaining reduction sets and differential formed part of the wheel assembly.

In the design of the wheel assembly a decision needed to be taken as to whether or not the

University of Cape Town

drive motor should be included as part of the assembly.

The advantage of including the drive motor in the wheel assembly was that wheel slippage would be prevented as the assembly would rotate as a unit. The disadvantage; however, was that slip rings and brushes would be required in order to provide power to the drive motors.

In contrast, not including the drive motor in the wheel assembly meant that although no slip rings and brushes would be needed, a control system would be required to ensure that the drive motor turned in unison with the assembly so as to prevent wheel slippage when the assembly changed direction.

When the above options were considered, it was decided that the latter option was preferable since, firstly, the slip rings and brushes would be less efficient and, secondly, it was anyway intended that a control algorithm would be included for speed control. The final design of the wheel assembly is shown in Figure 3.9.

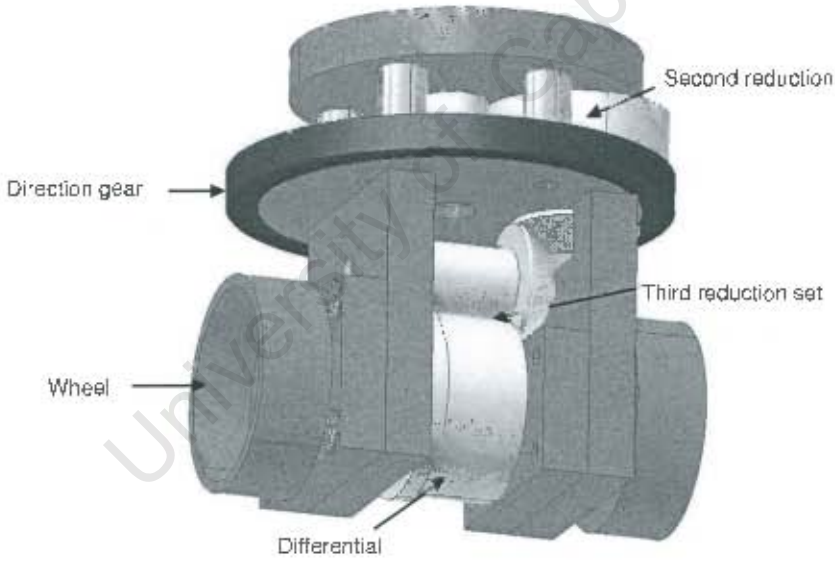


Figure 3.9: The wheel assembly
This shows the wheel assembly complete with the gears which made up the reduction sets, differential and wheels. The direction gear is shown in red.

University of Cape Town

3.2.4 Chassis design

As the chassis design was done after that of the wheel assembly, this meant that the drive motor and first gear set would need to be supported by the chassis. The main part of the chassis was made from plastic whilst the support for the drive motors was made from aluminium in order to form a heatsink for the motors. The support for the kicker was made from mild steel to ensure rigidity.

Since the wheel assembly was only supported by the main drive shaft, this shaft was also supported by the chassis through the use of bearings. The chassis is illustrated in Figure 3.10.

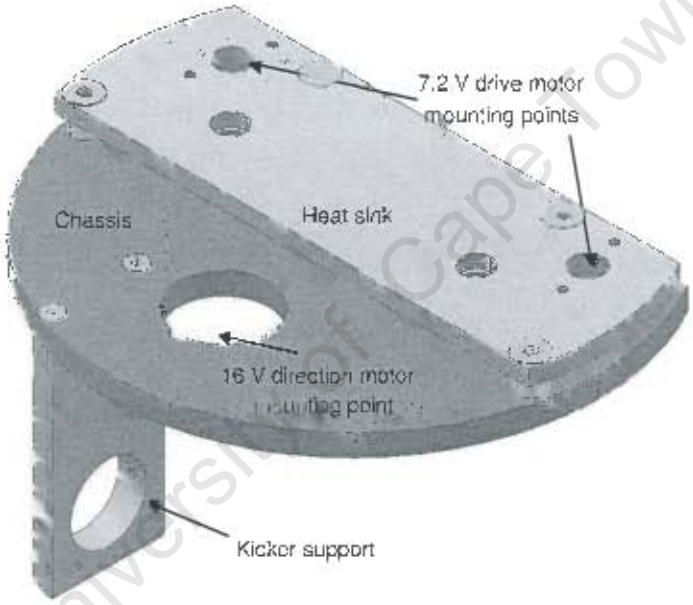


Figure 3.10: The chassis
This shows the chassis complete with the aluminium heatsink (which also served as the drive motor mounting points) and kicker support.

University of Cape Town

3.2.5 Kicker design

As previously mentioned, the design of the kicker was left until the drive system was completed and, as a result, there was very little space remaining for a kicker. Several kicker designs used by international teams [3, 4, 7] were studied, but the spinning kicker as

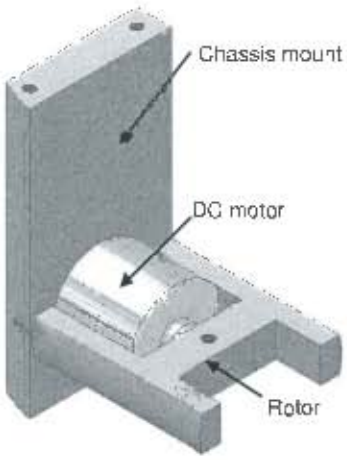


Figure 3.11: The Kicker
This shows the rotor and motor which formed the Kicker and the chassis mount which secured it in place.

used by the RooBots in 2001 [3] was the simplest to implement given the spatial constraints and the lack of development time that remained before the 2002 competition in Pretoria.

The kicker consisted of a $9V$ DC motor with the kicker attached directly to the drive shaft of the motor, as shown in Figure 3.11. In this way, not only was less space required, but there was an added advantage in that no gears or drive belts were needed. The motor was then held in place via a support structure which formed part of the chassis.

3.3 Concluding Remarks

During the building of the robot, the drive system was particularly difficult to assemble with slight modifications being required in order to compensate for manufacturing tolerances. Once completed, however, the robot proved to be very fast and agile, but unfortunately the high centre of gravity caused the robot to be prone to toppling over despite the skid pads that had been placed on the circumference of the robot.

At this point it became evident that the kicker would also require a re-design since, due to the size constraints, the width of the kicker meant that the robot was seldom correctly aligned to kick the ball. Lastly, the robot had been designed using the maximum outer diameter allowed by the rules and, as a result, there was no room for a shell around the

University of Cape Town

robot.

The design showed potential, but a re-design of the robot was required in which the centre of gravity had to be lowered, the outer diameter of the robot had to be reduced and a new type of kicker needed to be chosen. Finally, the advantages of using independent direction control for each wheel assembly, as opposed to a one motor joint direction control, meant that in future UCT robot soccer development the former should be the option of choice for this design.

Overall, after the 2002 competition in Pretoria it appeared that a simpler design would be more successful in the short term whilst all aspects of the UCT F180 RoboSoccer campaign were still undergoing development.

University of Cape Town

Chapter 4

Prototype Two: Design and Evaluation

University of Cape Town

4.1 Introduction

The birth of this prototype arose out of a strategic shift in the thinking behind the project as a whole. Instead of having the fastest and most mobile robot, irrespective of its complexity, it was decided to move to a simpler design in order to get from the design board, through prototyping and onto a production robot in as short a time period as possible.

The simplest known available design is the Turtle robot (Section 2.2.1) and this was selected as the basis for the UCT F180 RoboSoccer robots. The fact that this choice was also made by other F180 teams [7, 8] further reinforced this decision. The beauty of this design lies in the fact that only two Drive Systems and a set of casters or skid pads are required for propulsion. Each Drive System consists of a motor, gearbox, wheel and power electronics.

In order to structure the design of the new prototype a "Top Down Approach" was adopted. The macroscopic view of Prototype Two is shown in Figure 4.1. The microscopic details will be examined in detail within the relevant sections that follow later in the chapter.

The important areas to note from Figure 4.1 are the Drive System and Kicker which tie in with the design goals from Section 2.1. These can be seen in Figure 4.2. The electrical design will be detailed within Chapter 6.

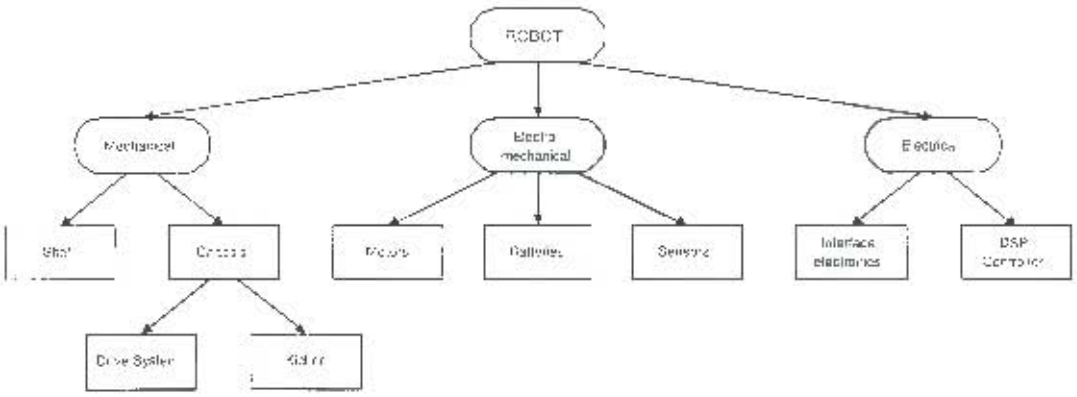


Figure 4.1: A macroscopic view of Top Down Approach
This shows the the breakup of the robot into its separate aspects, particularly the mechanical aspects of the drive and kicker systems.

4.2 Design

4.2.1 Choice of components

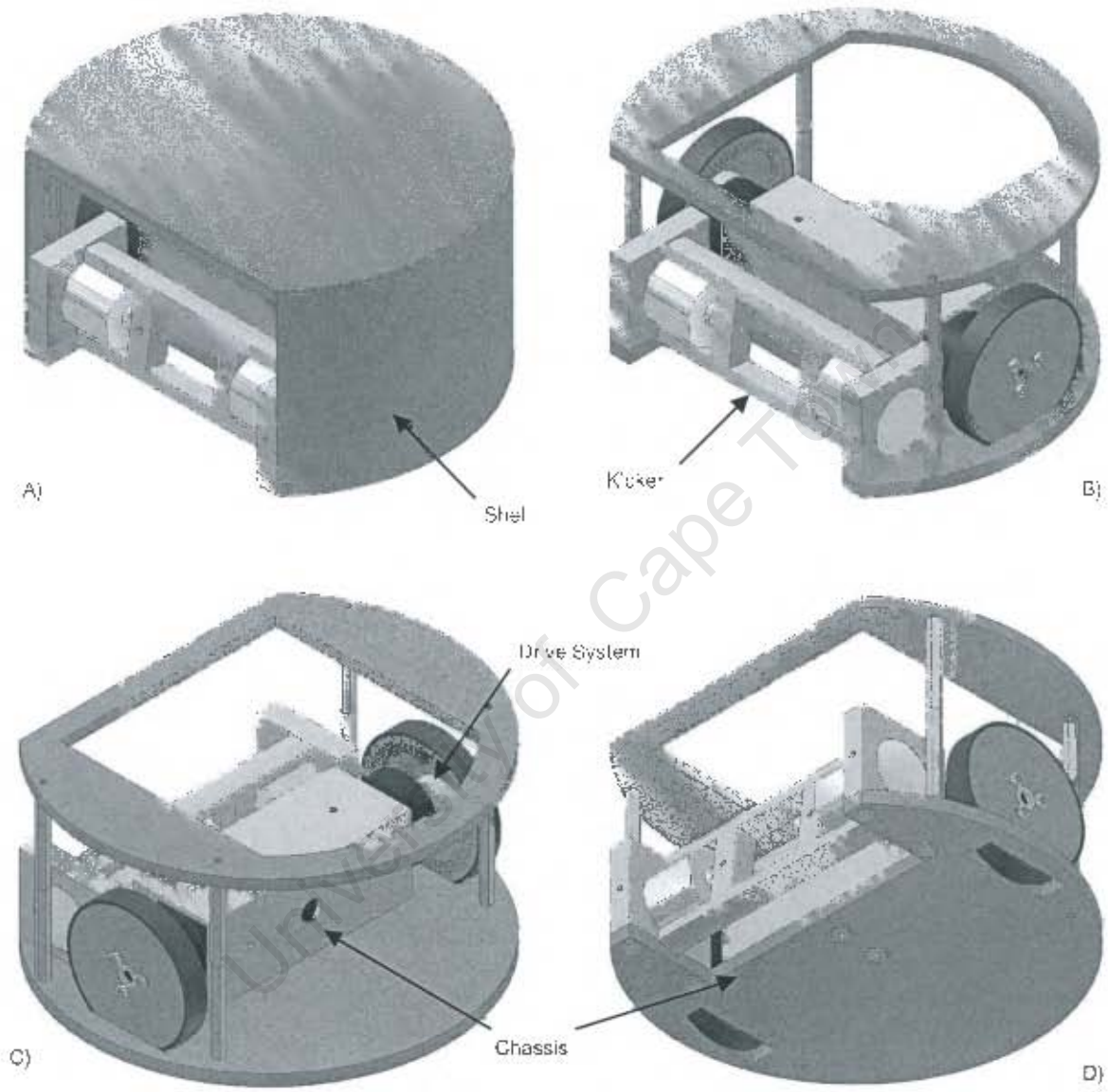
One area of the design which still requires discussion here is that of weight, since whichever components were chosen in the design and construction of the robot would have an effect on the overall mass. The two components which had the greatest effect on the mass, were the motors and batteries. Herein lay the need to compromise, as the more torque and power produced by the motors, the higher their energy requirements and therefore the greater the requirements from the battery.

Motor selection

The obvious choice for the Drive System was the DC geared motor used in Prototype One for direction control (Figure 3.3). This motor had proved reliable in Prototype One and, with its built-in gearbox, could supply sufficient speed and torque. The DC geared motor provides a speed of 425 RPM at 16 V. The DC geared motor draws 200 mA with no load whilst at full load this increases to approximately 350 mA.

In selecting a motor for the Kicker, two considerations needed to be met. Firstly, the motor had to have sufficient torque to spin the rotor up to speed quickly and, secondly, it had to be small enough to fit inside the Kicker to eliminate the need for gears or drive belts.

University of Cape Town



University of Cape Town

Figure 4.2: Prototype Two from various angles
The final design of Prototype Two is shown in (A) while (B,C,D) show the robot without the top cover and shell thereby exposing the Drive System.

Although the DC motor chosen for the kicker in Prototype One was under-powered, it was felt that the torque requirements could be met by using two such motors in parallel. The DC motors chosen for the kicker were rated at $12\,V$ DC and drew approximately $600\,mA$.

Battery selection

Since the batteries used in Prototype One were not in current use, the same National $1.2\,V$ AA NiCad $1000mAH$ cells were chosen for Prototype Two. In order to obtain a nominal voltage of $16\,V$, 14 cells were used in a series configuration. Since the electronics required only $5\,V$, the cells were divided into two packs of seven cells[1] thus giving each pack a nominal voltage of $8.4\,V$. From this arrangement both $8.4\,V$ and $16.8\,V$ were available.

To determine whether the cells would last for a full match, an approximation of load is required. In determining this it was assumed that the

- Electronics[2] drew $500\,mA$ for 100% of the time τ.
- Drive Systems drew $2 \times 350\,mA$ for 50% of the time τ.
- Kicker drew $2 \times 600\,mA$ for 10% of the time τ.

giving

$$1000\,mAH = 500\,mA \times \tau + 700\,mA \times 0.5\tau + 1200\,mA \times 0.1\tau \tag{4.1}$$

which gave

$$\tau \approx 1\,H$$

Considering a match is only $20\,min$ in total, it was decided that the batteries would sufficiently meet the energy requirements.

Sensor selection

As a result of keeping the robot design simple, the choice of sensors was greatly reduced. The principle requirement of a sensor in this design was its ability to measure the speed of the Drive System.

[1] By keeping the packs the same size, the re-charging procedure was kept as simple as possible which would be of great benefit during competition.

[2] Although the electronics drew $5\,V$, the linear regulator dissipated the current over the voltage difference and the current drawn was therefore considered to be over the full $8.4\,V$ range.

University of Cape Town

A tachometer, although very easy to use, is both expensive and, in principle, difficult to implement; therefore the standard choice in modern electrical apparatus (such as printers) is to rather use optical encoders. Although very expensive for pre-made units, infrared limit switches are inexpensive and easy to use when combined with microprocessors. Although many types of switches are commercially available it was decided, based on local market availability, to make use of the Fairchild H21A1 optical interrupter switch.

The interrupter mask was incorporated in the design of the wheel.

4.2.2 Spatial considerations

In considering the layout of the robot, the following conditions needed to be met:

- The axis of the Drive Systems needed to be as close to the COG as possible
- The width of the Kicker was required to be as wide as possible
- Space had to be left for the 16 AA cells.

As the height of the motors and AA cells was less than 50 mm, the electronics could be mounted above them and still be well within the height regulations. Through manipulating the positions of the two DC geared motors and the two DC motors within the base diameter the parts were best positioned to meet the required conditions.

As depicted in Figure 4.3, the DC geared motors were positioned centrally and as far apart as possible whilst still ensuring that the wheels would remain within the bounds of the robot. As the DC geared motors were moved outward, it became possible to increase the width of the kicker. Since the DC motors had to remain within the confines of the rotor, the internal diameter of the rotor therefore needed to be greater than the outer diameter of the DC motors themselves. However, as the rotor diameter increased, the width of the kicker in turn had to become less in order not to force the DC geared motors towards the back of the robot. This conflict was resolved at a later stage following the design of the Drive System and Kicker.

As a result of the design chosen the rear of the robot was then available to hold the 16 AA cells. Due to the cells being situated at the rear and the kicker at the front of the robot, the weight distribution remained neutral, thus keeping the COG roughly in line with the wheels.

University of Cape Town

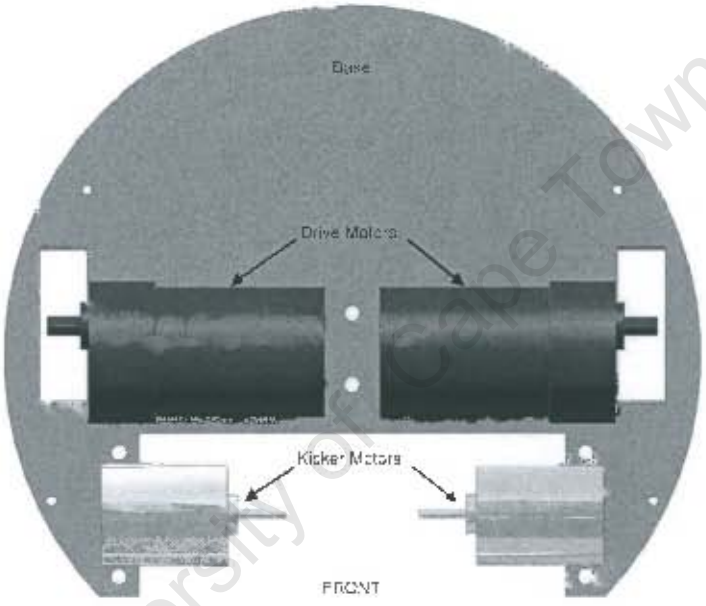


Figure 4.3: Part layout

This shows the spatial layout of the drive and kicker motors (which made up the drive and kicker systems) relative to the chassis.

University of Cape Town

4.2.3 Drive System design

The mechanical aspects of the Drive System concerned both the DC geared motor and the wheel. The method of mounting the Drive Systems forms part of the chassis.

Since the DC geared motor was a stock item, only the wheel needed to be designed. The use of door insulation foam for the tyres in Prototype One had proved very effective and was therefore again implemented. The smallest standard width available was $10\,mm$, with a minimum thickness of $3mm$, thereby rendering a minimum width of $10mm$ for the wheel. The diameter of the wheel needed to be calculated from the RPM of the DC geared motor and the speed requirement.

Given a maximum motor speed of $425\,RPM$ and a minimum speed requirement of $1\,ms^{-1}$ it follows from

$$Diameter_{Wheel} \geq \frac{Speed \times 60}{RPM_{Motor} \times \pi} \tag{4.2}$$

that the

$$Diameter_{Wheel} \geq 45\,mm.$$

As previously mentioned in Subsection 4.2.1 the mask for the optical encoder formed part of the wheel. As the width of the wheel was already set at $10\,mm$, the only space for the mask was the rim of the wheel. Thus fitting the H21A1 sensor between the motor and the surface of the wheel meant making the

$$Diameter_{Wheel} \geq 55\,mm.$$

In order to allow for easy removal of the H21A1 sensor, $2\,mm$ was added making the

$$Diameter_{Wheel} \geq 57\,mm.$$

Due to the addition of the foam tyre the wheel diameter became an extra $6\,mm$ bigger. However, due to the weight, the foam tyres compressed making the effective

$$Diameter_{Wheel} = 60\,mm.$$

Re-calculation from Equation 4.2 gave

$$Speed = 1.34\,ms^{-1}.$$

University of Cape Town

Working from a minimum cutting tool size of 1 *mm*, the CNC milling machine was able to machine 60 slots into the rim to form the encoder mask (Figure 4.4).

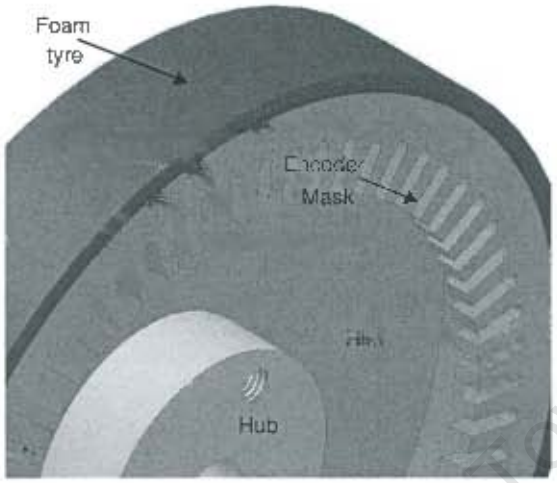


Figure 4.4: Enlarged view of the inside of the wheel
This shows a close-up view of the encoder mask on the inside of the wheel. The hub which fastened the wheel to the DC geared motors is also visible.

Through the use of a hub, a grub screw could be used to tighten the hub onto the motor shaft without the rim preventing allen key access. Thereafter the rim, complete with foam tyre, could be fastened to the hub. The complete mechanical construction of the Drive System can be seen in Figure 4.5 , with an exploded view depicted in Figure 4.6.

4.2.4 Kicker design

Although Prototype Two was distinctly different in design from Prototype One, the principle behind the kicker and its implementation could still be used for this second design. In order to bring the kicker used in Prototype One in line with the new design goals it needed to be extended in width. However, by making the kicker wider a new problem arose, as the rotor needed to be supported on both ends as opposed to just one end as in Prototype One.

After consideration of the difference in price for a bearing and the price of another DC motor, a second DC motor was chosen to support the other end. The slight difference in price was easily offset by the extra torque gained which, in turn, enabled the rotor to

University of Cape Town

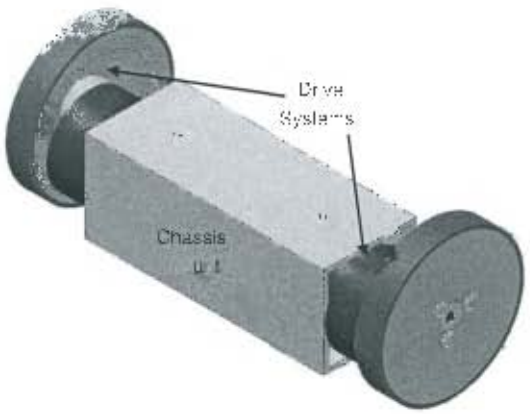


Figure 4.5: Drive System shown with chassis mount
This shows the chassis mount (made from aluminium square tube) which housed the Drive System.

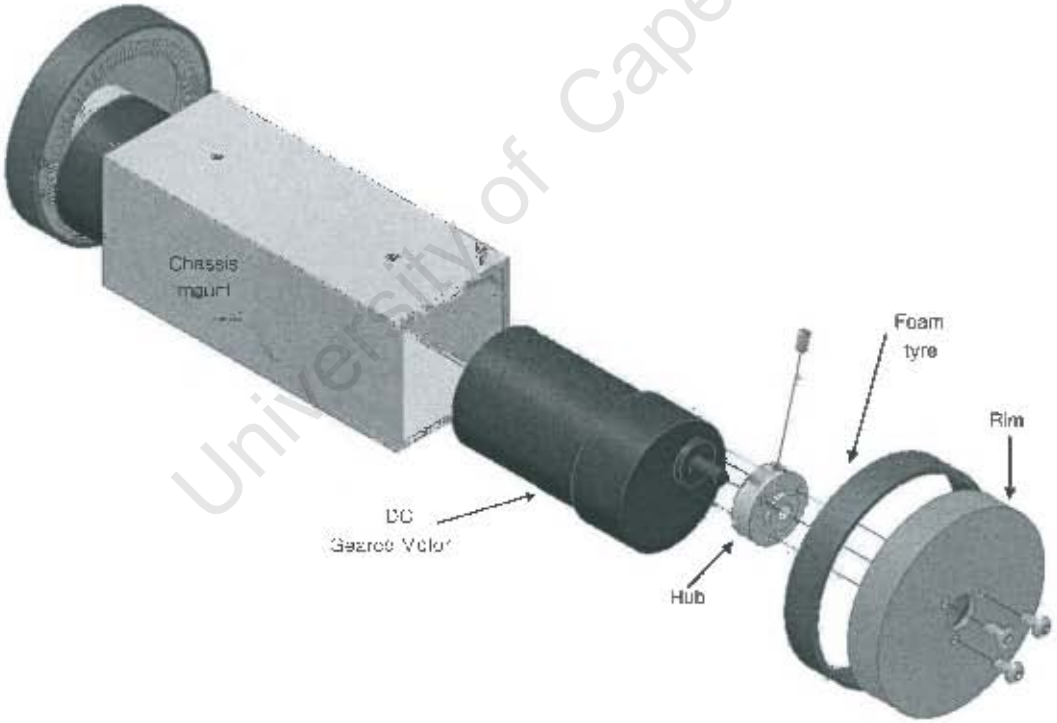


Figure 4.6: Exploded Drive System with chassis mount
This shows how the different components were assembled to form the complete Drive System.

University of Cape Town

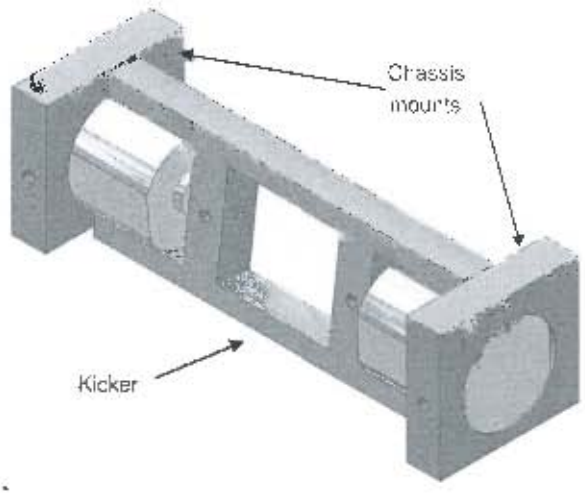


Figure 4.7: Kicker with chassis mounts
This shows the rotor and two motors which formed the Kicker and the chassis mounts which secured it in place.

accelerate more quickly. Just as in Prototype One, the DC motors made up the core whilst the rotor spun around the DC motors which were supported by the chassis (Figure 4.7).

Although the minimum internal diameter of the rotor was determined by the size of the DC motor, the outer diameter of the rotor was purposefully made larger to withstand the forces experienced when kicking the ball. However, as mentioned earlier, keeping the outer diameter as small as possible in turn allowed the width of the Kicker to be maximised. It was thus necessary to reach a point of compromise and in resolving this the final width of the Kicker was determined on completion of the Chassis design (Subsection 4.2.3).

An exploded view of the Kicker is shown in Figure 5.3. As can be seen in both Figures 4.7 and 5.3 the centre of the rotor was cut away to help reduce the mass of the rotor. In addition, the use of aluminium instead of steel for the rotor further assisted in reducing the mass of the rotor. A lower mass and therefore a lower inertia aided the rotor in reaching its kicking speed more quickly. Keeping the rotor symmetric helped to minimise any vibrations that could be introduced into the system during kicking.

University of Cape Town

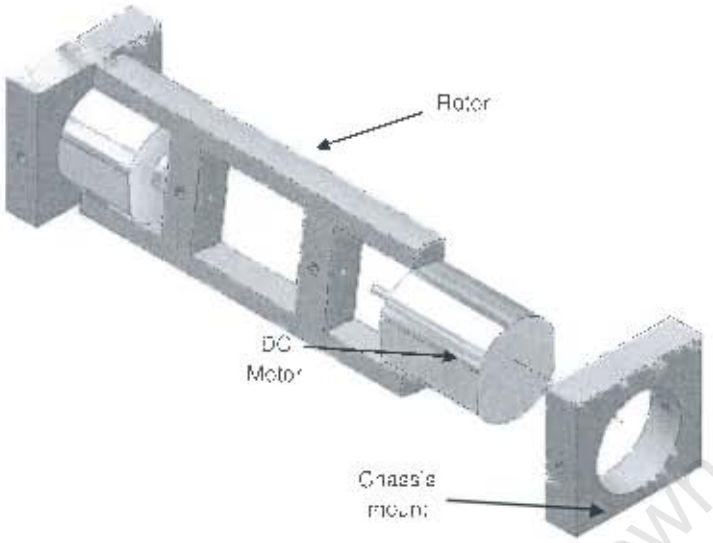


Figure 4.8: Exploded view of the Kicker with chassis mounts
This shows how the different components were assembled to form the Kicker.

4.2.5 Chassis design

As in Prototype One, the use of a solid modelling program greatly aided in the design of the components of the robot since 3D models of the Drive System and Kicker enabled the positions and dimensions to be changed at will to ensure an optimised layout for the robot.

The Chassis mount (Figures 4.5 and 4.6) for the Drive Systems came from a TuteBot [12] design using the same DC geared motors. The aluminium square tube was only slightly larger than would be required and as such did not use significantly more space than allowed.

The Chassis mounts (Figures 4.7 and 5.3) for the DC motors used in the Kicker were also drawn from experience gained from Prototype One. The width of the Chassis mount was made 10 mm while the DC motors were positioned to ensure the rotor was 3 mm above the field when kicking.

The most important consideration in the design of the chassis was to ensure a neutral balance from front to back. In other words, to ensure that the COG was on the same axis as the wheels. The reason for this extra requirement of precision stemmed from the use of skid pads, rather than castors, to balance the robot due to the greater friction inherent in the use of skid pads. If either end of the robot was heavier than the other, the robot would either drag or push along the surface depending on in which direction the robot

University of Cape Town

was moving and whether the front or back was heavier. Despite this issue it was still felt preferable to use the skid pads because of their lighter weight and smaller size. Through keeping a neutral balance the frictional losses were kept to an overall minimum.

When considering the Drive System, Kicker and batteries, the batteries had the greatest mass. Since the drive shaft on the DC geared motor was offset, the bulk of their body was positioned towards the front of the robot (Figure 4.3) to help reduce the effect of the batteries on the COG. The exact position of the motors was found by allowing for a $3mm$ clearance between the base of the robot and the field.

Having a complete Drive System with chassis mount (Figure 4.5), the width of the Kicker could be made as large as possible whilst still remaining within the limits of the robot. Once the final layout was determined, the Base design was finalised.

In order for the Shell to have enough mounting points a Top, secured to the Base using four sets of two $40mm$ stand offs, was added. The central portion of the Top was cut away so as to allow easy access to the inside of the robot.

The final chassis design can be seen in Figure 4.9.

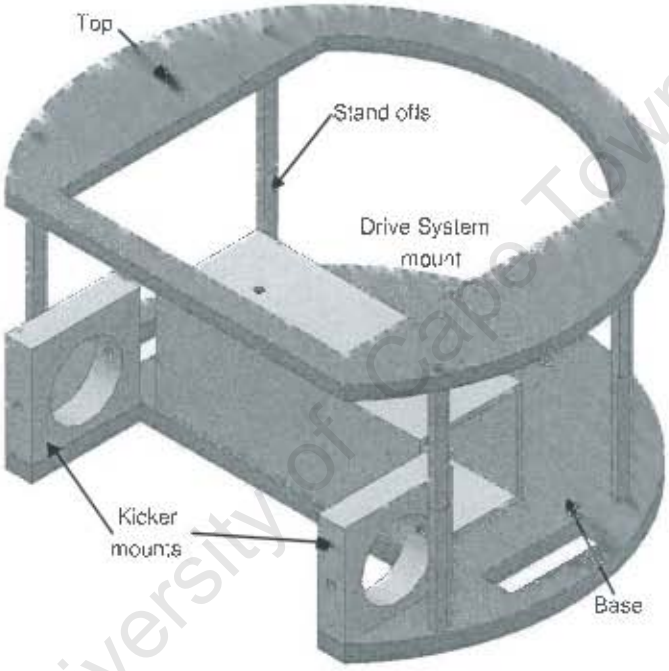
4.2.6 The Wheel Encoder

Although space was left for a H21A1 optical interrupter switch during the design of the wheel and the integrated encoder mask, this was only available on one side of the DC geared motor due to its offset drive shaft. As a result, the one mounting point of the H21A1 optical interrupter switch had to be cut away so it would fit inside the rim of the wheel.

In the prototype, it was felt that being able to adjust the ride height of the robot during testing would have several advantages with respect to both motion and kicking. The required changes in the ride height were achieved by rotating the motors within the chassis mounting. However, if the H21A1 optical interrupter switch was mounted on the chassis, the wheel encoder could possibly either prevent rotation of the wheel or move outside of the encoder mask rendering it useless.

To circumvent this problem, the H21A1 optical interrupter switches were therefore rather bonded to the plastic gear housing on the DC geared motor. In this way, the wheel encoders could work through a range of drive heights.

University of Cape Town



University of Cape Town

Figure 4.9: Robot Chassis
This shows the completed robot chassis without the Drive System or Kicker.

4.3 Evaluation

"Wow! Your robot is really quick!"

This exclamation, uttered by the senior technician in charge of the Control Laboratory where the UCT F180 RoboSoccer team was based upon seeing Prototype Two for the first time, was a welcome indication that the design efforts were finally bearing fruit.

After half an hour of testing, the robot was still going strong in spite of being driven into every conceivable object. The difficulty in driving was a result of the speed of the robot, since the slightest motion on the joystick sent the robot off in another direction. This first informal test proved the stability, reliability and robustness of Prototype Two.

In more serious testing, a top speed of $1.25 ms^{-1}$ and an average acceleration of $\pm 1.73 ms^{-2}$ was recorded, translating to a $0 \rightarrow 1\ ms^{-1}$ time of $\pm 0.3\ s$. This data was captured by the on-board DSP controller. Other than not being omni-directional, these figures surpass those required in the design goals.

Raising the ride height to greater than $3mm$ made the robot less stable whilst lowering the ride height slowed the robot down due to increased friction from a greater contact between the robot and the felt on the field. In experimenting with the ride height, the H21A1 optical interrupter switches finally broke away from the DC geared motors indicating that a more suitable position for these switches would have to be found in the production model.

The last test required was a kicking test. During the process of building the robot, the Kicker remained extremely difficult to set-up. This was due to the DC motors having too much play in the chassis mounts and the fact that the rotor was far from well made. Once the kicker was balanced and tightened as best as possible, the kicker tests were run.

In order to kick the ball the rotor was spun so as to impart top spin on the golf ball with the rotor hitting the ball in an upward motion. It was found that provided the robot was driven into the ball with the rotor spinning, the ball would be shot at pace across the field. On the odd occasion, the ball would only trickle forward slowly depending on how the spinning rotor caught the ball.

However, after a few solid kicks, the rotor would become unbalanced due to the force of the kick pushing the DC motors out of alignment which thus necessitated constant repairs.

Since future revisions of the robot would be used in competition, the outer dimensions were checked to ensure that any mechanical tolerances did not push the robot over the maximum

University of Cape Town

size allowed in F180 RoboSoccer. The diameter, when including all the fasteners for the shell, narrowly remained within the size limits.

4.4 Conclusion

At this point the UCT F180 RoboSoccer team was finally on track with a good robotic platform on which further development could be carried out.

The Drive Systems easily met the performance requirements, with only the H21A1 optical interrupter switches proving fragile when mounted to the DC geared motor. Having confirmed a ride height of 3 mm it was decided that the H21A1 optical interrupter switches could rather be mounted on the chassis and not the DC geared motor.

The rotating Kicker also worked well when it was functioning, although two areas of concern in the design were the rotor and chassis mounts. It was thought that the Kicker would perform much better if, in the production model, the rotor could be re-designed so as to ensure better balance and if the DC motors could be held in alignment during kicking.

University of Cape Town

Chapter 5

Production Model: Design and Evaluation

University of Cape Town

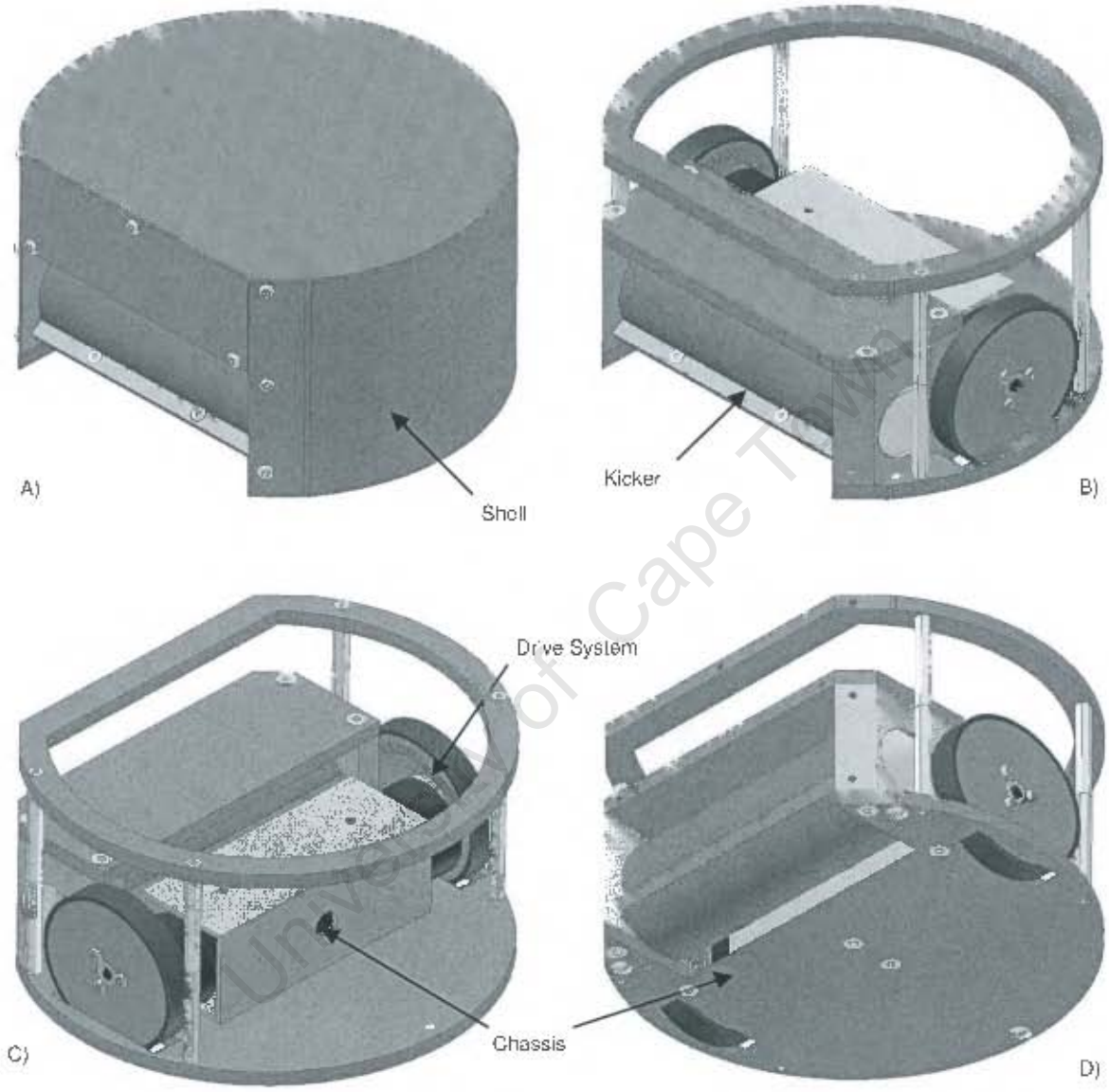
5.1 Design Objectives

Having found a prototype platform that was successful through the design phase, a Production Model was required. Building on the success of the Drive System and layout from Prototype Two, a revised kicker design and adaptations to the Chassis were needed to produce a good working robot.

The areas that needed revision were:

- the Kicker
- a reduction of the outer diameter
- a robust mounting for the H21A1 optical limit switch
- an enhanced access to internal components

Whilst the Kicker needed a whole new design, the rest of the revisions were more subtle as can be seen in Figure 5.1.



University of Cape Town

Figure 5.1: Production Model from various angles

The final design of the Production Model is shown in (A) while (B,C,D) show the robot without the top cover and shell thus exposing the Drive System. Note the changes to the Kicker.

5.2 Design of the Kicker

5.2.1 Required Improvements

The Kicker from Prototype Two had tremendous potential. To meet this potential the Kicker needed to have:

- greater support for the DC motors and
- a better manufactured rotor

whilst still having a wide kicking area.

During the background research one feature that was noted in the Cornell F180 RoboSoccer robots was their ability to dribble the ball [5, 10]. Though having a spinning rod that imparted back-spin onto the ball, the ball literally stuck to the front of their robots whilst they moved, thereby making it extremely difficult for other teams to get the ball.

In order to benefit from this feature, dribbling was added to the requirements of the kicker. By spinning the rotor in the opposite direction to kicking it was believed to be possible to be able to dribble the ball.

5.2.2 The Final Design

As greater support for the DC motors was required, it was decided that a 2.3 mm thick sleeve could extend into the rotor area as part of the chassis mounts. As a result the inner diameter of the rotor had to be 4.6mm bigger in order to compensate for the added support from the sleeve.

Since it was required that the Kicker also function as a dribbling mechanism, the rotor design from Prototype Two was no longer feasible. If the rotor from Prototype Two was spun in reverse, the ball would have been chopped into the playing field thus causing the robot to lift off the ground.

In order to prevent this from occurring, a rounded plate with increasing radius was added to the rotor. The modified Kicker is shown in Figure 5.2. As the rotor spun in reverse the ball experienced a slight forward push and back-spin was imparted on the ball. Thus,

University of Cape Town

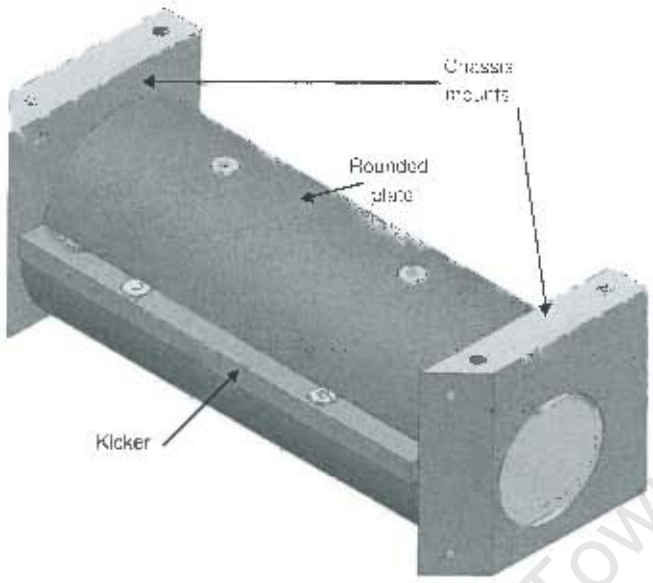


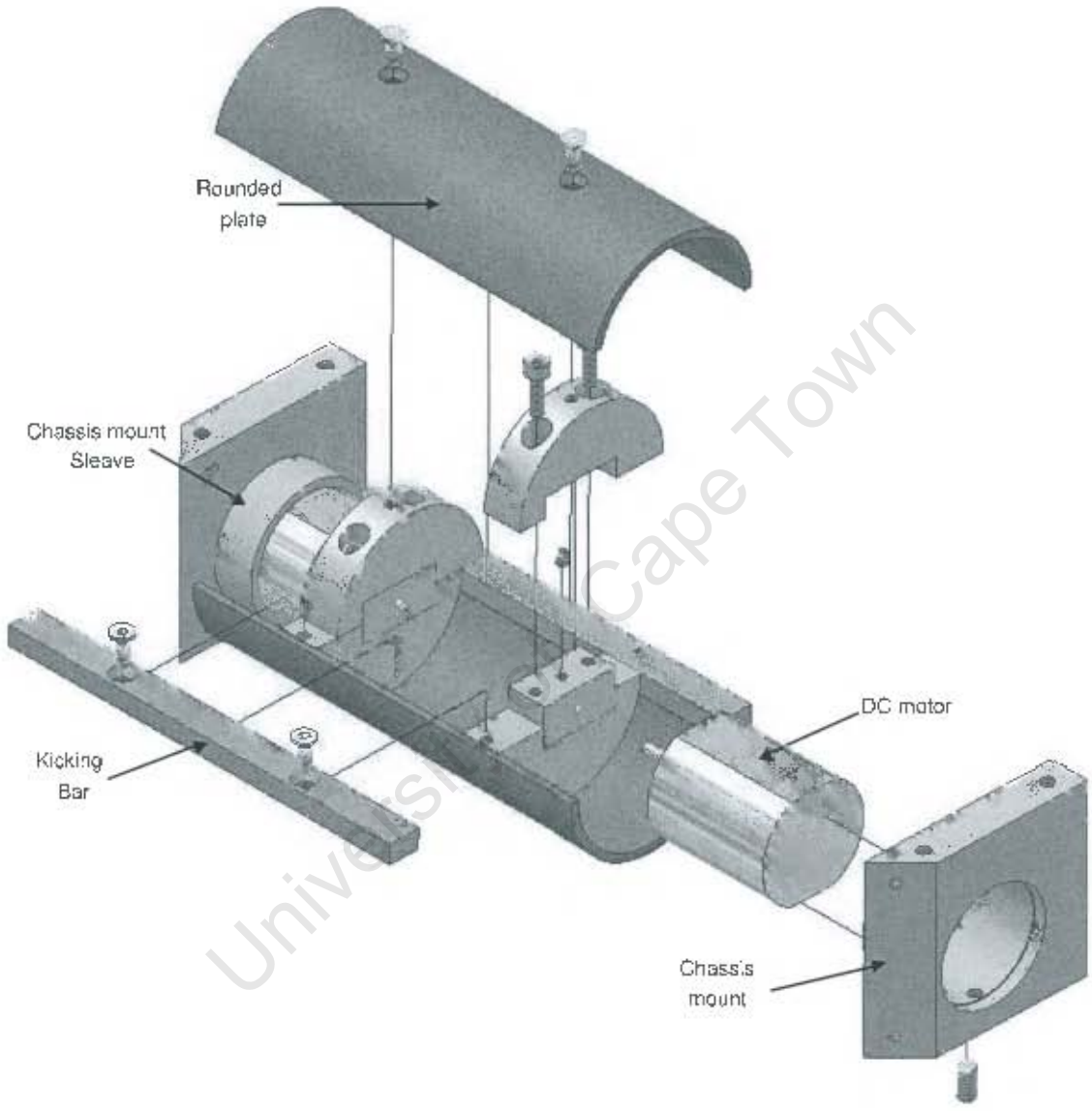
Figure 5.2: Modified Kicker used in the Production Model
This shows the final kicker design with the improved chassis mounts.

University of Cape Town

theoretically, the ball would be "kicked" a few centimetres ahead and the back-spin would in turn spin the ball back into the Kicker.

As in Prototype One and Two, when the rotor is spun in a forward direction, the ball would be kicked with top-spin as the new design had two rounded plates, decreasing in radius, to the point where the kicking plate made contact with the ball.

The rotor for the Production Model was made from Tool Steel as used in Prototype One as the aluminium used in Prototype Two proved too soft at the point where the DC motors made contact. For the unit to be manufactured it needed to be broken up into several components as depicted in Figure 5.3. As the rounded plate was to be made of plastic, round PVC piping was used to eliminate the need to shape plastic sheeting. The $40\,mm$ PVC piping was halved and cut to measure for each kicker. Finally, a sticky water-based bitumen compound was painted onto the outside of the rotor to increase the amount of spin due to frictional forces.



University of Cape Town

Figure 5.3: Exploded view of the Modified Kicker used in the Production Model
This shows how the different components were assembled to form the Kicker.

5.3 Design of the Chassis

5.3.1 Spatial Layout

Before any spatial layouts could be considered, a final diameter for the base of the robot had to be decided. It was determined that making the base diameter $170mm$, would allow for $5\,mm$ of space around the circumference of the robot, which gave ample room for the shell and the fasteners that were required.

Given the larger diameter of the Modified Kicker, the drive assembly was no longer able to remain the same as in Prototype Two. As depicted in Figure 5.4 it became necessary to turn the drive assembly around so that both the Kicker and drive assembly fitted within the smaller chassis limits. Despite this, the robot maintained its neutral balance due to the increased size and mass of the kicker.

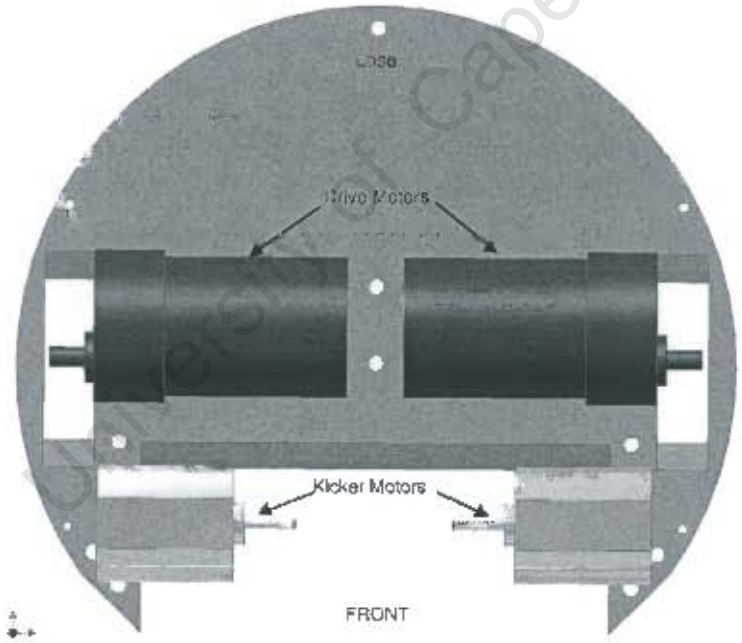


Figure 5.4: Spatial layout for the Production Model
This shows the spatial layout of the drive and kicker motors (which made up the drive and kicker systems) relative to the chassis. Note the changes to the position of the drive motors.

One design innovation which was crucial in enabling the Kicker to remain as wide as

University of Cape Town

possible, was the change in the shape of the chassis mounts. In Prototype Two, the chassis mounts holding the Kicker had a square face (Figure 4.9), while in the Production Model the chassis mounts were tapered which allowed them to be moved further outwards, whilst still remaining within the base diameter of the robots. The tapered chassis mounts had an added advantage in that a flat surface became available onto which the shell could be directly fastened.

5.3.2 The Final Design

Having determined a layout for the Drive Systems and Kicker which met the necessary requirements, the dimensions for the chassis were finalised. As the Kicker was larger than before, the amount of material between the cut-out for the rotor and the wheels had become less. To overcome this new problem, the cut-outs for the rotor and wheels were shaped to match their profiles, thus keeping a minimal clearance whilst increasing the amount of material in order to strengthen the chassis. To further stiffen the chassis, a plastic plate was added across the top of the Kicker which was, in turn, fastened onto the chassis mounts used by the Kicker. The completed chassis design with all the modifications is shown in Figure 5.5.

The last requirement for the chassis design was the need to enhance the access to the inside of the robot. In meeting this requirement, more of the Top of the chassis was cut-away than had been in Prototype Two (Figure 4.2.5), thus leaving only a thin rim around the perimeter of the robot. This final modification is shown Figure 5.5.

5.4 Design of Sensor mounting

The final design requirement for the Production Model was the need to find a acceptable way in which to mount the H21A1 optical limit switches onto the chassis. Upon placing an H21A1 optical limit switch in position it became apparent that a small plate, fastened to the chassis mount of the Drive Systems, would be at the correct height to hold the switch in place. The switch was then fastened through the use of its own mounting hole to a hole drilled into the mounting plate with a $3mm$ nut and bolt. The chassis mount for the Drive Systems, the mounting plate and the H21A1 optical limit switch is depicted in Figure 5.6.

University of Cape Town

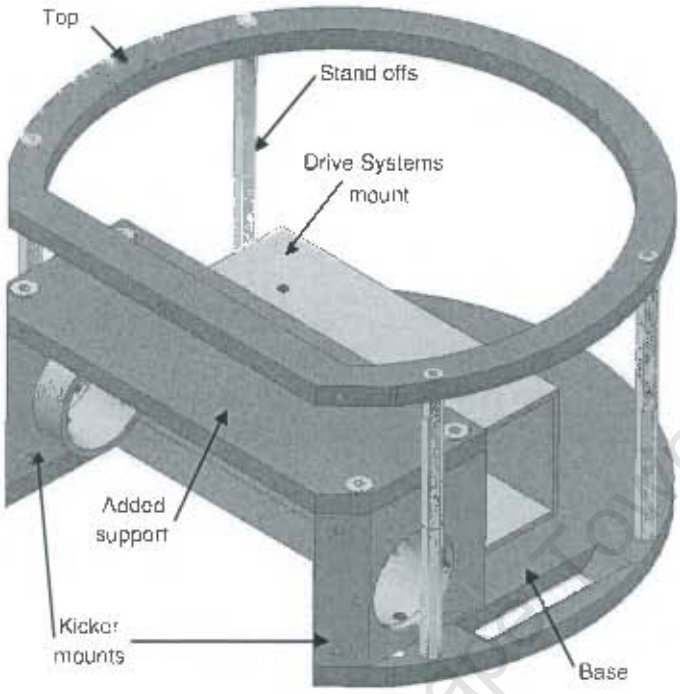


Figure 5.5: Production Model chassis
This shows the completed robot chassis without the Drive System or Kicker.

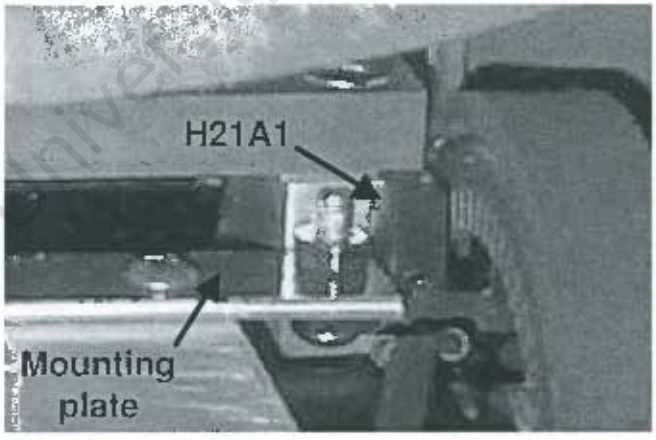


Figure 5.6: Photograph of wheel encoder
This shows the H21A1 optical limit switch aligned with the encoder mask (which was built into the wheel) being held in place by a plastic mounting plate.

University of Cape Town

5.5 Mechanical Evaluation

5.5.1 Construction of the Production Model

The one area of concern during the design of the Production Model was that there had been no Prototype Two revision. Instead, the parts for all five robots were machined as a single job. This was necessitated by the fact that, at that stage, there was insufficient time to have the parts for one robot machined for testing before the parts for the other four robots were made.

Thus, during the assembly, a few minor adjustments were needed in order to produce a finished robot. One such adjustment arose since, depending on how the Kicker was assembled, the rotor would not spin because it made contact with the chassis. It was found that this problem could be simply solved by turning the kicker bar (Figure 5.3) around.

Overall, the assembly of the five robots was both simple and expedient whilst minimising the chance of either damaging the components or assembling the robots incorrectly.

5.5.2 Motion

In general, the Production robots performed very similarly to Prototype Two as described on Section 4.3.

The only remaining area of concern was that each robot behaved slightly differently to the others. For example, some of the robots would tend to drift to the right while others would tend to drift to the left during both steady state velocity and acceleration tests.

5.5.3 Kicking

Apart from the previously mentioned problem with the kicker bar, both the DC motors and rotor were naturally correctly centred thus the rotor spun freely without the need for any adjustments as had been the case with Prototype Two.

In addition, it was found that with the bigger rotor the added inertia actually helped in kicking the ball. The resultant kicks were markedly more powerful and, in spite of the

University of Cape Town

greater inertia, the rotor took no longer to reach kicking speed. This was most likely due to the better balance and alignment achieved in this Kicker design.

As was found in Prototype Two, there remained occasions when the ball would only somewhat dribble forward upon kicking. As before, these occasions were limited by having the robot run into the ball so that a firm kick was made.

To date, the Kickers on the robots have been through the initial testing, undergraduate thesis open day demonstrations, further testing as well as the actual F180 RoboSoccer 2003 competition without needing any adjustments - thus demonstrating the robustness of the design.

5.5.1 Dribbling

The idea of adding dribbling to the function of the kicker had always seemed improbable but was deemed worth attempting.

In testing, when the rotor was spun in reverse at the same speed as when kicking, the ball would be pushed too far ahead for the back-spin to draw the ball back onto the Kicker. Reducing the speed of the rotor, however, meant the ball was only pushed a small distance ahead yet still had sufficient back-spin to draw the ball back.

In this way the robot was able to dribble the ball with moderate success in a straight line but, due to the ball still being pushed a little too far ahead, the robot was unable to maintain possession of the ball while turning.

Overall, the dribbling feature of the Kicker was not really sufficiently evolved to make an impact on the performance of the robot during match play. If further revisions of this type of dual-function kicker are made at some point in the future, the amount of change in the outer diameter of the rotor, relative to the ball, whilst rotating should be reduced. This would have the effect of pushing the ball even shorter distance ahead of the robot whilst providing the same amount of back-spin as before.

5.6 Conclusion

Even though this Production Model was not quite perfect when one takes into consideration the variations in motion, the odd miss-kick and the variable performance of the Kicker

University of Cape Town

whilst trying to dribble, it did provide an excellent robotic platform for the UCT F180 RoboSoccer team.

Overall, since the mechanical aspects of this current design allows the robots to perform in a reliable manner, they may adequately serve as a platform for future development in other projects.

University of Cape Town

Chapter 6

Electrical Design

University of Cape Town

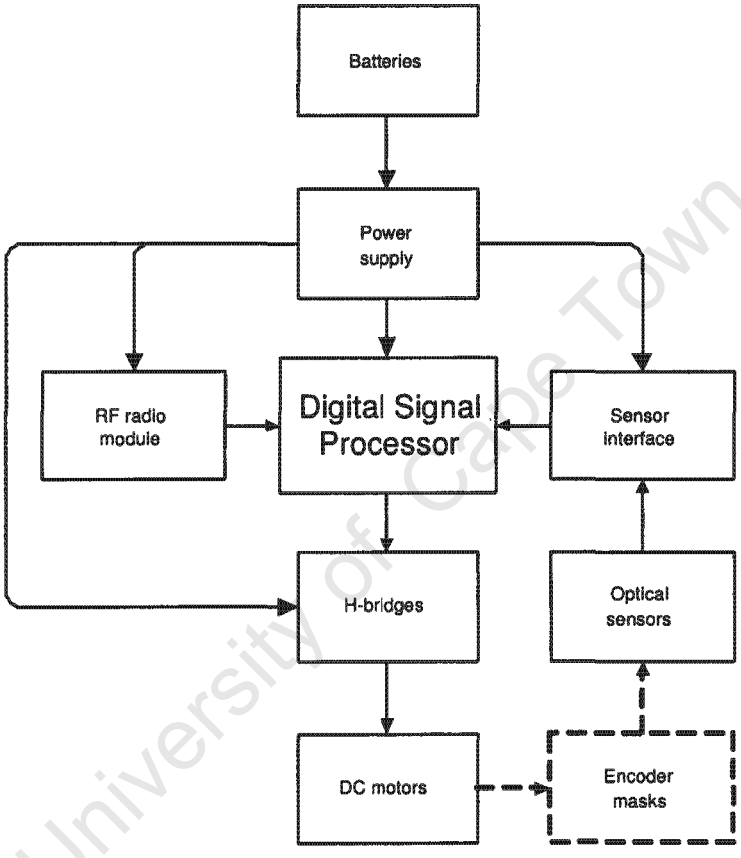
6.1 Introduction

From the beginning of the UCT F180 RoboSoccer campaign, there was an existing solid platform of prior experience on which to build. The key factor in the success of the Electrical Design stemmed from the TI TMS320F243PGEA Digital Signal Processor (DSP) Controller. Since the DSP controller was able to handle all the requirements of the robot designs, it was a case of having to interface the power supply, the power electronics (H-bridges) for the Drive Systems, the sensors and the communication module so as to produce a completed system (Figure 6.1).

Accordingly, the DSP controller board was the first of the electronics to be designed and implemented (Section 6.3) straight to PCB (Figure 6.2). As each prototype robot was produced, the electronics was first designed on breadboard and then implemented on veroboard. Once the Production Model was designed, the relevant circuits were combined into a single board and a PCB was designed, thus producing the final Interface board (Figure 6.3). Figure 6.4 highlights the final integration of the peripherals onto the Interface board.

6.2 Overall Requirements of the Electrical System

Since the requirements followed directly from the mechanical design and, in particular, the choice of the electro-mechanical components, these changed from Prototype One to



University of Cape Town

Figure 6.1: An overview of the Electronics
This outlines the overall structure of the electronics used for each robot. The encoder mask was part of the mechanical system, but is included for completeness.

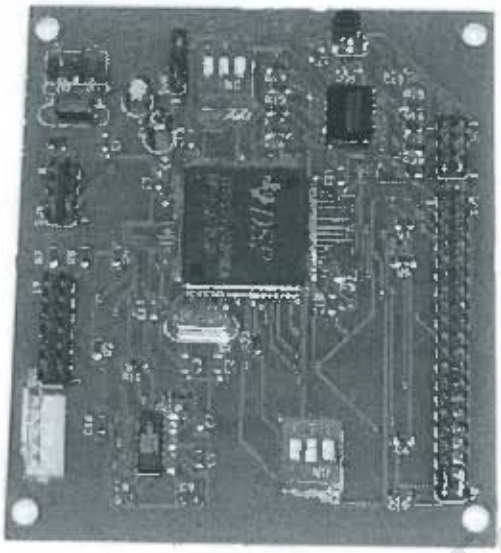


Figure 6.2: The DSP controller

This shows the on-board controller used by the robots to execute commands from the AI system.

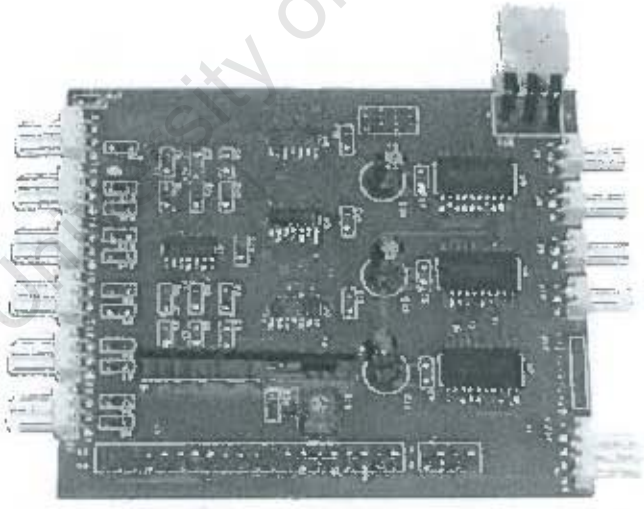


Figure 6.3: The Interface board

This shows the final interface board as implemented for the 2003 competition. This board plugged directly onto the DSP controller and provided connectors for the sensors, DC motors and RF radio

University of Cape Town

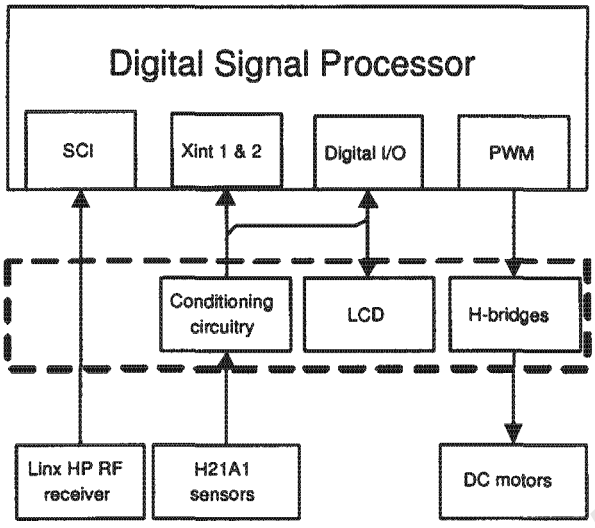


Figure 6.4: An overview of the DSP controller and the Interface board
This shows how the different peripherals were combined to form the Interface board (dashed block) as well as their connections to the DSP controller.

Prototype Two. However, as the Production Model closely matched Prototype Two, no further changes were made to the requirements at that stage.

For Prototype One, the electrical requirements were

- 16 V for the DC geared motor (350 mA)
- 7.2 $V \rightarrow$ 8.4 V for the two DC drive motors (11.6 A @ 7.2 V)
- 9 V for the DC kicker motor (600 mA)
- 5 V for the electronics ($\pm$250 mA) and DSP controller (250 mA)
- to provide support four H21A1 optical interrupter switches
- to provide RF serial communication

whilst for Prototype Two and the Production Model these became

- 16 V for the two DC geared motor (700 mA)
- 9 V for the two DC kicker motor (1200 mA)

University of Cape Town

- 5 V for the DSP controller (250 mA) and other electronics ($\pm$250 mA)
- to provide support two H21A1 optical interrupter switches
- to provide RF serial communication

Each of the above requirements could be met through the use of the following: H-Bridges, a linear regulator, interface board and radio modules. The implementation of each of these is discussed in the following sections.

6.3 DSP Controller

As mentioned before, the DSP controller formed the platform around which all the other electronics was built. As such, the DSP controller had to be capable of meeting all the control requirements of the electronics.

6.3.1 Requirements

As the controller would handle all the functions of the robot, it was necessary for the controller to have at least:

- a clock speed of 10 MHz
- four Pulse Width Modulation (PWM) channels
- two timer modules
- four external interrupts
- a serial communications interface (SCI)

It is important to note that these minimum requirements for Prototype One exceeded those of Prototype Two and, as such, were used unchanged for the later designs.

University of Cape Town

6.3.2 Controller Selection

The selection of the controller was essentially one of convenience since the researcher had previously used the Texas Instruments (TI) TI320F243PGEA Digital Signal Processor [6]. Furthermore, this DSP was more than capable of meeting the controller requirements and was therefore used in the design of the DSP controller.

6.3.3 Design of Schematics

The design of the schematics was based on a reference design from Spectrum Digital [13]. To simplify the overall design of the DSP controller, the external memory and its support circuitry was excluded as the on-board memory was sufficient. The reset circuitry was also modified so as to provide an on-board reset button. Furthermore, in order to add extra functionality to the controller, a TI TLC5628DW [14] Digital to Analog Converter (DAC) was added. The DAC also provided a means of debugging software as values could be written to the output of the DAC and read using an oscilloscope. Finally, switches were added to the RS232 SCI and Serial Peripheral Interface (SPI) to allow these to be enabled or disabled as required.

University of Cape Town

The Reset circuit

The choice of reset circuit design was determined by the availability of the MAX6816EUS-T [15] switch de-bouncer in the form of samples from MAXIM. The MAX6816 only required the addition of a reset button and a capacitor and also had an extremely small footprint which aided in the design of a compact PCB. The schematic is shown in Figure 6.5. The output RS^{-}connected directly to the RS^{-} (Pin 19) on the DSP.

The DAC circuit

The choice of the TLC5628DW was due to its SPI interface which enabled a direct connection with the DSP's SPI interface. The $DATA$, CLK and $LOAD$ were connected to $SPISIMO$ (Pin 60), $SPICLK$ (Pin 64) and $SPISTE^{-}$ (Pin 66) via a 3-way dip switch. The switches enabled the DAC to be disconnected in order for the SPI to be used to connect a different peripheral. The schematic in Figure 6.6 shows the relevant connections and the extra header.

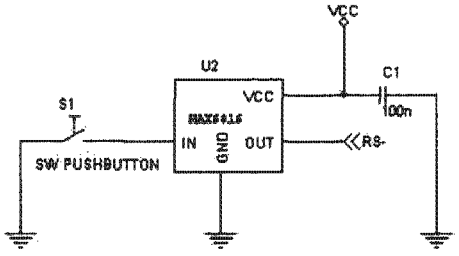


Figure 6.5: Schematic of the reset circuit implementation
This shows the implementation of the MAX6816 switch de-bouncer used for the reset circuit on the DSP controller.

University of Cape Town

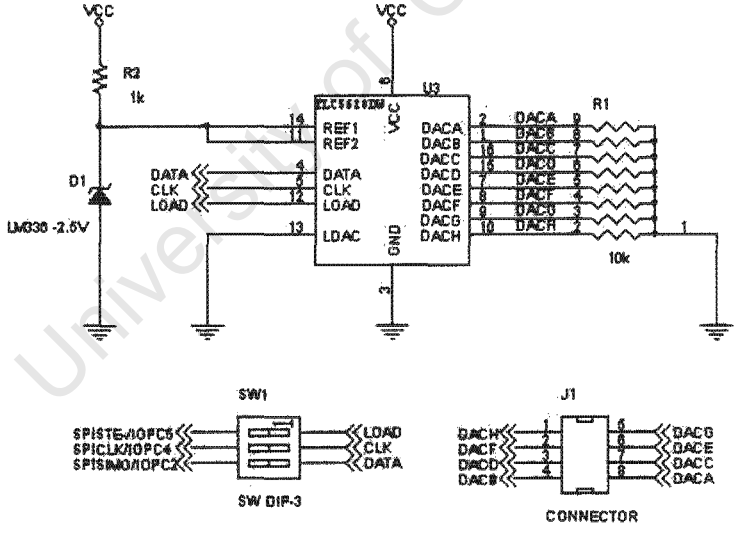


Figure 6.6: Schematic of the DAC circuit implementation
This shows the implementation of the TLC5628 DAC used on the DSP controller. The DAC was mostly used for debugging software.

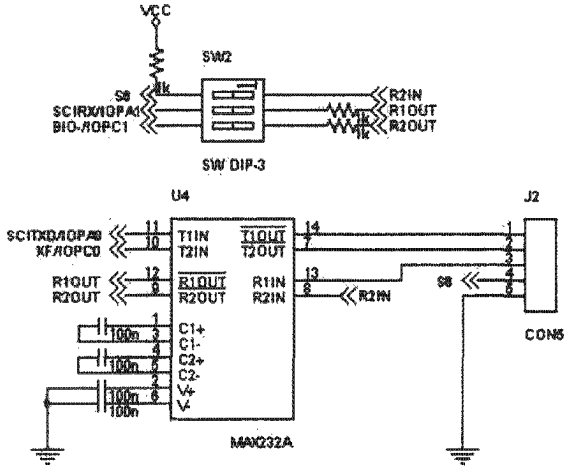


Figure 6.7: Schematic of RS232 circuit

This shows the implementation of the serial communications circuitry required for serial programming of the DSP.

The RS232 circuit

At the start of the project, it was decided that digital RF radio modules would be used for the serial communication. Although the choice of RF modules was not yet finalised, the ability to disconnect the RS232 on the controller board would enable a direct TTL compatible connection. The switch implemented was a 3-way dip switch where one was used for serial programming of the DSP (dip switch 3) whilst the other two allowed the outputs from the RS232 (*R1OUT and R2OUT*) to be disconnected from the DSP (*SCIRXD/IOPA1 and BIO$^-$/IOPC1*) as shown in Figure 6.7.

6.3.4 Layout of the DSP Controller

In designing the layout, several factors needed to be considered in order to optimise the DSP controller. The design needed to:

- minimise the size of the board
- allow for only one side to be populated
- only use two layers

University of Cape Town

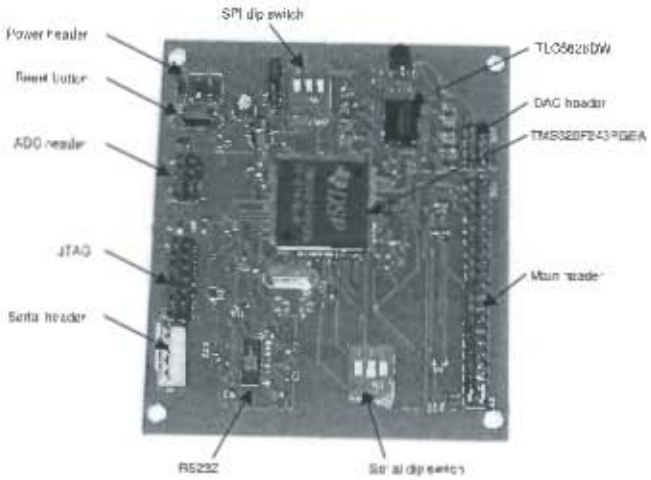


Figure 6.8: DSP controller layout
This shows the layout of the important components and headers found on the DSP controller.

- place the headers along two sides of the board
- ensure the initial design worked as required.

As the reference design from Spectrum Digital [13] was 62.5 mm by 100 mm, this was used as a starting point for the layout. As the design of the DSP had similar pin functions largely grouped together, the associated peripherals and headers could be located around the DSP to minimise the routing. Once the components were positioned, the actual board size was 76.0 mm by 90.0 mm. As the board was quite small, it was felt that trying to make the board smaller would only disproportionately increase the chance of a design error in relation to achieving only a small reduction in size. The final layout, as implemented, is shown in Figure 6.8.

6.4 H-Bridge Board

Once the 7.2 V DC motors were chosen for the first prototype, the next requirement was to determine a way to drive these motors. The high current requirement meant that no single semi-conductor package would suffice and, as such, a dedicated H-Bridge Board was implemented and is shown in Figure 6.9.

University of Cape Town

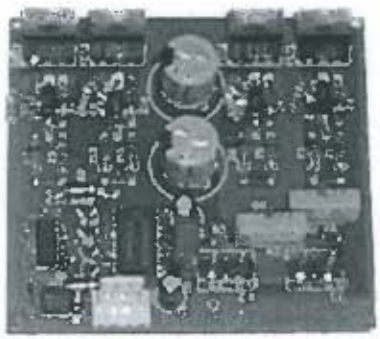


Figure 6.9: H-Bridge Board
This shows the H-bridge board used to control the 7.2 V drive motors used in Prototype One.

6.4.1 Requirements

As the motors drew 11.6 A @ 7.2 V the power requirements were already known but, in order for the H-Bridge to be used for F180 RoboSoccer, additional requirements were set as follows:

- the size of the board had to be kept as small as possible
- the bridges had to be able to handle a 20 KHz switching frequency
- the switching and on losses had to be kept to a minimum
- the board had to either have a full bridge or two half bridges

6.4.2 Component selection

The general components required for a full bridge include transistors or mosfets (for this voltage and current rating) and drivers. As driving an inductive load like a motor can cause problems with shoot through on full bridge designs, it is normal to also include snubbers to protect those devices.

Since mosfets provide a far greater frequency response and are available with very low on resistance, these were chosen over transistors. Driving the mosfets would then require either a full or half bridge driver, or individual mosfet drivers. On consultation with the

University of Cape Town

Senior Technician from the Power Group at UCT, it was decided to use the HIP4081A high frequency full bridge fet driver [16] in conjunction with SUB75N03-04 mosfets [17].

The combination of the HIP4081A and the SUB75's provided excellent performance that was more than capable of meeting the previously mentioned requirements.

6.4.3 Design of Schematics

The datasheet for the HIP4081A, along with a power-up application note [18], provided a basic framework for the circuit design. As no snubber or any other protection circuitry was shown in these documents, further research found a full bridge design which used the HIP4081A and mosfets in its design [19]. Based on the Open Source Motor Controller (OSMC) [19] a simplified full bridge was designed using four SUB75's and the HIP4081A.

The HIP4081 has the ability to control the two half bridges independently and, since the DSP controller was capable of doing this, logic circuitry was added to enable the H-Bridge to be driven from a single PWM signal or from two PWM signals. In doing this the flexibility of the H-Bridge board was increased and, if needed, the DSP controller could include dead band to assist in preventing shoot through. The modified logic circuitry is illustrated in Figure 6.10.

6.4.4 Layout of H-Bridge Board

Since Prototype One used two DC motors from RC cars, an aluminium heat-sink had already been built into the chassis of the robot. Accordingly, it was planned to use the space between these two motors for the H-Bridge board and this meant that both the area and height was of utmost importance.

For a single heat sink to be used, not only by all four mosfets on each board but rather by both boards, meant that they needed to be placed along one side of the board. As the DC motors flanked both sides of where the boards were to be placed the signal- , DC power- and DC motor connectors had to be placed opposite the mosfets.

As each board would be expected to deliver about 11 A, two final considerations were regarding the track width and that the track length between the gates of the mosfets and the HIP4081A had to be kept to a minimum.

University of Cape Town

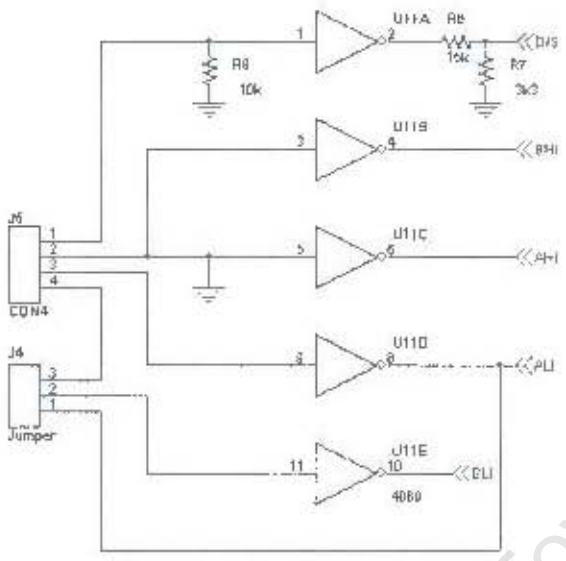


Figure 6.10: H-Bridge logic circuitry
This shows the control circuitry used to ensure both the safe start-up and the supply of the high and low side signals to the HIP4081A.

Once the board layout was completed it was found that, given a double-sided PCB, the board could be made 61 *mm* long by 66 *mm* wide which met the space requirements. A four layer board was not considered due to the large number of through hole components and cost of production.

The H-Bridge board is shown in Figure 6.11.

6.5 Interface Board

Like the mechanical design of the robot, the interface board went through several revisions. Throughout all the revisions, the requirements remained similar in that the H21A1 optical interrupter switches needed to be interfaced with the DSP and the PWM signal from the DSP needed to be interfaced with H-bridge chips to control the 16 V DC geared motor/s and 9 V DC motor/s.

The first genuine interface board was only built for Prototype Two (Figure 6.12) since Prototype One had merely a collection of circuits which had been wired together so as to perform the same function. The main advantage of a single interface board however, was

University of Cape Town

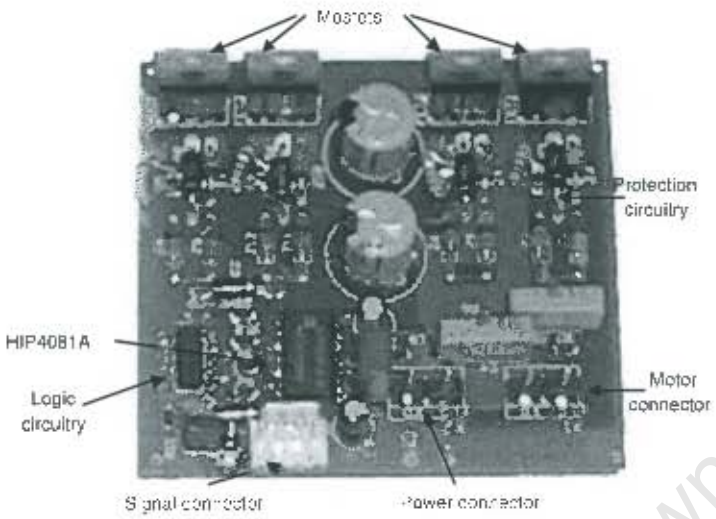


Figure 6.11: The Layout of the H-Bridge board
This shows the layout of the different components and connectors on the H-Bridge board.

that it plugged directly onto the DSP controller board and thus required no wires or leads whilst the sensors and motors connected directly.

Given the success of the first interface board, another interface board was designed and implemented on PCB (Figure 6.13). This second interface board served to combine the LCD panel onto the board as well, and thus gave a complete solution for the robot development. The interface board complete with LCD panel attached is depicted in Figure 6.14.

6.5.1 Requirements

Through the evolution of the interface board, each new design was made such that it would be backward compatible with the previous designs. This meant that, at any stage, the latest interface board could be used for any of the robot designs. As such, the requirements for the final implementation of the interface board were greater than those required by the actual Production Model robot yet still remained sufficient to control the Pluto mobile based robot (Prototype One).

These final interface requirements were:

- conditioning circuitry for six H21A1 optical interrupter switches

University of Cape Town

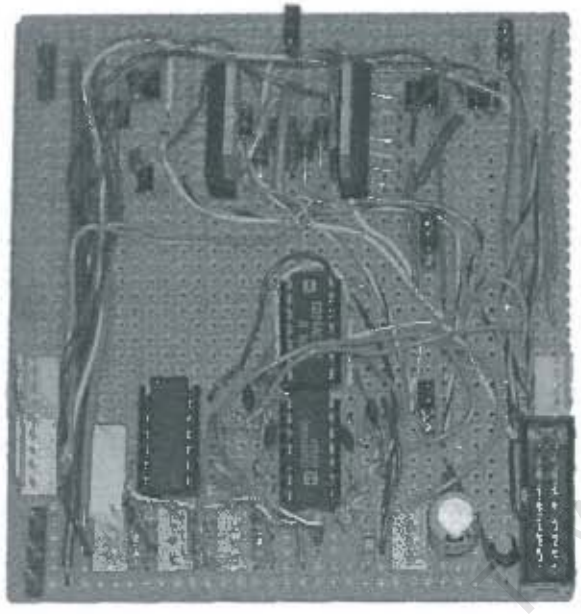


Figure 6.12: The interface board for Prototype Two
This shows the first interface board design (laid out on veroboard) which plugged directly onto the DSP controller.

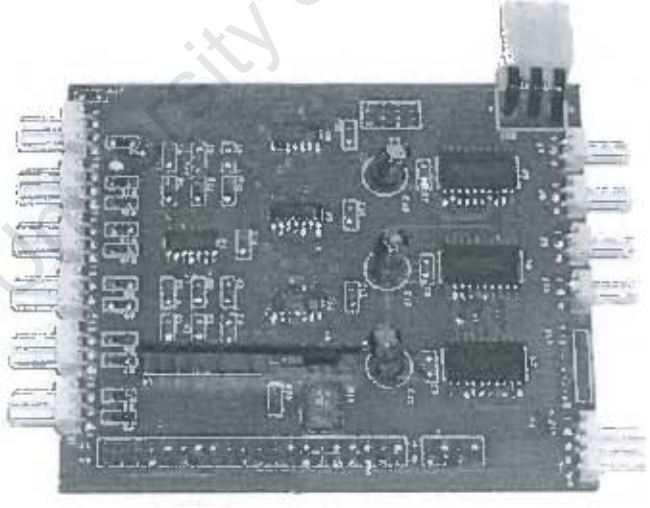


Figure 6.13: The interface board for the Production Model
This shows the final interface board as implemented for the 2003 competition. This board plugged directly onto the DSP controller.

University of Cape Town

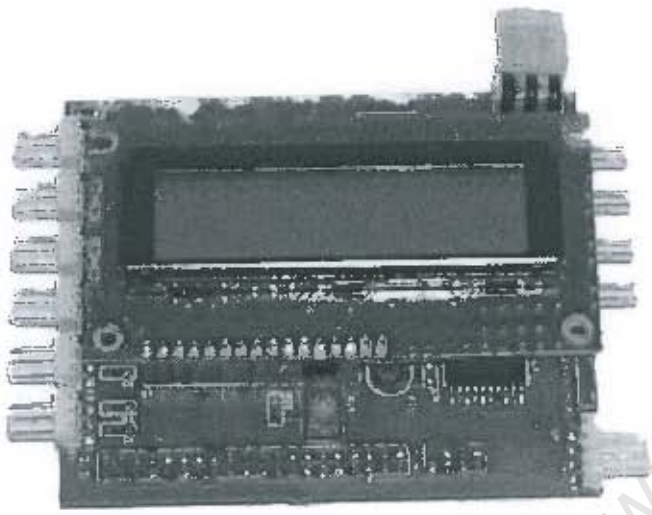


Figure 6.14: The interface board with LCD panel attached
This shows the final interface board with the intelligent LCD panel (which was used for displaying the values of internal registers) attached.

- one Xint trigger per three H21A1 optical interrupter switches (two in total)
- three integrated H-bridge circuits
- connector for the RF receiver module
- the relevant connectors for the sensors and motors

6.5.2 Component selection

The interface board, in essence, consisted of two parts; the circuitry for the sensors and the H-bridges.

There were two options for conditioning the sensor inputs as either a comparator circuit or Schmitt trigger logic gate could be used. The latter option proved the simplest since, by using a single 74HC14 Hex inverting schmitt trigger [20], a TTL logic signal was directly obtained. The conditioning circuitry was completed with the addition of an 74HC86 exclusive OR gate [21] and a 74HC32 two input or gate [22].

The circuitry for the H-bridges thus came full circle. Initially, on Prototype One, TPIC0107B PWM control intelligent H-bridges were used for the direction and kicker motors whilst

University of Cape Town

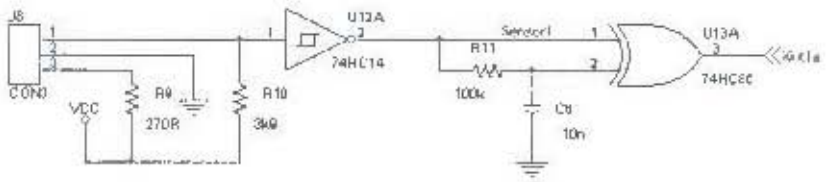


Figure 6.15: Conditioning circuitry for the H21A1 optical sensors
This shows the 74HC14 which digitised the signal from the H21A1 optical sensors and the 74HC86 (with a RC circuit) which served as an edge trigger for the digitised signal.

on Prototype Two L298 Dual full-bridge drivers were used for the drive system and kicker motors and then the TPIC0107B's were again finally used for the final interface design.

The reasons for this were simple since the TPIC0107B required no external circuitry to function and, being a surface mount package, minimal board space was used. The only disadvantage (when compared to the L298) was the lower switching frequency of 4 KHz which meant that the PWM frequency was audible.

6.5.3 Design of Schematics

In order to enable the interface board to handle the sensors, resistors were included to power the infra-red diode and to serve as part of the photo-transistor circuit. The output from the sensor was then connected to a 74HC14 which turned the analog signal into a TTL logic signal. This is illustrated in Figure 6.15.

The one limitation of the TI320F243PGEA DSP was its limited number of external interrupts. As it had only two interrupts, the DSP needed external circuitry to handle up to six sensors and to accomplish this task, an edge detector circuit was implemented using the 74HC86 as shown in Figure 6.15.

In the circuit in Figure 6.15 there are two outputs; an interrupt signal, which was connected to a 74HC32 with the other interrupt signals to form the Xint signal for the DSP, and the Sensor signal which the DSP could poll to check which sensor caused the interrupt.

The design of the schematic for the TPIC0107B followed directly from the datasheet [23] and was implemented as shown therein.

University of Cape Town

6.5.4 Layout of Interface Board

Since the interface board had to plug into the existing DSP controller, the positioning of the connectors needed to interface the two boards was pre-determined. The layout for the remainder of the board was kept as simple as possible and therefore the sensor circuits (with their respective connectors) were placed on one end whilst the H-bridges were placed on the other end (with their respective connectors).

As space along the sensor side was taken up by the sensor connectors, the connector for the RF receiver module was placed on the same side as the connectors for the motors. As all the IO pins on the DSP controller were so far unused, an extra connector was added to enable these IP pins to be used by other peripherals if required at a later stage.

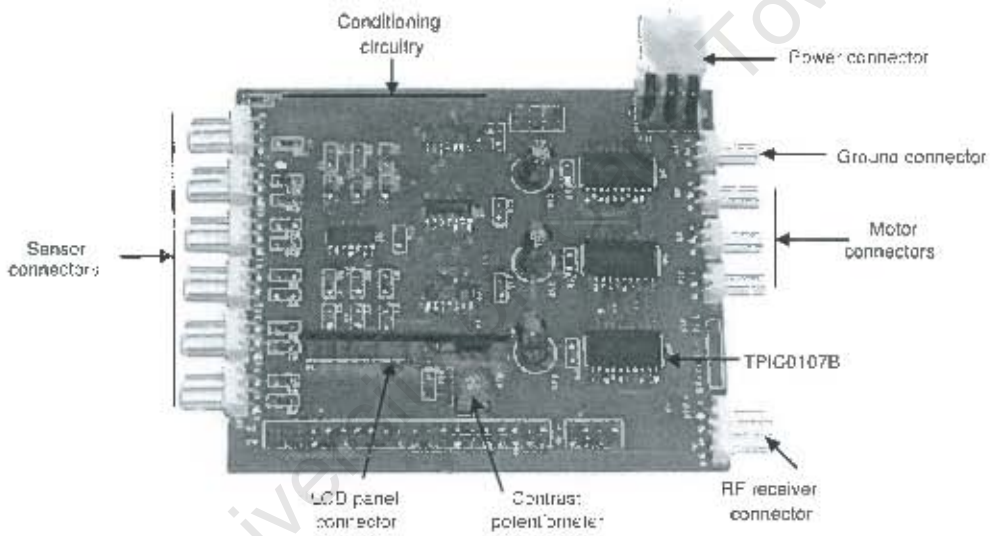


Figure 6.16: The layout of the interface board for the Production Model
This shows the layout of the important components and headers found on the final interface board used in the 2003 competition.

6.6 RF Radio Modules

The choice of the Linx HP RF modules was made as there was already limited experience within the F180 RoboSoccer research group with these radios. The modules are designed for high performance analog or digital data transfer in the $902-928\,MHz$ frequency band,

they have eight selectable channels and are capable of transmission up to a range of 1000 feet under ideal conditions.

The modules come in two forms, (receiver and transmitter modules), thus both points must have a receiver and transmitter module to enable duplex communication. However, as only simplex data transmission was required (the vision system providing feedback control), the computer required only a transmitter module and the robots only receiver modules.

The Linx HP RF receiver module, as used by the robots, is shown in Figure 6.17. The circuitry was insulated so as to prevent shorting on the other components within the robot. The complete transmitter unit which housed the Linx HP RF transmitter module is shown in Figure 6.18 along with the power supply and serial connection for the computer.

University of Cape Town

Figure 6.17: The RF receiver
This shows the Linx HP RF receiver module implemented on veroboard and insulated to provide protection. The connector lead is also shown.

6.6.1 Implementation of the Linx HP RF Modules

In order to implement the Linx modules these were mounted on veroboard with the extra components as required in the design guides [24, 25]. The only difference between the

Figure 6.18: The RF transmitter
This shows the RF transmitter unit (which housed the Linx HP RF transmitter module), power supply and serial cable.

University of Cape Town

receiver and transmitter modules was that the transmitter required both an RS232 circuit and a $5\,V$ power supply whilst the receiver interfaced directly with the interface board.

For the 2002 competition in Pretoria antennae were mounted directly to the antenna pins on both modules. The antennae were roughly designed as outlined in the design guides but, because of the lack of a ground plane and since the antennae were simply mounted directly to the pins, the radios proved unreliable for data transmission. A further hardware limitation which is discussed in Section 8.2.2 was also crucial to the failure of the serial communication between the computer and robots.

On further research, an application note from Linx [26] helped guide the design of a new antenna for both the transmitter and receiver modules. Following the re-design of the antennae and the other software changes that are discussed in Section 8.2.2, the Linx HP RF modules worked without any further problems.

As both the modules had eight channels, a dip switch was implemented in order to be able to select which frequency band the transmitters and receivers would use. This was done so that a separate transmitter could be used to control other robots on another channel. In this way, it was possible for human controlled robots to compete against AI controlled

robots using the same modules, but on different channels (frequency bands).

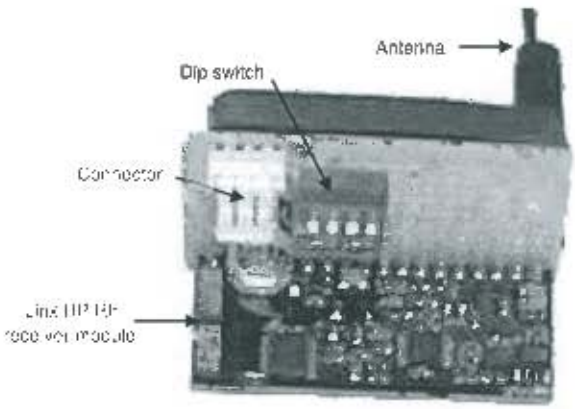


Figure 6.19: The RF receiver
This shows a close-up view of the Linx HP RF receiver module, serial connector and dip switch (for channel selection).

University of Cape Town

6.7 Linear Regulator

The power circuitry for the robot was initially set aside for an undergraduate thesis project however, unlike the vision and AI systems, the project was not chosen by any of the undergraduates. Consequently, as there were sufficient batteries and therefore ample power to meet the robots' power requirements, a simple linear regulator was implemented to provide the $5\,V$ required by the electronics.

The other voltages required by the different motors were dealt with by limiting their PWM duty cycle (δ) to produce average voltages which were within their operating limits from the unregulated battery voltage.

6.7.1 Final Implementation

The linear regulator chosen was the LM7805 [27] which was implemented as shown in the datasheet. Fuses were added in series with the batteries to ensure the safety of the electronics. The first linear regulator board that was made is shown in Figure 6.21 whilst the final implementation is shown in Figure 6.22.

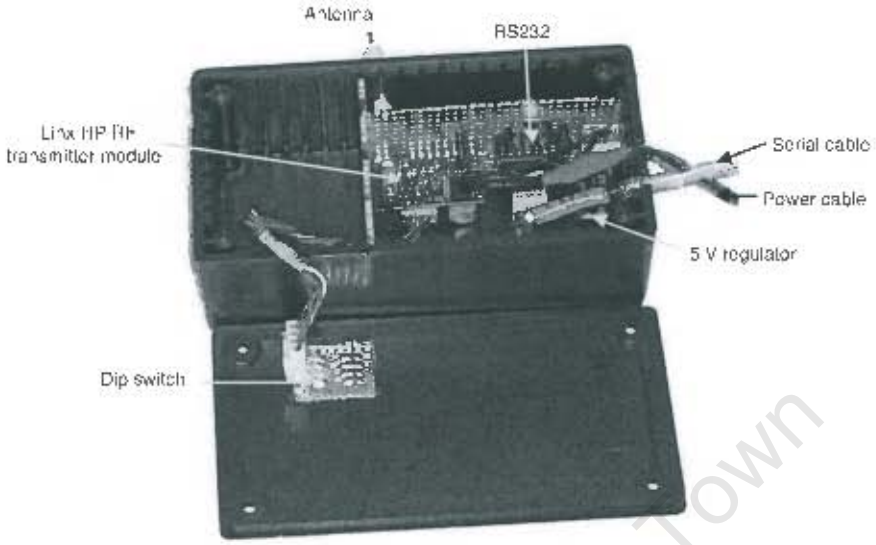


Figure 6.20: The RF transmitter
This shows the internal components of the RF transmitter unit.

Figure 6.21: The linear regulator for Prototype One and Two
This shows the initial regulator board used by the two prototypes. The board had fuses for protection and a switch which controlled both the voltage buses.

University of Cape Town

Figure 6.22: The linear regulator for the Production Model
This shows the final implementation of the regulator board as used in the 2003 competition robots. The board retained its fuses, but the switch was moved to the top of the robot.

6.8 Conclusion

The choice of the TI TMS320F243PGEA DSP for the controller board created an opportunity development both within the scope of this thesis and for future work. As the Interface board was also designed to be used for future development, this will allow the electronics to be used in future robots without any further expense or design being required. During the course of testing and the competition the electronics functioned reliably and, as such proved successful.

University of Cape Town

Chapter 7

Implementation of the software for the DSP controller

University of Cape Town

7.1 Introduction

Once the Texas Instruments TI320F243PGEA DSP had been chosen (Section 6.3.2) to perform as the controller for the UCT F180 RoboSoccer robots, the implementation of the software to control the robots was required. Since the DSP had already been used in other areas of research by this researcher [6], there was already an existing body of code which could be used as a starting point. In most instances where existing code was used in this project, revisions were made in order for the code to be streamlined and robust.

7.2 Operation of the DSP

7.2.1 Modes of operation

In using the DSP, a few initial decisions needed to be made in terms of how the DSP would operate. The DSP concerned has two hardware modes and two software modes of operation.

The two hardware modes of operation were:

- as a microprocessor using external memory

- as a micro-controller using the on-chip memory.

Upon consideration, it was decided to use the DSP in micro-controller mode since, firstly, the DSP performs faster when used as a micro-controller[1] and, secondly, the software memory requirements to control the robot would be within the limits of the on-chip memory.

The two software modes of operation were either:

- non-interrupt driven
- interrupt driven.

Both software modes run a continuous loop, but interrupt driven software has the advantage of being able to leave the continuous loop to perform other functions as determined by the interrupts. As the DSP controller was required to perform several functions simultaneously, the software was made interrupt driven.

7.2.2 The interrupts

The DSP used a peripheral interrupt expansion system [28] in order to accommodate the large number of possible interrupts that could be generated. This created six maskable interrupt levels which could be configured for a particular peripheral interrupt. Some peripheral interrupts have both a high and low priority Interrupt level, whilst others can only be configured to certain Interrupt levels.

These levels also have an order of priority, from highest (level 1) to lowest (level 6). If a maskable interrupt is generated, the peripheral interrupt would be handled. However, if a maskable interrupt from a higher level is generated during this process, the associated higher level peripheral interrupt would first be handled and, thereafter, the peripheral interrupt associated with the lower level maskable interrupt would be completed.

[1] The external memory bus was inherently slower than the on-chip memory.

University of Cape Town

Priority	Interrupt level	Masked	Interrupt name	Source	Description
Highest	1	Yes			
⋮	2	No	TPINT1	EV	Timer 1 period interrupt
⋮	3	No	TPINT2	EV	Timer 2 period interrupt
⋮	4	Yes			
⋮	5	No	RXINT	SCI	SCI receiver interrupt
Lowest	6	No	XINT1 & XINT2	External	External interrupt pins

Table 7.1: Interrupt Source Priority and Vectors

The structure of the interrupt levels and the associated source is depicted in Table 7.1. A description for each level is given in Appendix A.

7.2.3 The main routine

The main routine served two functions:

- As the DSP was powered-up, the configuration of each peripheral was handled.
- Once configured, the software code entered a continuous loop which may or may not have performed any other tasks.

The configuration of the peripherals was done by writing the required values for each desired setting to the specific registers for each of the peripherals.

7.3 Code implementation

The code for the DSP Controller was written in ANSI C and compiled using TI C compiler for the 320F2xxx series of DSP's. As mentioned before, there was a body of previously written code which could be modified to suit this application.

Due to the DSP being interrupt driven, the code was divided into sections. Each interrupt level that was implemented contained the relevant code to perform the necessary functions of the robot. These functions included:

University of Cape Town

- control of the duty cycle (δ) for the PWM drive
- the control system implemented in Chapter 10
- the serial communication
- the external interrupts for the optical limit switches

Apart from the interrupt code, there was also a main routine which continuously cycled, but had no function in the control of the robot. As the main routine would only be serviced when the DSP was not handling interrupts, it was decided to use this for displaying relevant data on an intelligent LCD found on the Interface Board as described in Section 6.5.

7.4 Results and Conclusion

The Texas Instruments TI320F243PGEA DSP worked well as implemented here but, in terms of future robots, would most likely be unable to handle all the necessary calculations a more complex robot would require due to it being a fixed-point DSP. Whilst the $20\,MHz$ clock speed of this DSP enabled it to handle all the interrupts without any negative effect on performance, there are faster DSP's available with more features and larger on-chip memory that could better serve future developments.

Since the software was written in C and compiled for the TI320F243PGEA DSP, the code did not always perform as expected and therefore the code was modified to achieve the desired result. The only method of improving the current software for increased performance would be to re-write the existing software in assembly code.

University of Cape Town

Chapter 8

The Global Controller

8.1 Introduction

The Global Controller formed the second of the aforementioned three key components and, in essence, may be viewed as the brain of the whole system.

The idea of a global controller evolved from the simple Artificial Intelligence (AI) system which was undertaken by an Electrical Engineering honours thesis student in 2002 [29] to a complete system which, in 2003, comprised three distinct sub-systems:

- a serial communications module
- an AI system
- a joystick robot controller.

The development and implementation of the Global Controller was undertaken by an Electrical and Computer Engineering honours thesis student [30]. However, since the the Global Controller needed to be compatible with the two other key components, the specifications of the Global Controller are reported in this chapter.

University of Cape Town

8.2 Serial Communications Module

8.2.1 Background

During the initial stages of the UCT RoboSoccer campaign in 2002, it was assumed that the serial communication would be easily implemented, since initial testing between the computer and robot had proved successful when using a serial cable. As such, no significant other development was done and the serial communications became an add-on to the AI Robot Controller.

However, following the initial implementation of the Linx RF modules (Section 6.6.1) before the F180 RoboSoccer 2002 competition, it was discovered that the serial data was being corrupted. Although data packets were being transmitted and received, the received data did not match the data that was being transmitted and, consequently, the robots could not interpret the commands from the AI system. Due to this problem the UCT F180 RoboSoccer team was unable to compete in the F180 RoboSoccer 2002 competition held in Pretoria.

University of Cape Town

8.2.2 Further Development

After the difficulties experienced in Pretoria, the implementation of the Linx RF modules was re-examined. As a result, two possible sources of error were identified. The first of these was the RF antenna and the solutions to this problem have already been dealt with in Section 6.6.1. The second source of error arose from the Linx RF receiver module only allowing a maximum time lapse of $33ms$ between data output transitions. If this maximum time period was exceeded, an error in transmission occurred if any of the lower four bits of a subsequent byte were 0's. In order to avoid this, either the delay had to be kept less than $33\,ms$ or the first byte sent following the delay had to have a value of 255 (0xFF)[24].

8.2.3 Serial Communications Protocol

Since both the DSP controller and computer supported the Universal Asynchronous Receiver Transmitter (UART) protocol, UART serial communication was chosen. This protocol required that, to ensure reliable serial data transmission, both the DSP controller and the Personal Computer (PC) had the same settings. As the DSP controller had more

limited settings, the settings on the computer were made to match the best solution that could be implemented on the DSP controller. The settings were as follows:

- parity bit disabled
- one stop bit
- 8 data bits
- data rate of 19200 baud.

8.2.4 Command Structure

In designing a command structure through which the AI system would send the robots commands, several aspects needed to be considered:

- the serial data would consist of 8 data bits (1 byte)
- the first byte had to be 255 (0xFF)
- the structure had to contain both a command and the required set-point
- the command structure had to be able to identify a robot for which the command is intended
- the structure had to be robust in order to prevent misinterpretation of commands.

In order to meet these requirements the data packet for each instruction was structured as follows:

```
255 (0xFF), robot number (x), command, set-point, command
```

Although the data packet could have been made smaller, the choice of this option simplified the implementation of the code across the five robots, the Joystick Robot Controller and the AI system. As all five robots received all the data packets, the structure enabled the robots to efficiently identify and perform relevant commands. The values for the commands and setpoint are shown in Table 8.1. Within this structure a set-point of 128 represents an actual setpoint of zero, with values less than 128 representing negative setpoints.

The values for each command were chosen to maximise the difference between the bit patterns so as to lessen the chance of an error occurring.

University of Cape Town

Command	Value	Set-point
Left Drive System update	15	$0 \rightarrow 255$
Right Drive System update	33	$0 \rightarrow 255$
Kicker update	204	$80 \rightarrow 200$
PWM enable	240	0/1

Table 8.1: Command values and set-point ranges

8.2.5 Implementation

The implementation of the serial communication protocol was done in parallel between the DSP controller SCI used by the robots and the UART serial port of the PC used by the Global Controller.

The implementation on the DSP controller SCI was handled through the use of interrupts as discussed in Section A.1.

8.3 Artificial Intelligence System

In order to meet the needs of the UCT F180 RoboSoccer team the AI system had to:

- communicate with the Vision System Server via a TCP/IP network for positional and velocity data
- use the first quadrant co-ordinate system
- be able to control between one and five robots
- be able to control the robots in either direction of play
- communicate with the serial communications module for sending commands to the robots
- be implemented in C++ under Linux

The structure of the network packet as developed between the Vision System and the Global Controller is defined in Table 8.2 [30].

University of Cape Town

Data	Type	Size (*Bytes*)	Range
Identification number	Unsigned Int	4	$1 \rightarrow 11$
X position	Unsigned Int	4	$0 \rightarrow 290\,cm$
Y position	Unsigned Int	4	$0 \rightarrow 240\,cm$
Heading	Unsigned Float	4	$0 \rightarrow 360^{o}$
Velocity	Unsigned Float	4	$0 \rightarrow 5\,ms^{-1}$

Table 8.2: Network Packet Structure

8.4 Joystick Robot Controller

The rationale for the use of joysticks to control the robots was based on two influencing factors: firstly, the observation that other teams used gamepads to control their robots in human controlled games in the Pretoria 2002 campaign and, secondly, the need to devise a method by which to test the UCT F180 RoboSoccer system. Based on these factors a Joystick Robot Controller was decided upon.

The system had to:

- support up to five USB Microsoft Sidewinder Joysticks®
- allow for linear control of the robots
- be user configurable for the buttons and robot number
- use the slider for Kicker speed
- kick / dribble only while the relevant button was being pushed.

University of Cape Town

8.5 Results

In testing the serial communication between the robots and the Global Controller it was found that the Linx RF modules worked well given the new command structure, and no significant error was noted. Overall, the serial communications module performed well throughout testing and during the F180 RoboSoccer 2003 competition.

The Joystick Robot Controller was successfully implemented with only a few changes needing to be made to the sensitivity settings of the joystick. The control of the robots through

the use of the joysticks was intuitive but led to a simple solution for the control of the robots.

In the end the AI system was not fully tested due to the problems associated with the Vision System that are outlined in Section 9.7 and, as such, no results were derived for this system.

8.6 Conclusion

The implementation of the Serial Communications Module and the Joystick Robot Controller proved very successful and could thus be re-used without the need for any changes, thereby allowing future development to focus on other areas that require more urgent attention.

The AI system will need to be re-examined in more detail once the Vision System is properly implemented.

University of Cape Town

Chapter 9

The Vision System

9.1 Introduction

The Vision System is the last of the three vital components since without the Vision System the AI is, quite literally, blind thereby rendering the other components redundant.

At the start of the F180 RoboSoccer campaign in June 2002, the Vision System benefitted from being an Undergraduate Honours thesis topic. The Vision System was required to:

- Capture an image of the playing surface using a CCD camera and TV capture card
- Transform the image to correct for distortion
- Determine the positions of the ten robots (including their orientation) as well as the ball
- Create a GUI showing the position and orientation of the robots and ball.

The Vision System was undertaken by a final year Electrical Engineering student [31]. Like the rest of the F180 RoboSoccer research group, this student had very little background in image capturing and processing which, if not for this student's hard work, would have become a serious problem. Although the Vision System was completed, it was not very reliable and was extremely tricky to calibrate.

In order to circumvent the problems from the previous year, the Vision System was handed over for supervision to a staff member based within the UCT Digital Image Processing

University of Cape Town

Group. A new set of requirements was drafted which can be seen in Section 9.2. In July 2003 the vision system was again the focus of an undergraduate project. However, the results of this proved inadequate for the RoboSoccer needs and thus the Vision System was undertaken by a team of collaborators.

An outline and the results of the collaboration are reported in this chapter. Details of the collaborators may be found in the Acknowledgements.

9.2 Requirements

It was decided that the Vision System needed to:

- Act as the Server providing data for the AI client
- Capture at least 10 frames per second, and preferably 15 or more
- Have a process latency of less than or equal to 2 frames
- Determine the position, orientation and velocity of the UCT F180 RoboSoccer robots
- Determine the position and velocity of the orange golf ball
- Determine the position of the Opposition F180 Robosoccer robots
- Have a GUI showing the above information
- Be easy to configure
- Be accurate and robust
- Be implemented in Matlab to simplify development.

University of Cape Town

9.3 Vision System Hardware

The selection of the hardware for the Vision System was difficult. Due to a limited budget and with no guarantee on the successful implementation of the hardware it was difficult to motivate for expensive hardware as used by international F180 RoboSoccer Teams. For example, Cornell [32] uses the following:

- a Dual Xeon 2 GHz machine with 1 Gb system memory
- a Matrox Meteor II Multi-Channel frame grabber card
- Mil 7.0 Library and hardware dongle
- Sony DCX-9000 camera with zoom lens

for tracking the robots and

- a Dual Xeon 1.7 GHz machine with 480 Mb system memory
- Two Matrox Meteor II Multi-Channel frame grabber cards
- Mil 6.0 Library and hardware dongle
- 2× Pulnix TCM-6700 high resolution cameras

for tracking the ball.

However, since the cost for any of the above items would have used the entire UCT F180 RoboSoccer budget, alternate options had to be sought.

University of Cape Town

9.3.1 Camera selection

In order to save on expenses on vision system hardware, it was decided that either a IEEE1394 firewire web camera available from the Digital Image Processing Group at UCT or the existing vision system hardware would be used.

On testing the two different systems it became apparent that the IEEE1394 firewire web camera would perform better. The camera produced a higher quality image and did not require any added hardware such as a TV capture card. Furthermore, since most computers and laptops come standard with an IEEE1394 port there were no compatibility problems.

9.3.2 Computer selection

As a Celeron 400 MHz machine with 128 Mb of system memory was already available it was decided that it would be used until a faster machine was required. The Celeron thus acts as a benchmark on which to base buying a faster machine.

Linux was chosen as the operating system and Slackware 9.1 was installed and configured using 2.4.22 linux kernel and gcc-3.2.3. Other libraries and software included:

- The Intel Performance Libraries (IPP)
- Matlab 6.0 including the Image Processing Toolbox.

For the F180 RoboSoccer 2003 competition, a HP NX9010 laptop was used for greater processing power[1] and portability.

9.3.3 Frame grabber selection

The choice of the IEEE1394 firewire web camera negated the need for a frame grabber card.

9.4 Lighting

9.4.1 Introduction

Although lighting is not the only area of the Vision System that is important, it does form the platform on which the other components are built. For optimal conditions to exist, an even level of clean light is required across the whole playing surface of the field. Provided these conditions are met, the RGB values for each colour marker will remain the same anywhere on the playing surface.

[1]The HP NX9010 had an Intel 2.4 GHz P4 with 512 Mb of system memory.

University of Cape Town

9.4.2 Light source selection

The selection of a light source developed through the second semester of 2003. Originally, the playing field was lit with both sunlight and fluorescent tubes. Through the first stages of software development allowances for the lighting conditions could be made. Although not an optimal solution, it did allow progress to be made with the image processing software.

As the F180 RoboSoccer 2003 competition would be held indoors with no windows for sunlight, the windows were blacked out with black plastic sheeting to eliminate direct sunlight from the playing surface. The small amount of indirect sunlight was minimal and therefore ignored.

Once the lines were painted onto the field, the areas of the captured images around the lines became very noisy. This noise was significantly reduced by switching off the fluorescent lights in the Control Laboratory and the use of halogen floodlights. Since the fluorescent lights in the Control Laboratory were determined to be the problem, they needed to be replaced.

As it was not known if the noise was inherent to fluorescent lights or if the fluorescent lights in the Control Laboratory were faulty, it became necessary to select a different light source.

A quick investigation into light sources revealed metal-halide floodlights to be the industry standard when used for broadcasting live events at night. However, the problem with using the metal-halide floodlights was their cost. In comparison, the halogen floodlights previously tested were significantly cheaper which made them a more viable solution.

After more testing, the halogen floodlights were determined to be the best choice given the circumstances.

University of Cape Town

9.4.3 Experiments with positioning of lighting

In order to meet the minimum illumination levels required for competition, it was determined that a minimum of two halogen floodlights would be needed. To prevent shadows from the robots being visible to the camera the floodlights would have to be mounted as close to the camera as possible.

Unfortunately, irrespective of their orientation, the floodlights created visible intensity hot-spots. After more experimentation it was found that the best layout with two floodlights

was when they were spaced away from the centre of the field with their focal points directed at the opposite goals (Figure 9.1).

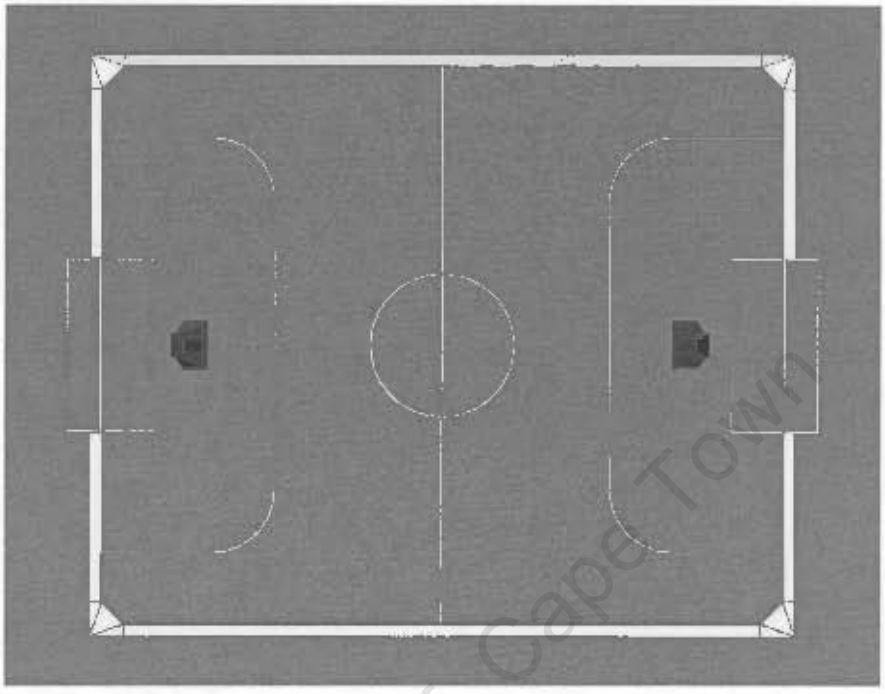


Figure 9.1: Top view of two floodlight layout
This shows an overhead view of the halogen floodlights above the playing field with their focal points directed at the opposite goals.

University of Cape Town

To further test the layout in Figure 9.1 an image was captured using the IEEE1394 firewire camera and a grayscale transform was done on the image to determine the intensity image (Figure 9.2).

The normalised intensity values across the X and Y axis (Figure 9.2) were then plotted in Figures 9.3 and 9.4 respectively. The spikes seen in Figures 9.3 and 9.4 represent the white field markings and boundaries.

From Figure 9.3, it can be seen that the intensity varied approximately 10 percent across the width of the field which was unacceptable.

To better understand the lighting set-up at the competitions, a Cornell highlights video of the 2003 RoboSoccer World Cup in Italy [10] was examined to ascertain the lighting set-up used in that event. A few shots showed similar floodlights mounted vertically high above the perimeter of the playing field.

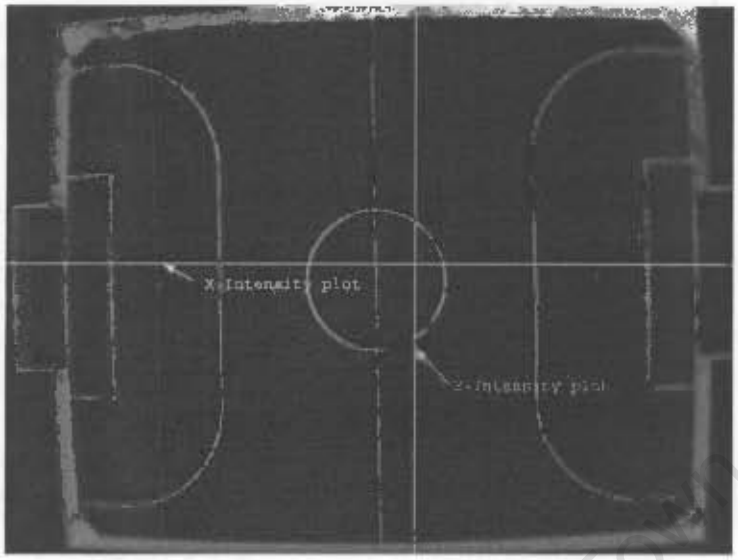


Figure 9.2: Intensity image of the playing field

This shows the light intensity from a captured image using the vision system. The intensity of light across both the length and width of the field (as shown) was also plotted to simplify measurement.

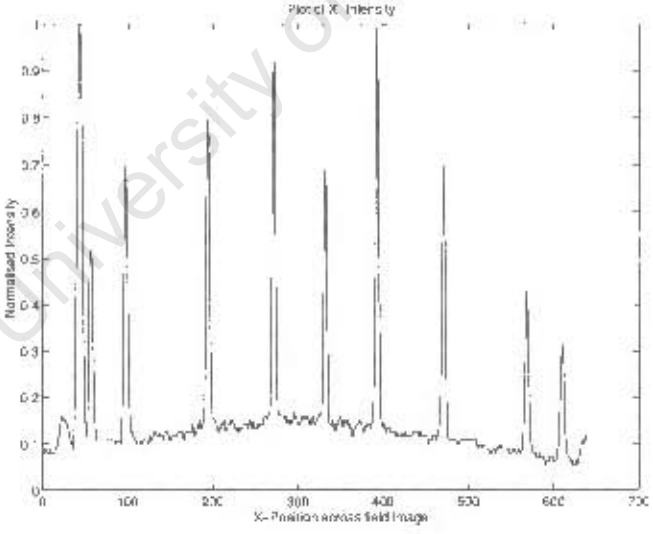


Figure 9.3: Plot of X-Intensities for two floodlight layout

This shows the intensity across the length of the playing field. The intensity rises from the goals to a maximum at the centre of the field with the spikes indicating the white field markings.

University of Cape Town

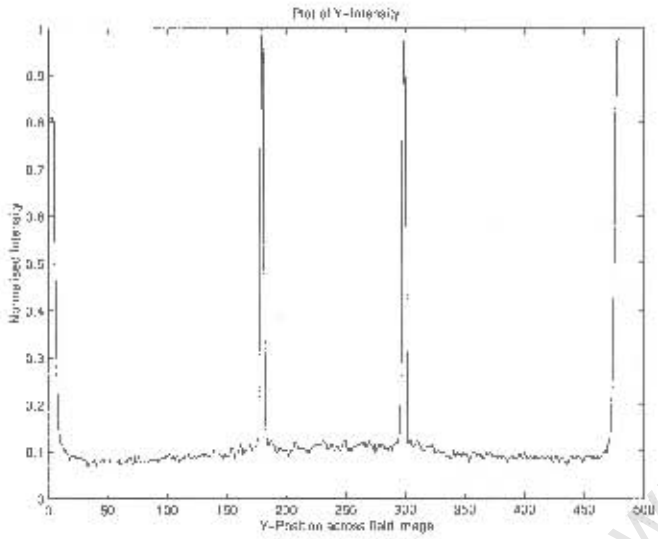


Figure 9.4: Plot of Y-Intensities for two floodlight layout
This shows the intensity across the width of the playing field. The intensity rises slightly from the side-lines to a maximum at the centre of the field with the spikes indicating the white field markings.

Based on this observation one floodlight, facing downward, was mounted vertically above each corner of the field. The lights were mounted just below the ceiling in the Control Laboratory so that they were $3\,m$ above the playing surface. The exact layout is depicted in Figure 9.5.

The normalised intensity values across the X and Y axis (Figure 9.2) were again plotted in Figures 9.6 and 9.7 respectively. As can be seen in Figure 9.6 a much better result was achieved when compared to the earlier values shown in Figure 9.3.

The slight increase in intensities on the left half of the playing field over the right half was probably due to the light from the floodlights above the left corners being reflected off a nearby wall on the left hand side of the playing field.

During the actual F180 RoboSoccer 2003 competition, another base image was captured and the normalised intensity values across the X and Y axis (Figure 9.2) were plotted in Figures 9.8 and 9.9 respectively. As the playing field was out in an open hall for the competition, the results from Figure 9.8 prove that the wall in the Control Laboratory was indeed the cause for the increased intensity on the left half of the field as shown in Figure 9.6.

University of Cape Town

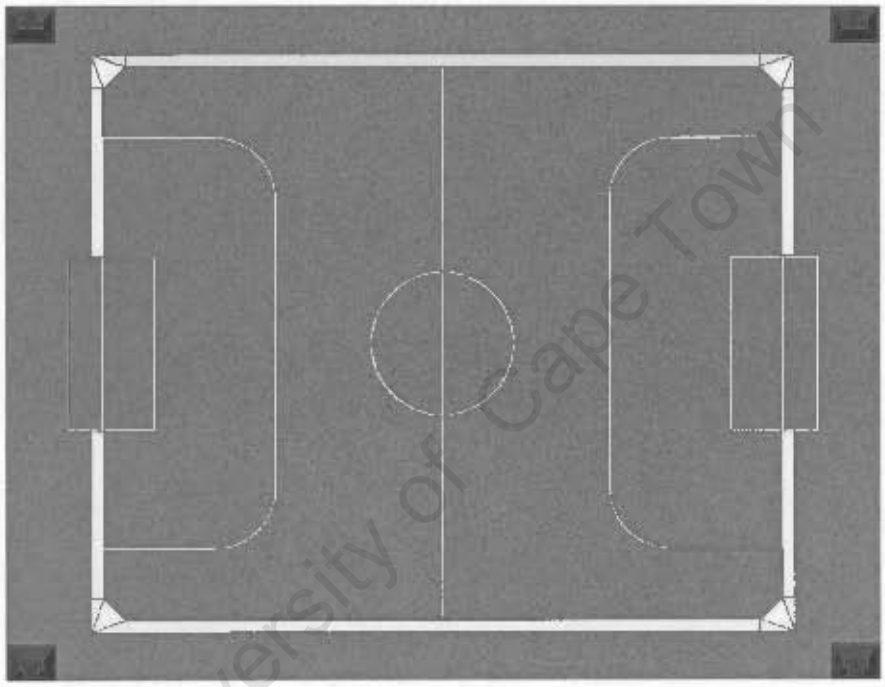


Figure 9.5: Top view of four floodlight layout

This shows an overhead view of the halogen floodlights above the playing field with their focal points directed straight downward.

University of Cape Town

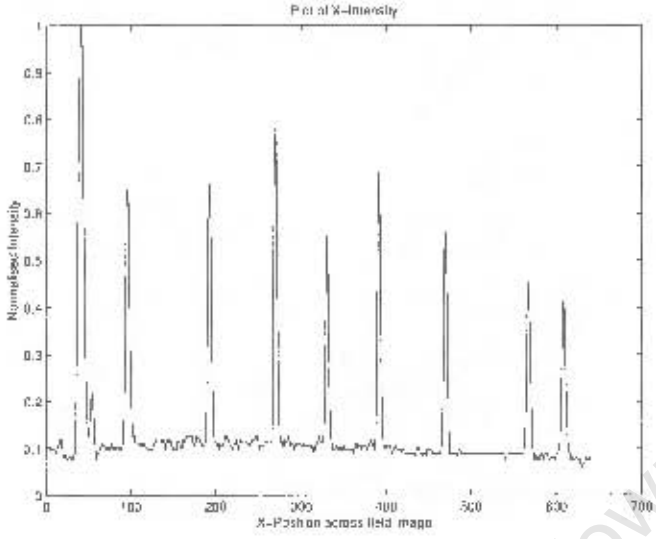


Figure 9.6: Plot of X-Intensities for four floodlight layout

This shows the intensity across the length of the playing field. The intensity was slightly raised at the one goal, but was relatively even across the rest of the field with the spikes indicating the white field markings. This raised intensity was from a nearby wall reflecting extra light over the area.

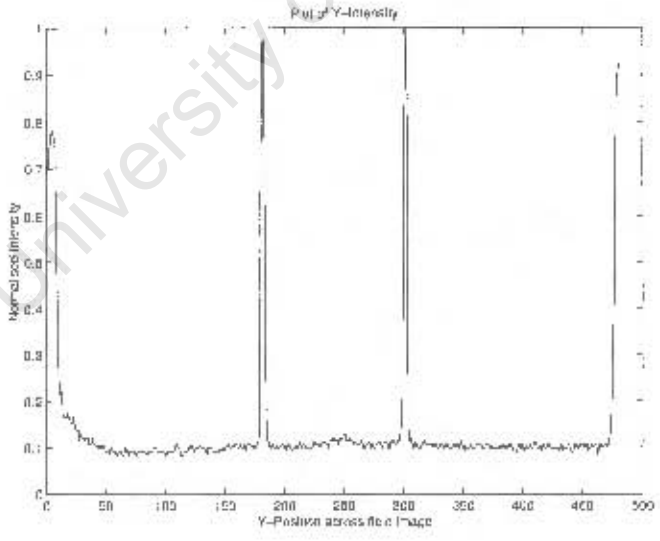


Figure 9.7: Plot of Y-Intensities for four floodlight layout

This shows the intensity across the length of the playing field. The intensity remained relatively even across the width of the field with the spikes indicating the white field markings.

University of Cape Town

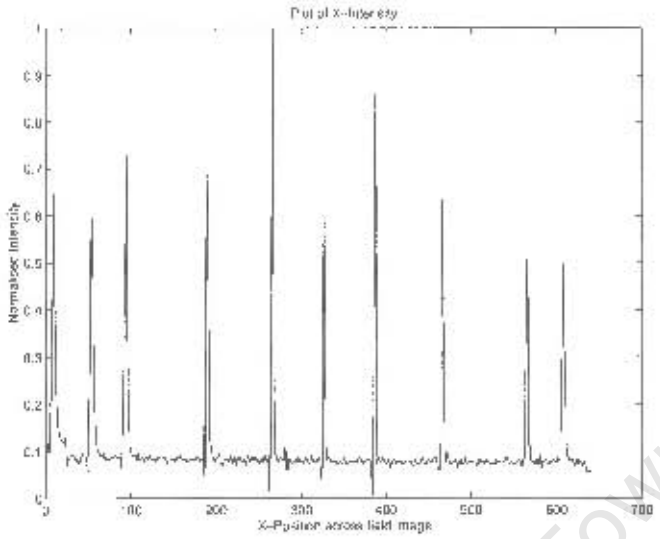


Figure 9.8: Plot of X-Intensities for the competition
This shows the intensity across the length of the playing field during the 2003 competition. The intensity was even across the field with the spikes indicating the white field markings.

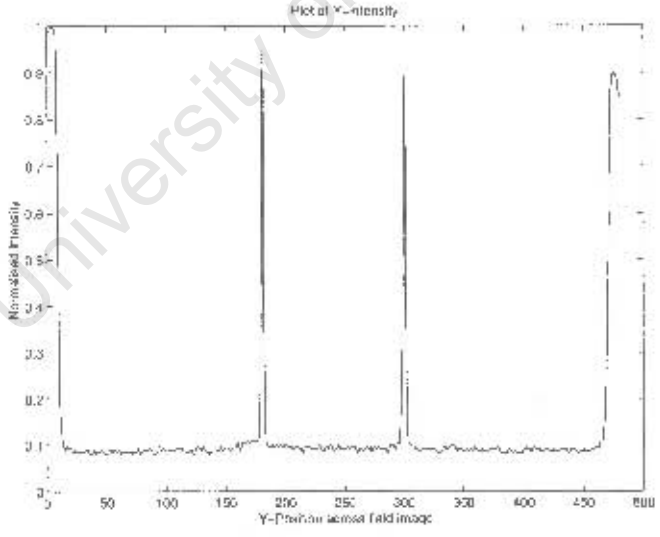


Figure 9.9: Plot of Y-Intensities for the competition
This shows the intensity across the width of the playing field during the 2003 competition. The intensity was even across the field with the spikes indicating the white field markings.

University of Cape Town

9.4.4 Conclusion

Through the use of four halogen floodlights set-up as in Figure 9.5, a simple and cost effective lighting system was created which provided the different teams with an even distribution of light across the entire playing field.

9.5 Robot Marker System

9.5.1 Choice of Colour Markers

In order to distinguish between the teams during a match, each team had either a yellow or blue team marker placed in the visual centre of their robots. The team marker was circular with a diameter of 40 mm.

According to international regulations teams may use either black or white colouring, without restriction, to identify individual robots. The teams may also use light green, light pink and cyan colour markers (as defined by the competition organisers) during the competition. [2].

9.5.2 Marker Pattern

As the team marker is circular and the UCT robot was mostly circular, the markers used for orientation and identification were also kept circular. In keeping all the markers circular, the image processing software would require only one algorithm thereby simplifying the software.

Needing to meet the following requirements of:

- position
- orientation
- identification

[2]UCT hosted the F180 RoboSoccer 2003 competition and were therefore the competition organisers.

University of Cape Town

meant that one colour disc was needed for orientation and another three colour discs for identification markers. The diameter of these discs was also made 40 mm. If the discs were made larger, the markers would infringe on each other in the captured image and the position of the centre of the disc would become less accurate. If the disks were made smaller, there would be insufficient colour information to determine the exact colour of the disc.

Light pink was chosen for the direction marker and light green was chosen for the identification markers. As either a blue or yellow marker disc was located in the centre of the robot, the light pink marker disc was placed in the front right hand corner of the robot with a binary pattern of light green marker discs at the other "corners" of the robot. The marker pattern for each robot can be seen in Figure 9.10.

9.6 Image Processing

9.6.1 Introduction

Due to the changes from 2002 to 2003, the image processing from 2002 was scrapped and developed from scratch. Unfortunately for the project, the design of the Vision System was completed before the final version of the rules was decided. Thus, when the the field markings based on the final rules were painted, these white field markings caused erroneous results due to limitations in the design of CCD technology which ultimately rendered the vision system unsatisfactory. As a result further developments were undertaken.

9.6.2 Initial Vision System

After defining the marker system a vision system implementation was written in Matlab 6.0. The low-level image processing functionality made use of Intel's performance libraries, which has a C calling interface. Where appropriate, this code was incorporated into Matlab by compiling it into the form of a mex function which enabled the Vision System to run at 8 frames per second. The flow chart of this work can be seen in Figure 9.11.

University of Cape Town

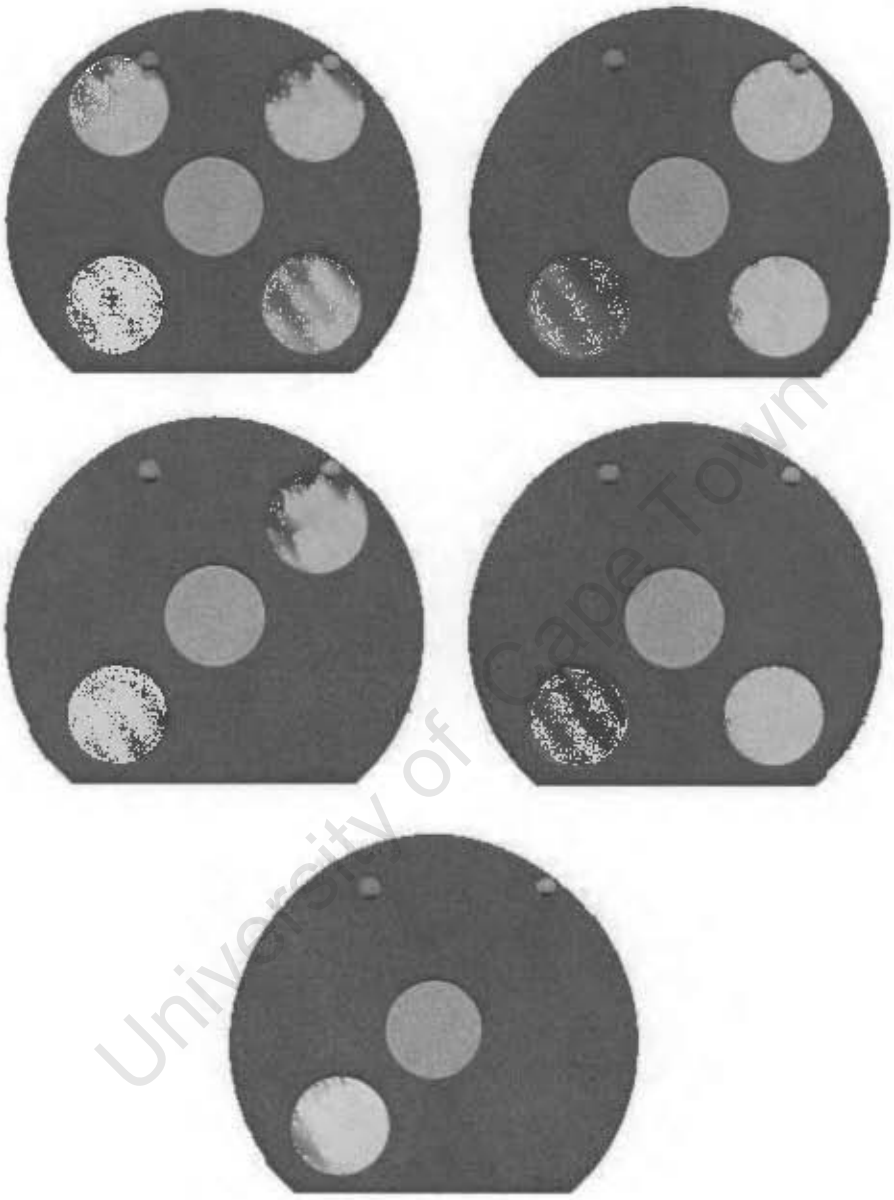
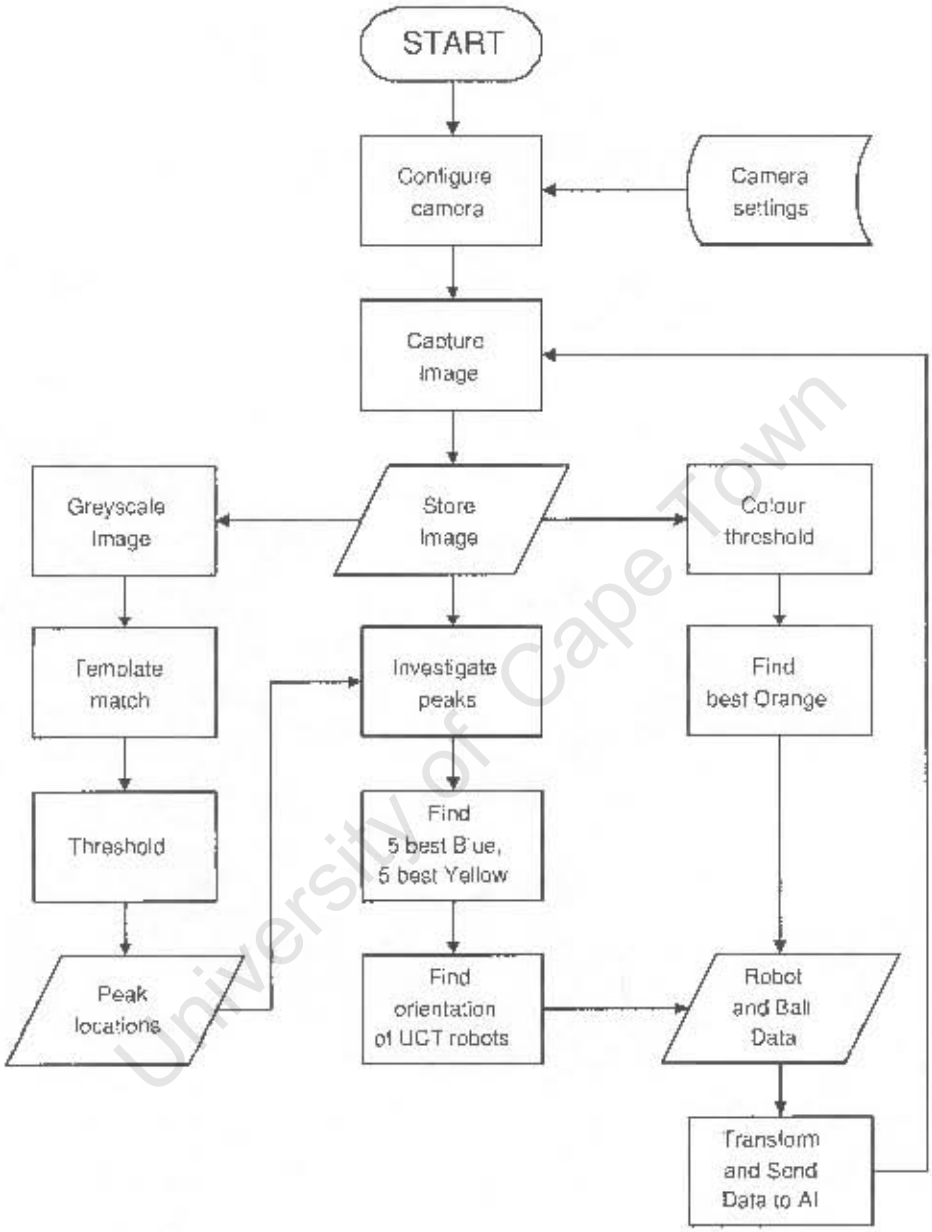


Figure 9.10: Marker patterns for each robot

This shows the actual tops of the robots and their respective marker patterns used in the 2003 competition. UCT played with the blue markers whilst the pink marker represented direction and the light green markers were used to identify the individual robots.

University of Cape Town



University of Cape Town

Figure 9.11: Flow chart of the Vision System

This shows the image processing routine employed by the vision system. The identification of the ball and robots was done using separate paths within the routine.

9.6.3 Further Developments of the Vision System

After the addition of the white field markings, the Vision System also needed to be improved thereby requiring adaptations in the vision system code.

The simplest solution was to take the captured image and process it so as to remove the lines before using the new image in place of the captured image in the vision system as outlined in Figure 9.11.

The standard method used in these cases is that of background subtraction where a base image without the robots or ball is subtracted from the captured image. Further steps are then required to correct any imperfections in the image before it can be used as a template to extract the relevant data from the captured image to create a new image. This process is outlined in the flow chart in Figure 9.12.

9.7 Results

The first implementation (Figure 9.11) proved successful without the white field markings but unsuccessful when the field markings were later added. The final implementation of the Vision System was inconsistent. The exact problem was unknown, but was suspected to lie within the implementation of the adaptations as outlined in Figure 9.12.

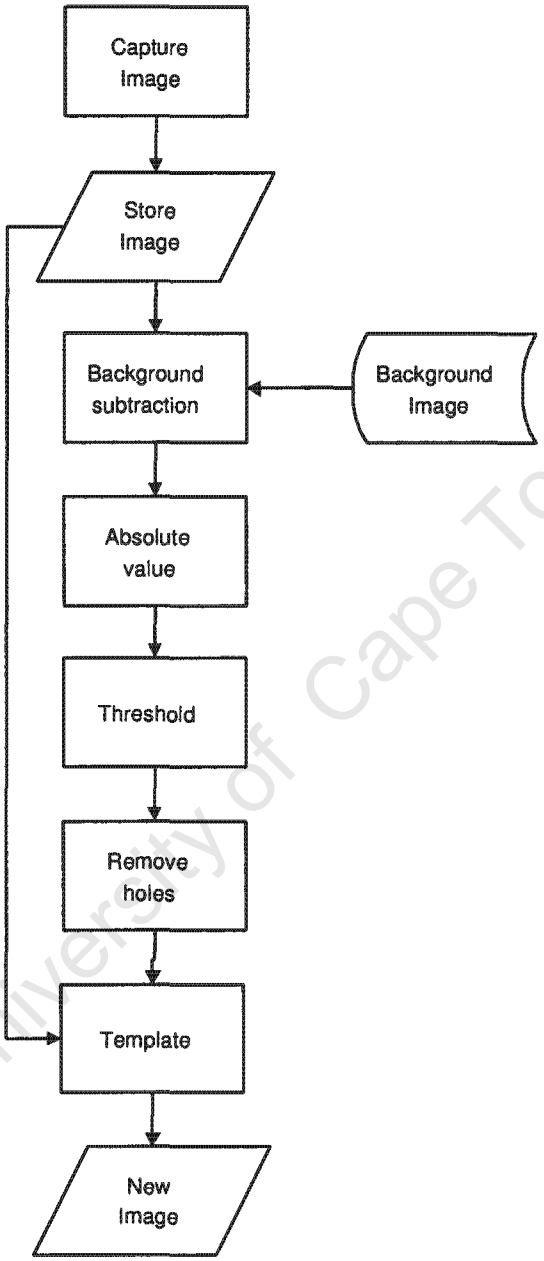
The successful parts of the Vision System were:

- The lighting system
- The implementation of a IEEE1394 firewire camera
- The robot marker system
- The use of Matlab 6.0 as a development environment

9.8 Conclusion

Even though the Vision System failed to work overall, many parts of the Vision System were very successful. It is these successful components that may be carried over in future development in the hope that, through building on a sound foundation, an efficient and robust Vision System will be developed.

University of Cape Town



University of Cape Town

Figure 9.12: Flow Chart of vision system modifications
This shows the changes that were attempted to enable the previous image processing routine to cope with the white field markings.

Chapter 10

Motion Control

10.1 Introduction

Once all the sub-systems of the RoboSoccer project had been combined, proper testing of the whole system could be implemented. Apart from problems regarding individual sub-systems (as previously discussed in their relevant sections) it became clear that the robots were difficult to control. Although their response to AI system set points could be predicted for a given set of operating conditions (for each of the five robots), as these operating conditions were, in reality, continuously changing it became apparent that some form of control was required.

University of Cape Town

10.2 Determination of a System Model

As a starting point, it was decided to model each of the drive systems as an independent system. This simplified the input to the systems and also stayed in line with the AI system which simply set a speed for both the left and right wheels. The System Model consisted of a Drive System (motor / gearbox and wheel) which drove the mass of the robot. In this model the effect of the forces generated from one drive system on the other drive system were assumed to be negligible. Furthermore, as both Left and Right Drive Systems were the same, a common Open Loop System Model (Figure 10.1) could be used.

The input to the Drive System from the AI system is a PWM value from 0 to 255. This PWM value sets the duty cycle (δ) for the H-Bridge which is equivalent to setting a voltage

V on the input. The output from the Drive System is measured in counts per sample time (C_x) where x represents the sample period in ms.

Figure 10.1: Open Loop System Model of the Drive System
This shows an open loop representation of the robots drive system (the drive system included the DC geared motors, wheels and robot inertia).

Since the Drive System consisted of a geared DC motor acting on a load (the mass of the robot), the dynamic performance of each Drive System was assumed to be a standard first order system with the following transfer function [33]

$$g(s) = \frac{A}{1 + sT} \tag{10.1}$$

where

- A was the gain in $[C_x\delta^{-1}]$
- T was the time constant $[s]$

A series of Step Tests were performed under different conditions to determine the gain A and time constant T for the Drive System.

In order to derive T from the Step Tests, the relationship between time t and time constant T could be found for any step response $y(t)$ as $y(t)$ approaches BA. This could be calculated as follows from Equation B.3 in Appendix B.

$$t = -T \times \ln\left(1 - \frac{y(t)}{BA}\right) \tag{10.2}$$

It therefore followed that the time t taken for $y(t)$ to reach 90% of BA was related to the time constant T as follows

$$t = 2.3T \tag{10.3}$$

For each series of Step Tests, a code was written for the on-board controller to perform. At the end of each run the final value for speed and the number of sample periods (count) taken to reach a set-point were displayed on the LCD panel.

University of Cape Town

10.2.1 Step Test 1

In previous testing, the robots performed differently to each other even when their operating conditions were the same. This meant that either the drive systems were incorrectly aligned on each robot or each drive system was sufficiently different to cause these variations. So as to test the Drive Systems independently, the robot was lifted off the ground thereby cancelling all effects of inertia from the mass of the robot.

The Program Flow Chart is shown in Appendix B.3.1. The results from Step Test 1 are summarised in Table 10.1 with the complete results available in Appendix B.3.2.

Drive System	Direction	Sample Time (ms)	Max Count	Time Constant T
Right	F / R	100	92 / 92	200
Left	F / R	100	96 / 86	200
Right	F / R	25	23 / 23	200
Left	F / R	25	24 / 21	175

Table 10.1: Summary of Step Test 1 results

University of Cape Town

Results

- The two Drive Systems performed significantly differently to each other in final speed (Max Count) and only slightly in acceleration (Time Constant T).
- Although the Right Drive System performed the same in both forward and reverse motion step tests, the Left Drive System had a large variation for forward and reverse step tests with regard to final speed possibly due to DC geared motor imperfections.
- The 25 ms sample time provided a greater resolution compared to a larger sample time (100 ms) for the time constant and hence provided better accuracy.
- The 12.5 ms sample time returned vastly inconsistent values which were too far out of range and these values were therefore deemed to be outliers and were accordingly excluded. Even after many iterations and code revisions the same type of results were returned. As only the interrupt period was effectively changed and the time was still well within the chip specifications, these inconsistencies could have been attributed to a C-compiler bug or a chip model flaw.

Conclusions

- The difference in maximum speed for each drive system matched the fact that at steady state the robot did not move in a straight line. Therefore the maximum speed for each Drive System needed to be governed so as to ensure motion in a straight line. This also had the effect of having the same top speed for a larger range of battery voltages i.e. as the batteries discharged.
- The difference in acceleration (time constant T) did not match the amount of difference in the acceleration between the two Drive Systems during normal testing which therefore indicated that one Drive System must have been actually slipping during acceleration.
- Based on the test results it was decided that a 25 ms sample time would be adopted for the control loop.

10.2.2 Step Test 2

Having determined that slippage was the greatest cause of the robots veering to one side during acceleration it was decided to run the second Step Test on a much rougher surface in order to increase the friction between the wheels and the ground. Due to both the balance of the robot and the small contact patches used for this balance the negative effect of the extra friction on the time constant (T) could be ignored.

With the maximum speed expected to be above 20.0 C_{25} (Counts per 25 ms) from the previous results in Appendix B.3.2, the 63% value was discarded in favour of using a set value. A speed of 20.0 C_{25} was adopted in establishing the time constant, as this ensured a reasonable number of sample periods.

The modified Program Flow Chart is shown in Appendix B.4.1. The results of the second Step Test are summarised in Table 10.2 whilst the complete results are presented in Appendix B.4.2.

Ave Max Speed Left	Ave Count	Ave Max Speed Right	Ave Count
21.5 C_{25}	15.7	20.4 C_{25}	20.3

Table 10.2: Summary of Step Test 2 results

University of Cape Town

Results

From Table B.5 the calculated average maximum speed was $21.5\,C_{25}$ and $20.4\,C_{25}$ whilst the average count is 15.7 and 20.3 for the Left and Right Drive Systems respectively. Given that the number of sample periods was measured up to a set-point of $20.0\ C_{25}$, it was possible to determine the time taken to reach 93.0% and 98.0% of BA for the Left and Right Drive Systems respectively.

From Equation 10.2 it follows that $t = 2.66T$ for the Left Drive System and similarly $t = 3.92T$ for the Right Drive System. This gave a final time constant T of $0.15\ s$ for the Left Drive System and $0.13\ s$ for the Right Drive System.

The gain A was calculated from Equation B.3 and found to be 0.084 and 0.080 $[C_x\delta^{-1}]$ for the Left and Right Drive Systems respectively.

Conclusion

Two results from this second Step Test differed from what had been expected. Firstly, the time constants from Step Test 2 were quicker than those under no load conditions in Step Test 1. Secondly, the time constant of 0.13s was suspicious given that the time constant was calculated from the 98.0% point where the chance of error was increased.

University of Cape Town

10.2.3 Step Test 3

In order to confirm the results from Step Test 2, another series of step tests was needed. To ensure that the time constant would not only be correct, but also comparable between the Drive Systems, it was found that the 90% point of the final speed yielded values of [1] $19.0\,C_{25}$ and $18.0\,C_{25}$ respectively for the Left and Right Drive Systems. These new speed values were then used in place of the speed of $20.0\,C_{25}$ that had been used in Step Test 2. Apart from the new speed values the third Step Test was, in all other aspects, identical to Step Test 2.

The results of the third Step Test are summarised in Table 10.3 with the complete results presented in Appendix B.5.1.

[1] The 90% point of the final value was a good compromise between a high count and the chance of error from using a high set-point.

Ave Max Speed Left	Ave Count	Ave Max Speed Right	Ave Count
21.5 C_{25}	13.6	20.0 C_{25}	14.5

Table 10.3: Summary of Step Test 3 results

Results

From the results in Table 10.3 and Equation 10.2 where $t \approx 2.30T$, a time constant T of $0.16\ s$ was calculated for both Left and Right Drive Systems.

The gain A was found to be 0.084 and 0.078 $[C_x\delta^{-1}]$ for the Left and Right Drive Systems respectively.

Conclusion

In this last iteration of the Step Test, the values for both Drive Systems had converged. These results confirmed the data obtained in Step Test 1 which had indicated that, although the Drive Systems rotated in opposite directions, that the Left and Right Drive Systems should have had similar time constants since the direction of rotation of the Drive System had no difference on the time constant.

University of Cape Town

10.2.4 Open Loop System Model

From the previous step tests, the time constant T was considered to be $0.16 \pm 0.2\ s$ whilst the gain A was $0.081 \pm 0.003\ [C_x\delta^{-1}]$ which should have covered any differences between the Drive Systems.

The Open Loop System Model for the Drive Systems therefore had the following transfer function

$$g(s) = \frac{0.081 \pm 0.003}{1 + s\,(0.16 \pm 0.02)} \left[C_x\delta^{-1}\right] \tag{10.4}$$

In order to implement the controller digitally, a Step Invariant pulse transfer function was required for the following digital representation (Figure 10.2). The change in input range in Figure 10.2 from before is due to the change in period for the PWM module.

Figure 10.2: Open Loop Digital System.
This shows a digital open loop representation of the robots drive system (the drive system included the DC geared motors, wheels and robot inertia).

The transfer function for the zero order hold circuit (ZOH) [34] was given by

$$h(s) = \frac{1 - e^{-sT}}{s} \tag{10.5}$$

where

- T was the period of the sampling loop

The pulse transfer function for the Open Loop Digital System was given by

$$gh(z) = Z\left[g(s)h(s)\right] \tag{10.6}$$

$$= Z\left[\frac{0.081 \pm 0.003\left(1 - e^{-sT}\right)}{s\left(1 + s\left(0.16 \pm 0.02\right)\right)}\right] \tag{10.7}$$

where $T = 25ms$, simplifying the equation yielded

$$gh(z) = \frac{11.825 \pm 0.001}{z - 0.8534 \pm 0.0169} \tag{10.8}$$

10.3 Controller Design

The main function of the control was to reduce the effect of the model changes on the motion of the robot.

University of Cape Town

10.3.1 Closed Loop Digital System

The Closed Loop Digital System for the Drive System is depicted in Figure 10.3.

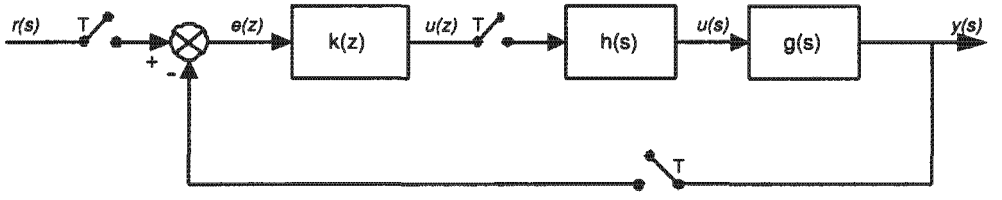


Figure 10.3: Unity Feedback Digital Control Loop
This shows digital closed loop representation of the robots drive system (the drive system included the DC geared motors, wheels and robot inertia). All the system blocks except g(s) are actually within the DSP controller.

where

- $k(z)$ was the controller
- $r(s)$ was the set-point
- $e(z)$was the error $(r(z) - y(z))$

10.3.2 Design of k(z)

In order to design a control law for the Drive Systems, the Root Locus (in the z-plane) design method was chosen. In Figure 10.4, the position of the open loop pole lay between 0.8365 and 0.8703 (Equation 10.8) for the Open Loop Digital System.

In order to ensure that the set-point $r(t)$was tracked with zero error a PI controller was used as the starting point for the design. The form of the PI controller [33, 34] was

$$k(z) = K\frac{(z-\beta)}{(z-1)} \tag{10.9}$$

where

University of Cape Town

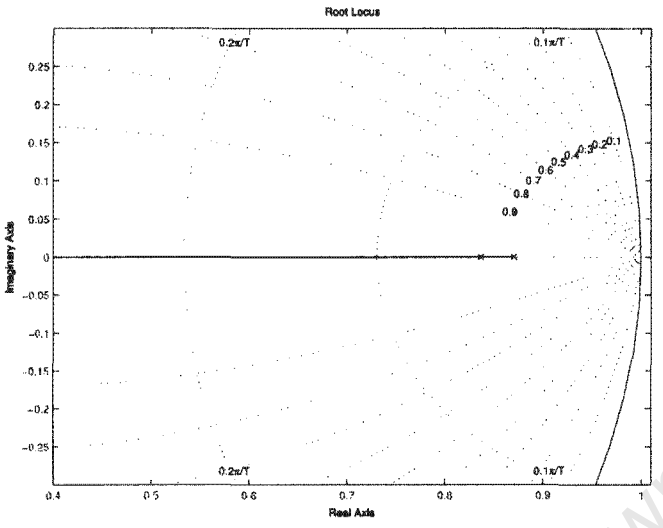


Figure 10.4: Root Locus diagram for $gh(z)$ in equation 10.8
This shows how the position of the open loop pole moved (between the two x's) with respect to changes in the drive system dynamics.

- K was the controller gain
- β was a constant (dependent of the time constant for the integrator).

In choosing the value for β the following was considered:

- As $u(t)$was limited to the voltage from the battery pack making β small would have limited the region of stability (For a larger range of gain the closed loop poles were off the z axis.).
- If the zero was placed too close to the open loop pole from the Drive Systems pole zero cancellation would have occurred and the internal dynamics would, in all likelihood, have caused problems.

Through making $\beta = 0.7$, $k(z)$ became

$$k(z) = K\frac{(z-0.7)}{(z-1)} \tag{10.10}$$

yielding Root Locus diagrams (Figures 10.5 and 10.6) for the Closed Loop Digital System shown earlier in Figure 10.3.

University of Cape Town

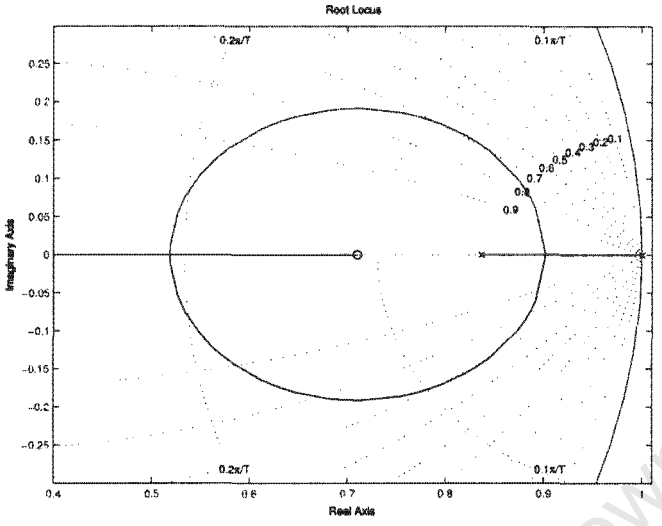


Figure 10.5: Root Locus diagram for the fast Drive System pole ($T = 0.14$)
This shows the position of the closed loop poles for the drive system with the fast system pole. The gain of the controller was set to ensure the closed loop poles had a damping factor greater than 0.9 *and remained on the positive side of the real axis.*

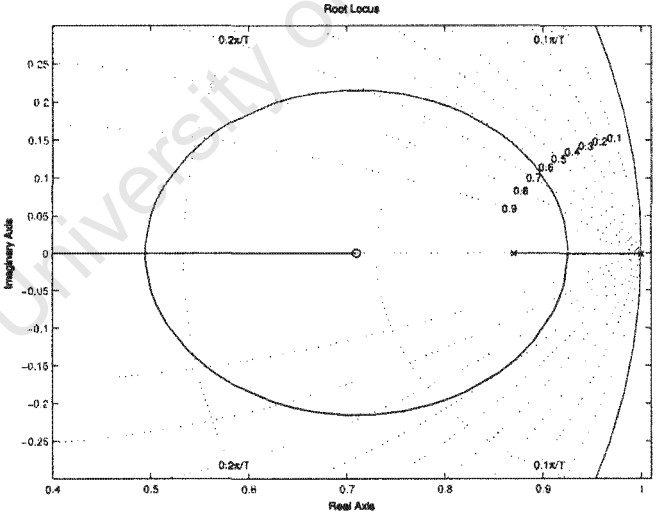


Figure 10.6: Root Locus diagram for the slow Drive System pole ($T = 0.18$)
This shows the position of the closed loop poles for the drive system with the slow system pole. The gain of the controller was set to ensure the closed loop poles had a damping factor greater than 0.9 *and remained on the positive side of the real axis.*

University of Cape Town

Gain K was then chosen to ensure that the closed loop poles were stable ($0 \leq z \leq 1$), damped ($damping \geq 0.9$) and suitably fast ($z \leq 0.8$). Based on Figures 10.5 and 10.6 the gain K had to be between 10 and 13 in order to ensure that variations in the Drive Systems ($\pm 12.5\%$ change in time constant) would give predictable results.

10.3.3 Implementation of k(z)

Having determined the digital control law

$$k(z) = 12\frac{(z-0.7)}{(z-1)} \tag{10.11}$$

direct programming [34] was implemented to find the necessary algorithm. This algorithm was derived from

$$k(z) = K\frac{(z-\beta)}{(z-1)}\frac{z^{-1}}{z^{-1}} = \frac{u(z)}{e(z)} \tag{10.12}$$

which yielded

$$u_n = u_{n-1} + Ke_n - K\beta e_{n-1} \tag{10.13}$$

on simplification. In addition, e_n had to be calculated as follows

$$e_n = r_n - y_n \tag{10.14}$$

Once u_n was calculated the values for the past values of u_n and e_n needed to be stored for use in the next loop of the algorithm as follows

$$e_{n-1} = e_n \tag{10.15}$$

and

$$u_{n-1} = u_n \tag{10.16}$$

As a precaution, limits were set for u_n to ensure $u(t)$ remained within its constraints. u_{n-1} and e_{n-1} were initially set to zero as the Drive System started from rest. As there were no further poles in $k(z)$, no further initialisation was required.

University of Cape Town

10.3.4 Evaluation of Closed Loop Digital System

After vigorous testing it was found that the robots now ran in a straight line in steady state at all speeds. The robots also decelerated in a straight line without fail. However, on occasion, during quick acceleration the robots again veered off the planned path. As this behaviour did not present itself during slow acceleration tests this led to the conclusion that either one or both wheels was losing traction. Preventing this from occurring would have meant either replacing the tyres [2] or introducing a slower pole in order to slow down the acceleration (dynamic response). The latter option was a simpler solution and was implemented using a pre-filter.

10.4 Pre-filter Design

10.4.1 Outline of this approach

The basic theory behind this approach was to slow the acceleration just as had been done in testing. Instead of using a maximum step, the input is ramped through the use of a low pass filter. The output of the low pass filter then became a set-point for the Closed Loop Digital System whilst $c(t)$ became the input to the low pass filter (Figure 10.7).

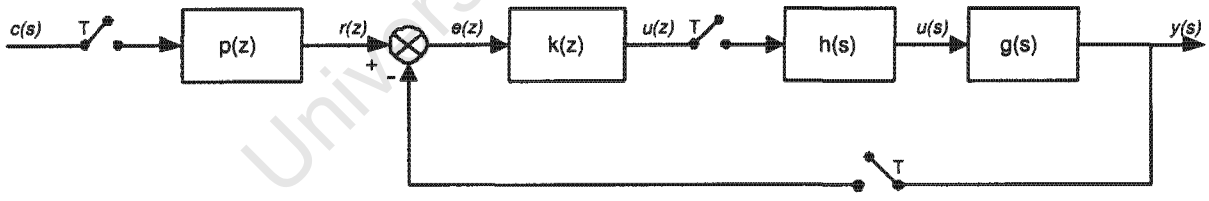


Figure 10.7: Addition of the pre-filter on the Closed Loop Digital System
This shows the inclusion of a pre-filter in the DSP controller for the control of the drive system.

[2] Requires a mechanical solution which would be difficult to implement within the F180 RoboSoccer rules.

University of Cape Town

10.4.2 Design of p(z)

Knowing that a pole at $s = -5$ was required[3] gives [33]

$$p(s) = \frac{1}{1 + 0.2s} \tag{10.17}$$

and since $T = 25ms$ it was determined that with a ZOH [34]

$$p(z) = \frac{0.1175}{z - 0.8825} \tag{10.18}$$

10.4.3 Implementation of p(z)

The algorithm for the pre-filter could then be calculated as follows[4]

$$p(z) = \frac{0.1}{z - 0.9} \frac{z^{-1}}{z^{-1}} = \frac{r(z)}{c(z)} \tag{10.19}$$

where

- $c(z)$ was now the input from the AI
- $r(z)$ was the input to the Closed Loop Digital System as before

which yielded

$$r_n = 0.1c_{n-1} + 0.9r_{n-1} \tag{10.20}$$

upon simplification. Since the duration of any step change $\gg$ than $25ms$, c_{n-1} could be replaced with c_n which helped to reduce the number of past values required. Now only

$$r_{n-1} = r_n \tag{10.21}$$

was required.

[3]The pole at $s = -5$ was only slightly slower than the Open Loop Drive System.

[4]The constants were rounded off as they would have had a negligible effect on the actual calculated values from the fixed point DSP.

University of Cape Town

10.4.4 Evaluation of Modified Closed Loop System

Following the inclusion of the pre-filter, the robot performed optimally since the pre-filter's dominant pole ensured both Drive Systems responded equally. Unfortunately, this not only helped in smoothing the acceleration but also smoothed the deceleration, thus preventing the robot from making the quick stops that had previously been possible. In all other aspects the performance of the robots exceeded design specifications.

10.5 Logic Bypass Controller

Since the inability to stop rapidly had only arisen because of the inclusion of the pre-filter, it was decided that this problem could be circumvented if the pre-filter was only used when the robot accelerated and bypassed when the robot was required to stop. This, in turn, necessitated the creation of a logic bypass controller.

University of Cape Town

10.5.1 Implementation

To this end the algorithm for the pre-filter was changed to include a conditional statement. If the input c_n was non zero the pre-filter ran as before but, if the input c_n was zero, the values for r_{n-1} were set to zero before Equation 10.20 was evaluated. In order to further increase this effect, within the same conditional statement, the values for u_{n-1}used in Equation 10.13 were set to zero and the H-Bridge was disabled thereby shorting the DC geared motors.

The final form of the code that was implemented is presented in Appendix B.6 where $\beta = 0.7$ and $K = 24$.[5]

10.5.2 Evaluation

A complete series of tests that were done to determine both dynamic and steady state responses showed that the control was finally working as required. Running the code on the other four robots yielded the same results and even demonstrated that the robots performed in a synchronised manner when run simultaneously with the same set-points.

[5] A value of $2 \times K$ was used to compensate for code changes in moving to a closed loop control system.

10.6 Conclusion

Due to the use of a methodical approach and continuous testing the Drive Systems, with their control algorithms, enabled the robots to perform not only predictably but also in a repeatable manner thereby guaranteeing a set response to set-points from the AI system. However, in any given system, obviously not all movements of the robots can be predicted, for example, instances such as when the robots collided with each other or with the boundary walls.

University of Cape Town

Chapter 11

Concluding Remarks and Recommendations

University of Cape Town

11.1 Concluding Remarks

Although the UCT F180 RoboSoccer team did not perform adequately so as to allow autonomous soccer play for the 2003 RoboSoccer competition, the system as a whole had made tremendous progress over the last 18 months.

The three key components that were initially described in the Introduction are discussed separately in the following sub-sections.

11.1.1 The Robots

The design of the robots benefitted from two prototypes before the final model was made for the 2003 UCT F180 RoboSoccer campaign. As a result, this area of research proved to be the most successful component that was actually tested.

The Turtle design used for the final version of the robots has intrinsic limitations which, in the long term, will hamper the progress of this research within UCT. In the interim though the robots perform reliably with a high degree of repeatability which, given the stages of development of the other two key components, builds in a degree of predictability in the movement and control of the robots.

Accordingly, through the use of predictive algorithms, the AI should be able to control the robots with limited feedback from the vision system. Although this is obviously not ideal, it does help simplify the overall system.

At this stage there are five complete Production Model robots and one semi-functioning Prototype Two robot which could make up two teams of three robots that could be used for continuing development of the other key components.

11.1.2 The AI / Global Controller

The AI or Global Controller was implemented by two undergraduate thesis students over the period of this research. Unfortunately neither of these students was able to really test the functioning of their research.

Some important areas within this component included the design and implementation of serial communication and the implementation of a joystick controller. Both these components worked flawlessly and should ideally remain unchanged in the short term.

Although the AI component was untested in 2003, the level of research undertaken in 2003 should provide a solid foundation for future research in this field.

The only area of concern that remains is the communication between the Vision System and the Global Controller. Although there is an underlying network layer built into both the systems, more research will be required to improve on the current system.

11.1.3 The Vision System

This was the last of the key components and is also the weakest at this stage. Unfortunately, several factors led to the final implementation of the Vision System being unsuccessful.

Successful components of this system which have developed from the research done leading up to the competition in 2003 include:

- near perfect lighting conditions
- a marker system which is easy to discriminate and identify within the vision system software
- the choice of a firewire web camera

University of Cape Town

11.2 Recommendations for Future Research

The factor that will be most significant in influencing the progress of this research is the need for an increased number of researchers at both undergraduate and postgraduate level. Ultimately, it is only through such increased participation and interest that the UCT F180 RoboSoccer campaign will be driven onto greater heights and eventual success.

The areas of research that should be considered for each of the key components in the short term are discussed below.

11.2.1 The Robots

The robots require the least research in the short term, but would be improved through further development of the kicker / dribbling mechanisms.

Although not essential, a development that would prove helpful would be the design of a battery charger for the NiCad batteries used by the robots.

The last recommendation for the robots is that another team of five robots should be built. This will enable testing of the overall system as if in competition conditions. It should also be considered that if a more complete set-up was available more researchers might be attracted to this field of research.

11.2.2 The AI / Global Controller

As this area of research remains untested, the work done until this point should first be tested upon completion of the Vision System and then, where required, the AI can be modified to improve the teams performance.

As the most work remains to be done on the Vision System, the underlying network protocol and implementation should be handled here, with consideration given to the structure of data that would be produced by Matlab.

11.2.3 The Vision System

As this area of research is still under development, the hardware, as used for the 2003 competition, should be re-used and a robust Vision System designed around this. The

University of Cape Town

current hardware should be capable of around 8 frames per second which would provide a good starting point for the Global Controller to be tested.

Once a robust Vision System is implemented, the limitations of the firewire web camera and computer can be revisited and better hardware purchased specifically for this purpose. The idea behind this is that, with the inclusion of new hardware, the existing Vision System should be able to function at higher frame rates which would lead to better real-time control from the Global Controller.

University of Cape Town

Bibliography

[1] "Official robocup small size league." http://www-2.cs.cmu.edu/Ebrettb/robocup/, Accessed 10 February 2002.

[2] "Robocup f180 rules repository." http://www.itee.uq.edu.au/ wyeth/F180Rules/index.htm, Accessed 10 February 2002.

[3] J. Thomas, A. Peel, "RooBots." http://www.cs.mu.oz.au/robocup/index.html, Accessed 10 February 2002.

[4] L. Wilson, C. Williams, J. Yance, J. Lew, R. L. Williams II, P. Gallina, "Design and Modeling of a Redundant Omni-directional RoboCup Goalie." zen.ece.ohiou.edu/ robocup/papers/mech/65.pdf, Accessed 10 February 2004.

[5] G. Anderson, C. Chang, D. Chung, P. Dingle, L. Evansic, H. Law, S. Richardson, J. Roberts, K. Sterk, J. Yim, *2003 Cornell RoboCup Documentation - Mechanical Design.* Cornell University, 2003.

[6] G. McPhillips, "Design of a line-following robot." Undergraduate Mechatronics Assignment, University of Cape Town, 2001.

[7] G. Wyeth, B. Browning, A. Tews, "UQ RoboRoos: On-going Design of a Robot Soccer Team." www.itee.uq.edu.au/ wyeth/RoboRoos/robocup99.pdf, Accessed 10 February 2004.

[8] M. Veloso, P. Stone, K. Han, S. Achim, "Prediction, Behaviors, and Collaboration in a Team of Robotic Agents." http://www.cs.cmu.edu/ mmv, pstone, kwunh, Accessed 10 February 2004.

[9] P. J. McKerrow, *Introduction to Robotics.* http://www.awl.com.au: Addison-Wesley, 1991.

University of Cape Town

[10] "Cornell University Robocup 2003." Highlights Video, Accessed 10 February 2003. http://robocup.mae.cornell.edu/.

[11] S. A. Williams, "Transwheel Robot." Undergraduate Thesis, University of Cape Town, 2003.

[12] S. T. Marais, "Motor Housing." TuteBot, 2002.

[13] Spectrum Digital Incorporated, www.spectrumdigital.com, *F243 DSK Technical Reference*, June 1999.

[14] Texas Instruments Incorporated, http://www.ti.com, *Octal 8-Bit Digital-to-Analog Converters (Rev. E)*, April 1997.

[15] Maxim, http://www.maxim-ic.com, *MAX6816, MAX6817, MAX6818 +/-15kV ESD-Protected, Single/Dual/Octal, CMOS Switch Debouncers.*

[16] Intersil, http://www.intersil.com, *HIP4081A - 80V/2.5A Peak, High Frequency Full Bridge FET Driver*, November 1996.

[17] VISHAY, http://www.vishay.com, *SUB75N03-04 N-Channel 30-V (D-S) 175C MOSFET*, June 2001.

[18] G. Danz, "HIP4080 and HIP4081 Power-up Application," Tech. Rep. TB321, Intersil, 1993.

[19] D. R. Moeller, *Open Source Motor Control.* http://www.robot-power.com, 2001. Accessed 10 February 2004.

[20] Fairchild Semiconductor Corporation, http://www.fairchildsemi.com, *74HC14 Hex Inverting Schmitt Trigger*, February 1999.

[21] Fairchild Semiconductor Corporation, http://www.fairchildsemi.com, *74HC86 Quad 2-Input Exclusive OR Gate.*

[22] Fairchild Semiconductor Corporation, http://www.fairchildsemi.com, *74HC32 - Quad 2-Input OR Gate.*

[23] Texas Instruments Incorporated, http://www.ti.com, *TPIC0107B PWM CONTROL INTELLIGENT H-BRIDGE*, April 2002. SLSI067A.

University of Cape Town

[24] Linx Technologies, http://www.linxtechnologies.com, *HP SERIES-II RECEIVER MODULE DESIGN GUIDE*, 1999.

[25] Linx Technologies, http://www.linxtechnologies.com, *HP SERIES-II TRANSMITTER MODULE DESIGN GUIDE*, 1999.

[26] Linx Technologies, http://www.linxtechnologies.com, *Application Note AN-00500 - Antennas: Design, Application, Performance*, 1997.

[27] National Semiconductor Corporation, http://www.national.com, *LM7805 Linear regulator.*

[28] Texas Instruments, http://www.ti.com, *TMS320F243/F241/C242 DSP Controllers Reference Guide - System and Peripherals*, January 2000.

[29] E. J. Luckin, "Soft Computing for Strategic Play in Robot Soccer." Undergraduate Thesis, University of Cape Town, 2002.

[30] M. Beyers, "Artificial Intelligence and Joystick Software Control for a Robot Soccer System." Undergraduate Thesis, University of Cape Town, 2003.

[31] M. Comline, "RoboSoccer Vision System." Undergraduate Thesis, University of Cape Town, 2002.

[32] J. G. Loomis, J. D. Palmer, P. Pandit, "Performance development of a real time vision system," Master's thesis, Cornell University, 2003.

[33] M. Braae, *Control Theory for Electrical Engineers.* UCT Press, Cape Town, 1994.

[34] M. Braae, *Control Engineering - 2.* UCT Press, Cape Town, 1996.

[35] J. Ilett, "How To Use Intelligent L.C.D.s - Part One," *Everyday Practical Electronics*, February 1997.

[36] J. Ilett, "How To Use Intelligent L.C.D.s - Part Two," *Everyday Practical Electronics*, March 1997.

University of Cape Town

Appendix A

Background and Code for the DSP Controller

University of Cape Town

A.1 Interrupt levels and configuration

On the TI DSP Controller, there are six interrupt levels which can be used. Each of these is able to be masked[1] and configured to a specific peripheral device. Their configurations are outlined below.

Interrupt level 1 Masked

Interrupt level 2

As this was the highest priority interrupt utilised, it was used in the generation of the pulse width modulation (PWM) that drove the DC geared motors and the DC motors. As such, this interrupt was configured as a Timer 1 period interrupt (TPINT1). The length of the period was configured so that the carrier frequency of the PWM was initially 4.95 KHz, whilst in the final implementation of the software the frequency was changed to 2.475KHz.

[1] As a result of being masked the interrupt was disabled.

Interrupt level 3

Although not initially used, this was finally configured as a Timer 2 period interrupt (TPINT2). In this instance, the Timer 2 was used for the controller as described in Chapter 10. Depending on the requirement of the controller, the period for Timer 2 was adjusted to be between $100\,ms$ and $25\,ms$.

Interrupt level 4

Masked

Interrupt level 5

In order for the masked interrupts to be used for the Serial Communication Interface (SCI), the SCI receiver interrupt (RXINT) could either be configured to be Interrupt level 1 or Interrupt level 5. Since both the Timers had to have a higher priority, the Interrupt level 5 was chosen for the SCI receiver interrupt. The SCI was configured as outlined in Section 8.2.3. The SCI receiver interrupt was triggered when data was received into the SCI receiver buffer thereby indicating that data was available for use.

Interrupt level 6

As the last function requiring an interrupt, the external interrupts that were used for the wheel encoders were assigned to this Interrupt level. It would have been preferable to assign the external interrupts to a higher level than the RXINT but, as was the case with the RXINT, the only other level with which they could be associated was Interrupt level 1, which would have given them a higher priority than the Timers and, accordingly, this option was not chosen.

Unlike the other Interrupts, two peripheral interrupts needed to be assigned to Interrupt level 6. These were, respectively, XINT1 and XINT2 both of which were external interrupt pins. Theoretically it would have been possible to assign more than one peripheral interrupt to any of the masked Interrupts, however, for this to function, the individual peripherals would have needed to be polled in order to establish which peripheral interrupt had been triggered.

University of Cape Town

A.2 Code implementation

A.2.1 Code Convention

The following code conventions were used:

- FULL CAPS represented internal registers, for example **CMPR1*
- a single CAP represented a control system variable with either an r, l or d being used to indicate right, left or kicker
- lower case variables represented counters or temporary storage

A.2.2 Timer 1 period interrupt (TPINT1)

In order for the duty cycle of the PWM to be updated, the Timer 1 period interrupt was used. Once the interrupt was triggered, the new set-points for the Right Drive System *(Urn)*, Left Drive System *(Uln)* and Kicker *(Rdn)* were loaded into relevant the compare registers **CMPRx*. The code used was as follows:

```
*CMPR1 = Urn;
*CMPR2 = Uln;
*CMPR3 = Rdn;
```

A.2.3 Timer 2 period interrupt (TPINT2)

Once the period of Timer 2 had been set and the Timer started, this Interrupt would be executed at the end of every period and, as such, was ideal to both sample the speed of the wheels and compute a new input to the Drive System.

The code for this feature can be found in Appendix B.6.

A.2.4 SCI receiver interrupt (RXINT)

In order to ensure successful serial communication the transmitter and receiver had to use the same protocol. The details of the protocol, required settings and command structure are dealt with in Section 8.2.

University of Cape Town

Since the serial communication was uni-directional, the DSP was only required to receive data. Although not strictly required, as no other peripheral was assigned to Interrupt 5, the receive buffer interrupt vector *(SCIRXST -> rxrdy)* was polled before reading the buffer.

```
if ((SCIRXST -> rxrdy) == 1)
```

Each command sent to the DSP consisted of five bytes, however, since only a single byte of data could be received by the buffer **SCIRXBUF* at a time, an array *rtx[n]* was created in code.

```
rtx[4] = rtx[3];
rtx[3] = rtx[2];
rtx[2] = rtx[1];
rtx[1] = rtx[0];
rtx[0] = *SCIRXBUF;
```

Thereafter, once the array had been updated, a simple pattern match was performed to check if a new instruction (Section 8.2.4) had indeed been received. x is the robot number (1 to 5)

```
if ((rtx[4] == 255) && (rtx[3] == x) && (rtx[2] == rtx[0]))
```

If the array matched the format of an instruction, the relevant variable was updated from *rtx[1]* and suitably scaled. The commands are shown in Table 8.1.

```
if (rtx[2] == 15)
{
 Cln = (rtx[1])-128;
 Cln = Cln*5;
 Cln = Cln>>5;
}
if (rtx[2] == 33)
{
 Crn = (rtx[1])-128;
```

University of Cape Town

```
 Crn = Crn*5;
 Crn = Crn>>5;
}
if (rtx[2] == 204)
{
 Rdn = rtx[1]*4;
 if (rtx[1] == 128) PBDATDIR -> En3 = 0; /*disable kicker pwm drive*/
 else PBDATDIR -> En3 = 1;               /*enable kicker pwm drive */
}
if (rtx[2] == 240) run(rtx[1]);  /*enable or disable all pwm drives*/
```

Once a receive error occurred, the RXINT was disabled. In order to prevent any further loss of data, the SCI had to be reset in order to clear the buffer and enable the RXINT. Since errors occurred along with data being available on the receive buffer **SCIRXBUF*, the *(SCIRXST -> rxerror)* could be polled so as to check for errors and the SCI software reset *(SCICTL1 -> swreset)* could then be reset if required.

```
if ((SCIRXST -> rxerror) == 1)
{
 SCICTL1 -> swreset = 0;
 SCICTL1 -> swreset = 1;
}
```

A.2.5 External interrupts (XINT1 and XINT2)

Due to the lack of External interrupts, the hardware implementation outlined in Section 6.5.3 was used as a means of compensation. This meant that there were six external interrupts available, that is, three for each External interrupt. In order to simplify the code, the actual inputs *((PADATDIR-> Sensor3) and (PADATDIR-> Sensor6))* from each sensor were polled to check for changes in state as opposed to polling XINT1 and XINT2. Once a change of state was found, the velocity count *(sen1 or sen4)* was either incremented or decremented according to the last set-point of the motors *(Urnn or Ulnn)* and the state *(S1 or S2)* updated accordingly.

```
if ((PADATDIR-> Sensor3) != S1)
```


University of Cape Town

```
{
 if (Urnn >= 0) sen1++;
 else sen1--;
 S1 = (PADATDIR-> Sensor3);
}
if ((PADATDIR-> Sensor6) != S2)
{
 if (Ulnn >= 0) sen4++;
 else sen4--;
 S2 = (PADATDIR-> Sensor6);
}
```

A.2.6 The main routine

As the robot required no code to be implemented in the main routine in order for the robot to function, the routine is simply a continuous loop as shown below.

```
for(;;)
{}
```

For details on the implementation of the Intelligent LCD follows directly from the literature [35, 36].

University of Cape Town

Appendix B

Control Background and Data

University of Cape Town

B.1 Speed Conversion

Given that the Drive System had a wheel with a diameter of 60 mm the actual speed in ms^{-1} could be calculated as follows

$$Speed = C_x \times \frac{\pi}{2x} \left[ms^{-1}\right] \tag{B.1}$$

where

- C_x was the number of counts from the wheel encoder in x ms period

Therefore, given 20 C_{25}, the speed was 20 counts within a 25 ms sample period. Through the use of Equation B.1 the actual speed could be calculated to be 1.26 ms^{-1}.

B.2 First Order Response to a Step Input

When a first order system is perturbed by a step response, the transients decay within a period defined by the time constant T. The response to a step input $u(s)$ where [33]

$$u(s) = \frac{B}{s} \tag{B.2}$$

is simply

$$y(t) = BA(1 - e^{\frac{-t}{T}}) \left[ms^{-1}\right] \tag{B.3}$$

where

- Final value $y(\infty) = BA$
- Initial value $y(0) = 0$
- The time taken for $y(t)$ to reach 63% of BA gives the value for the time constant T.

B.3 Step Test 1

Each Drive System was stepped from zero velocity to maximum velocity for both forward and backward motion. The Step Test was done on the Left and Right Drive Systems independent of the robot. The time constant T was found using the time t for $y(t)$ to reach 63% of the final velocity. The DSP controller ran the Step Test as outlined in Figure B.1.

B.3.1 Flow chart

The flow chart for the step test program on the DSP controller is outlined in Figure B.1.

B.3.2 Step Test results for each Drive System

The results for each of the four step tests done in Step Test 1 are presented as follows:

- Left Drive System forward motion in Figure B.1
- Right Drive System forward motion in Figure B.2
- Left Drive System reverse motion in Figure B.3
- Right Drive System reverse motion in Figure B.4

University of Cape Town

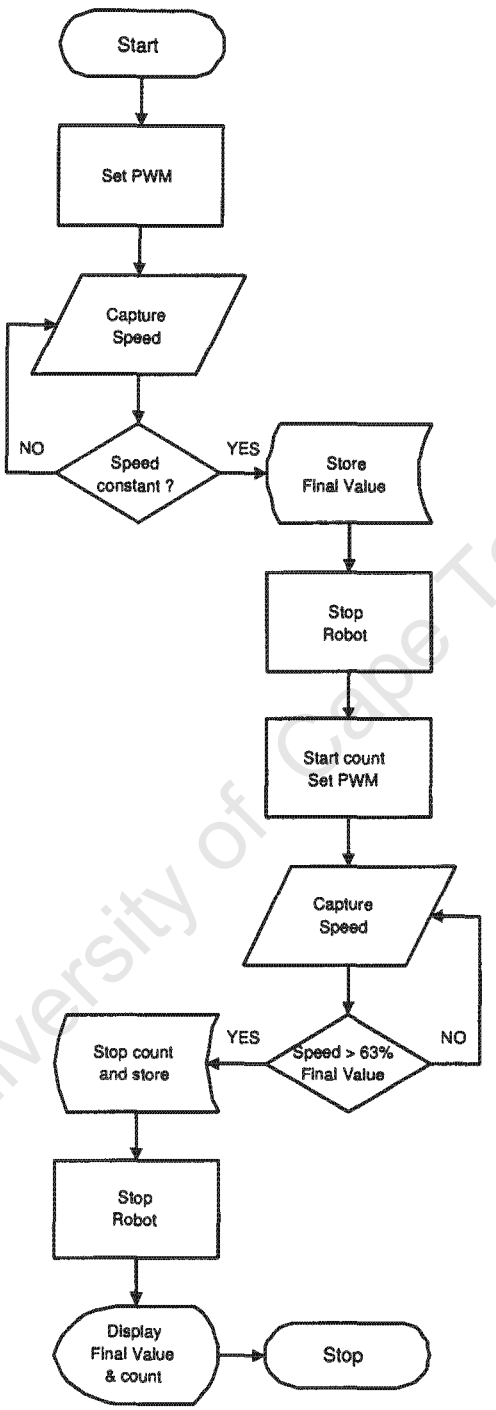


Figure B.1: Flow chart for Step Test 1

This shows the routine the robot executed for Step Test 1.

University of Cape Town

Sample Time (ms)	Max Count	Count to 63%	Time Constant (ms)
100	96	2	200
25	24	7	175
12.5	13	7±2	Err

Table B.1: Forward motion Step Test 1 results for Left Drive System

Sample Time (ms)	Max Count	Count to 63%	Time Constant (ms)
100	92	2	200
25	23	8	200
12.5	12	8±2	Err

Table B.2: Forward motion Step Test 1 results for Right Drive System

Sample Time (ms)	Max Count	Count to 63%	Time Constant (ms)
100	86	2	200
25	21	7	175
12.5	11	7±2	Err

Table B.3: Reverse motion Step Test 1 results for Left Drive System

Sample Time (ms)	Max Count	Count to 63%	Time Constant (ms)
100	92	2	200
25	23	8	200
12.5	12	8±2	Err

Table B.4: Reverse motion Step Test 1 results for Right Drive System

University of Cape Town

B.4 Step Test 2

Each Drive System was stepped from zero velocity to maximum velocity for forward motion only. The Step Test was done on the Left and Right Drive Systems simultaneously whilst driving the robot over the playing surface. The time constant T was found by taking the time t for $y(t)$ to reach a count of 20.0 in a 25 ms period and then calculating the time constant T from Equation 10.2. The DSP controller ran the Step Test as outlined in Figure B.2.

B.4.1 Modified Flow chart

The flow chart for the modified step test program on the DSP controller is outlined in FigureB.2.

B.4.2 Step Test results for each Drive System

The data for Step Test 2 for all ten step tests is shown in Table B.5. The average times were then used to determine the time constant for each Drive System in Step Test 2.

Max Speed Left	Count Left	Max Speed Right	Count Right
21	16	20	22
22	16	21	20
22	17	21	21
21	16	20	20
22	15	21	19
22	14	20	19
22	15	21	19
21	17	20	22
21	15	20	20
21	16	20	21

Table B.5: Forward motion Step Test 2 results

University of Cape Town

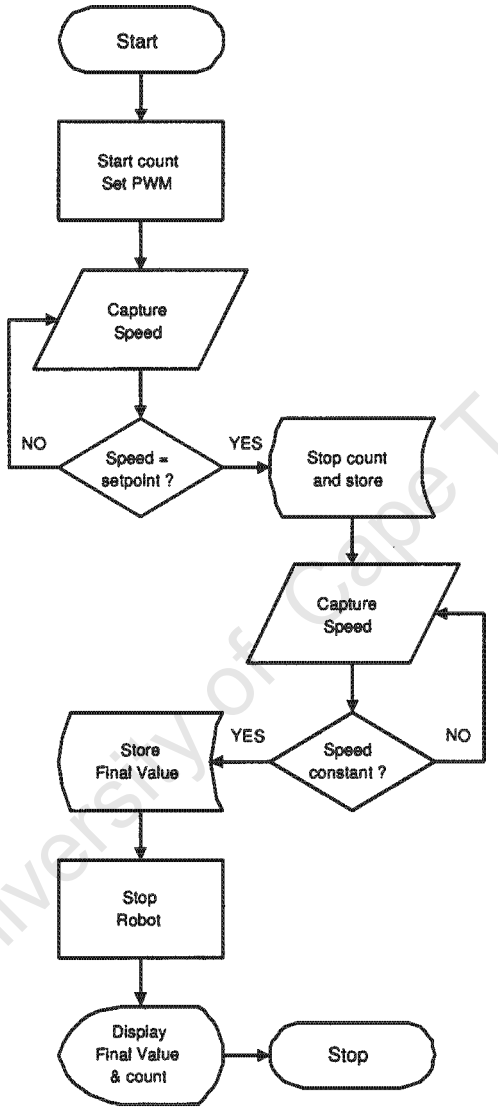


Figure B.2: Flow chart for Step Test 2 and 3

This shows the routine the robot executed for Step Test 2 and 3.

University of Cape Town

B.5 Step Test 3

Each Drive System was again simultaneously stepped from zero velocity to maximum velocity for forward motion only whilst driving the robot over the playing surface. The time constant T was found by taking the time t for $y(t)$ to reach 90% of the final velocity. The DSP controller ran the Step Test as outlined in Figure B.2 but using different set-points.

B.5.1 Step Test results for each Drive System

The data for Step Test 3 for all ten step tests is shown in Table B.6. The average times were then used to determine the time constant for each Drive System in Step Test 3.

Max Speed Left	Count Left	Max Speed Right	Count Right
21	14	20	14
22	14	20	15
21	13	20	15
21	14	20	14
22	14	20	15
21	13	20	15
22	13	20	14
22	14	20	14
22	14	20	14
21	13	20	15

Table B.6: Forward motion Step Test 3 results

University of Cape Town

B.6 Controller Code

The controller code below was implemented in Interrupt level 3 on the DSP controller (Section A.2.3). The code followed directly from the theory presented in Chapter 10.

As the Interrupt was triggered, both the right and left speeds *(sen1 and sen4)* were read into the control variables *rspeed* and *lspeed* respectively. Both speeds were then reset.

```
rspeed = sen1;
```

```
sen1 = 0;
lspeed = sen4;
sen4 = 0;
```

The logic controller was then implemented; if either set-point for the Left or Right Drive Systems *(Cln or Crn)* was zero the respective previous control values were set to zero *(Rlnn & Ulnp or Rrnn & Urnp)* and the respective H-Bridge disabled *(PBDATDIR -> En1 or PBDATDIR -> En2)*. If the set-points were non-zero, the H-Bridges were enabled.

```
if (Crn == 0)
{
 PBDATDIR -> En1 = 0;
 Rrnn = 0;
 Urnp = 0;
}
else PBDATDIR -> En1 = 1;
if (Cln == 0)
{
 PBDATDIR -> En2 = 0;
 Rlnn = 0;
 Ulnp = 0;
}
else PBDATDIR -> En2 = 1;
```

University of Cape Town

Once the set-points were established, the pre-filter could be implemented. It should be noted that if, from above, the set-points for the Drive Systems were zero their respective outputs from the pre-filter would have been zero. If this was not the case, the pre-filter used its past values to find a new set-point *(Rln and Rrn)* for the comparator. The past values *(Rlnn and Rrnn)* were then stored for the next sample time.

```
Rrn = (0.1*Crn) + (0.9*Rrnn);
Rrnn = Rrn;
Rln = (0.1*Cln) + (0.9*Rlnn);
Rlnn = Rln;
```

The error for each Drive System *(Elnn and Ernn)* was then calculated from their respective speed and set-point values from the pre-filter.

```
Ernn = Rrn - rspeed;
Elnn = Rln - lspeed;
```

Since the error and past values for each Drive System were known, the new set-points for the Drive Systems could be calculated using the PI controller.

```
Urnn = Urnp + (24*Ernn) - (17*Ernp);
Ulnn = Ulnp + (24*Elnn) - (17*Elnp);
```

In order to ensure that the set-points remained within the limits of the Drive Systems, these were changed as required and the range changed from $-511 \rightarrow 512$ to $0 \rightarrow 1023$ to suit the Drive Systems.

```
if (Urnn > 512) Urnn = 512;
if (Urnn < -511) Urnn = -511;
if (Ulnn > 512) Ulnn = 512;
if (Ulnn < -511) Ulnn = -511;
Urn = Urnn + 511;
Uln = Unll + 511;
```

Finally, the past values for error and set-point were stored for use in the next control loop.

```
Urnp = Urnn;
Ulnp = Ulnn;
Erpn = Ernn;
Elnp = Elnn;
```

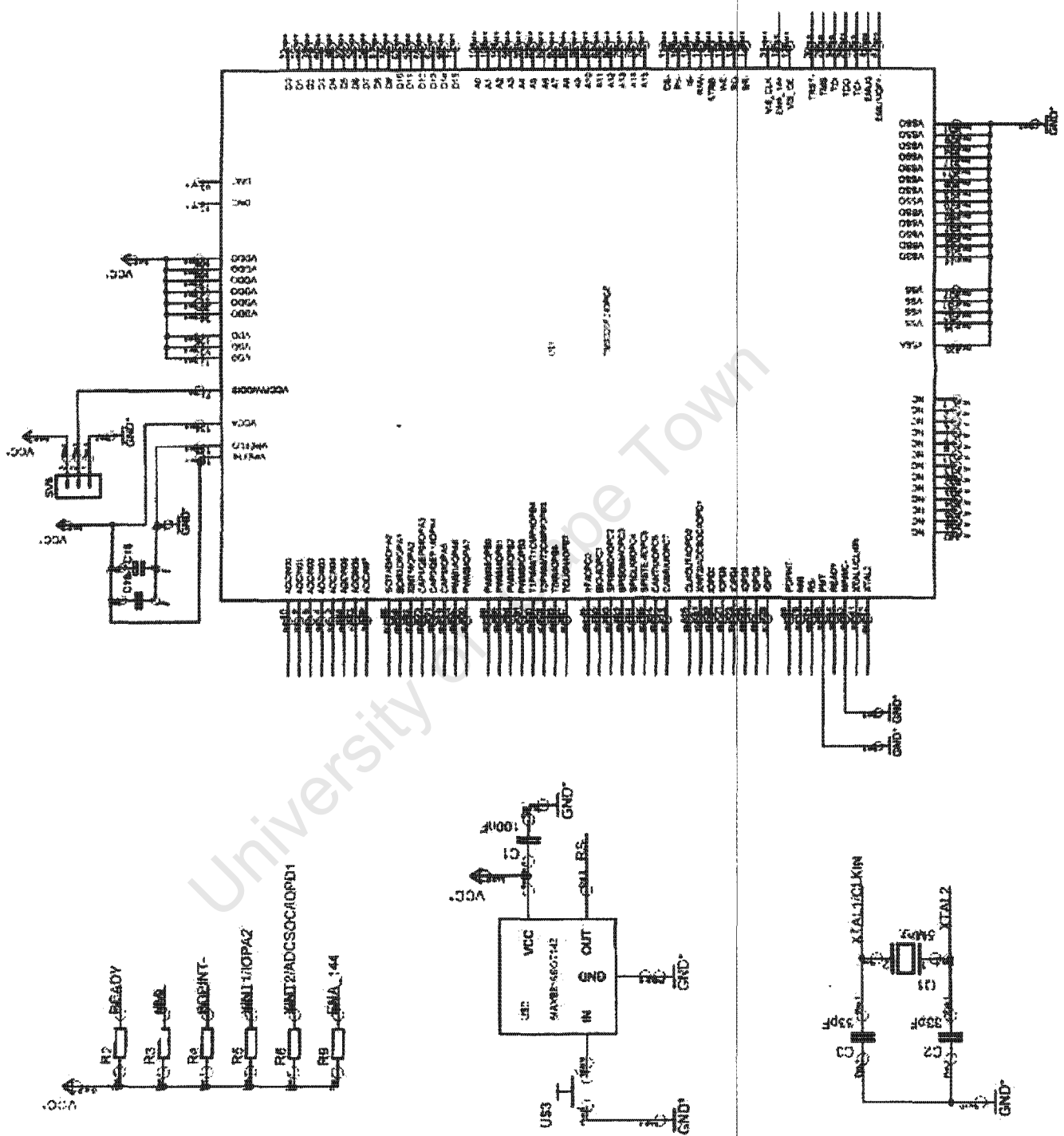
University of Cape Town

Appendix C

Schematics

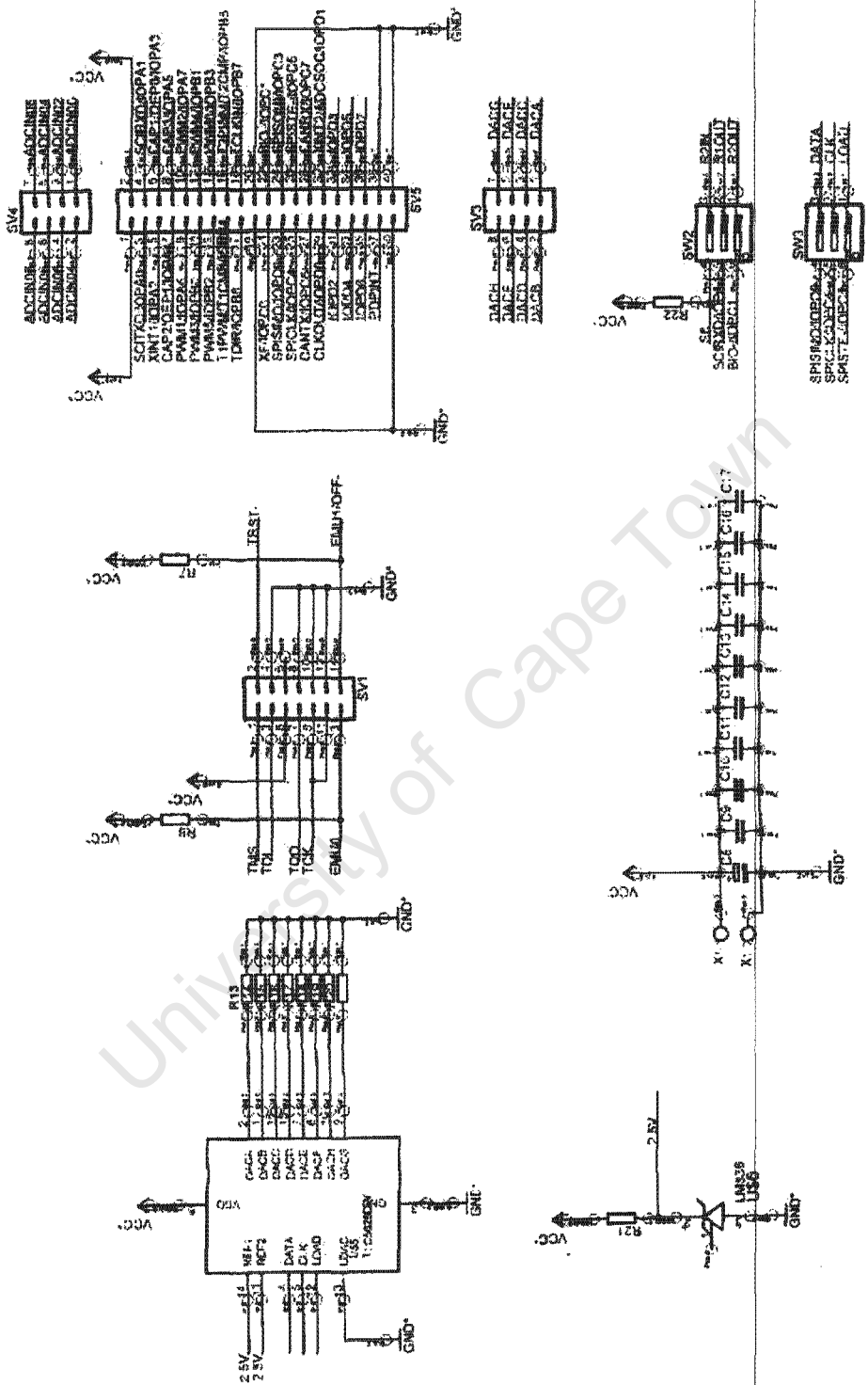
The following pages contain the schematics for the DSP controller, interface board and H-bridge as used in the prototype and competition robots.

University of Cape Town



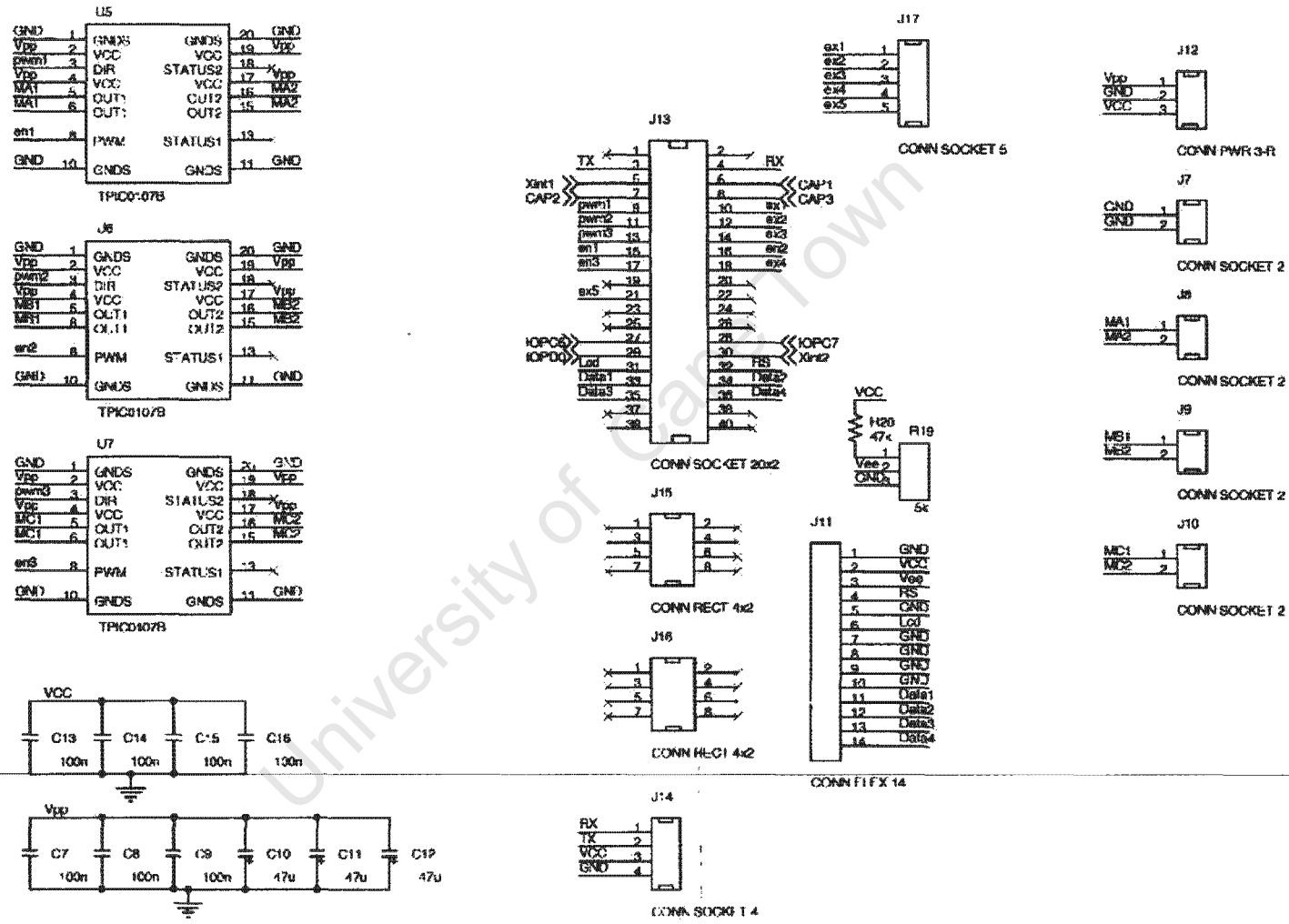
DSP controller page 1 of 2

University of Cape Town

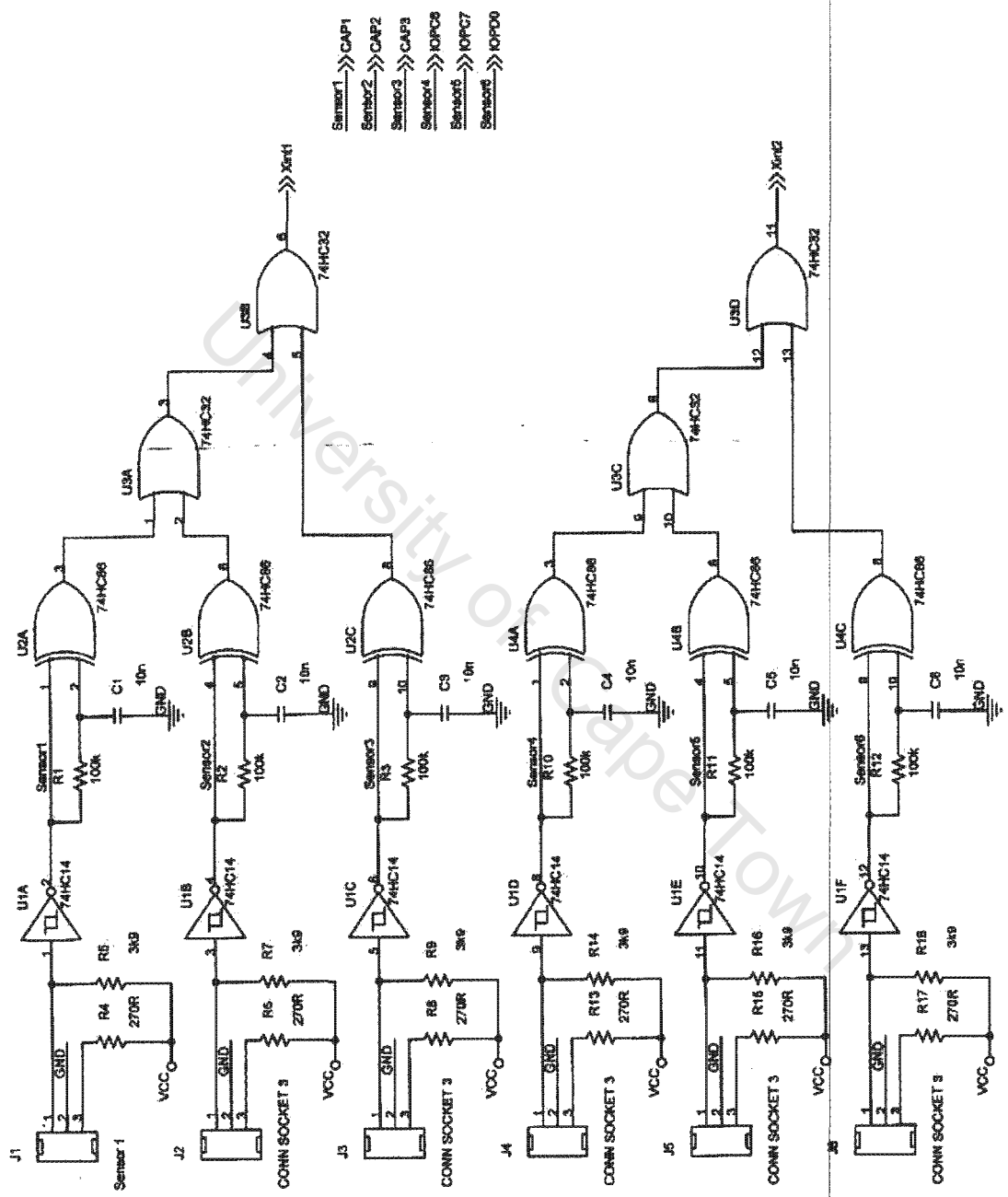


DSP controller page 2 of 2

University of Cape Town

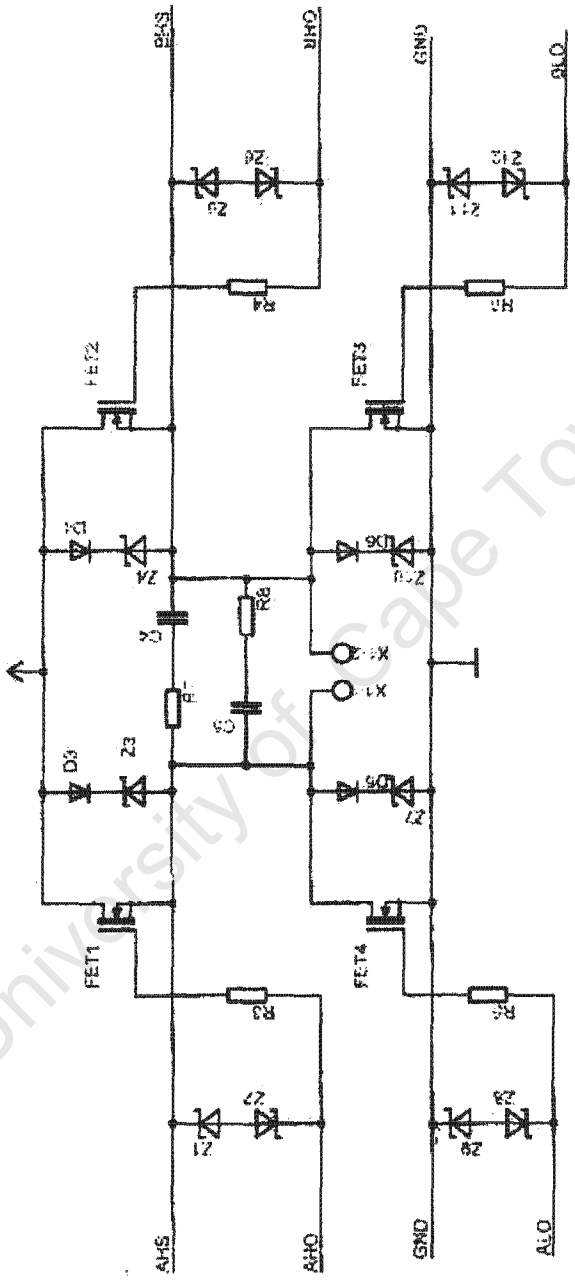


University of Cape Town



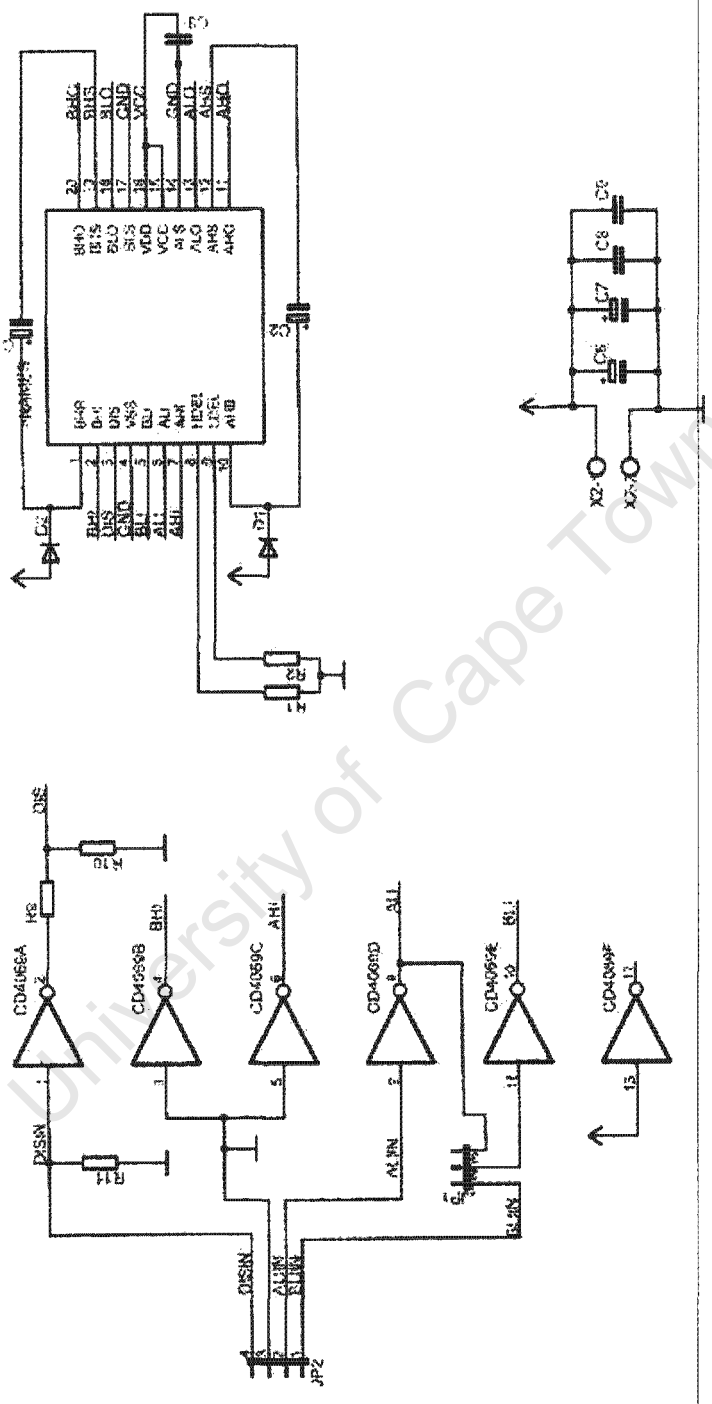
Interface board page 2 of 2

University of Cape Town



H-bridge board page 1 of 2

University of Cape Town



H-bridge board page 2 of 2

University of Cape Town

Appendix D

Code

The following pages contain the entire source code for the DSP controller board used in the competition robots.

University of Cape Town

D.1 Main Source Code

```
/* Final Code for F180 RoboSoccer Robot */
/* Writen By Graeme McPhillips */
#include "portd.h"
#include "portc.h"
#include "portb.h"
#include "porta.h"
#include "serial.h"
volatile unsigned int *ACTR = (volatile unsigned int *) 0x7413;
volatile unsigned int *DBTCON = (volatile unsigned int *) 0x7415;
volatile unsigned int *COMCON = (volatile unsigned int *) 0x7411;
volatile unsigned int *GPTCON = (volatile unsigned int *) 0x7400;
volatile unsigned int *T1CON = (volatile unsigned int *) 0x7404;
volatile unsigned int *T2CON = (volatile unsigned int *) 0x7408;
volatile unsigned int *OCRA = (volatile unsigned int *) 0x7090;
volatile unsigned int *OCRB = (volatile unsigned int *) 0x7092;
volatile unsigned int *T1PR = (volatile unsigned int *) 0x7403;
volatile unsigned int *T1CNT = (volatile unsigned int *) 0x7401;
volatile unsigned int *T2PR = (volatile unsigned int *) 0x7407;
volatile unsigned int *T2CNT = (volatile unsigned int *) 0x7405;
volatile unsigned int *EVIMRA = (volatile unsigned int *) 0x742C;
volatile unsigned int *EVIMRB = (volatile unsigned int *) 0x742D;
volatile unsigned int *EVIMRC = (volatile unsigned int *) 0x742E;
volatile unsigned int *EVIFRA = (volatile unsigned int *) 0x742F;
volatile unsigned int *EVIFRB = (volatile unsigned int *) 0x7430;
volatile unsigned int *IMR = (volatile unsigned int *) 0x0004;
volatile unsigned int *IFR = (volatile unsigned int *) 0x0006;
```

```
volatile unsigned int *SPICCR = (volatile unsigned int *) 0x7040;
volatile unsigned int *SPICTL = (volatile unsigned int *) 0x7041;
volatile unsigned int *SPISTS = (volatile unsigned int *) 0x7042;
volatile unsigned int *SPIBRR = (volatile unsigned int *) 0x7044;
volatile unsigned int *SPIRXEMU = (volatile unsigned int *) 0x7046;
volatile unsigned int *SPIRXBUF = (volatile unsigned int *) 0x7047;
volatile unsigned int *SPITXBUF = (volatile unsigned int *) 0x7048;
volatile unsigned int *SPIDAT = (volatile unsigned int *) 0x7049;
volatile unsigned int *SPIPRI = (volatile unsigned int *) 0x704F;
volatile unsigned int *SCICCR = (volatile unsigned int *) 0x7050;
volatile unsigned int *SCIHBAUD = (volatile unsigned int *) 0x7052;
volatile unsigned int *SCILBAUD = (volatile unsigned int *) 0x7053;
volatile unsigned int *SCIPRI = (volatile unsigned int *) 0x705F;
volatile unsigned int *SCITXBUF = (volatile unsigned int *) 0x7059;
volatile unsigned int *SCIRXBUF = (volatile unsigned int *) 0x7057;
volatile unsigned int *CMPR1 = (volatile unsigned int *) 0x7417;
volatile unsigned int *CMPR2 = (volatile unsigned int *) 0x7418;
volatile unsigned int *CMPR3 = (volatile unsigned int *) 0x7419;
volatile unsigned int *XINT1CR = (volatile unsigned int *) 0x7070;
volatile unsigned int *XINT2CR = (volatile unsigned int *) 0x7071;
volatile unsigned int *PIACKR1 = (volatile unsigned int *) 0x7015;
long int i,j;
int in;
int key_disp;
int key_dispms;
int key_displs;
int sen1,sen2,sen3,sen4,sen5,sen6,Crn,Cln,Rrnn,Rlnn,Rdn,Rrn,Rln,Uln,Urn,Urnn,Ulnn,Urnp,Ulnp,Ernn,Ernp,Elnn,Elnp;
int Thou,Hund,Ten,Unit;
int S1,S2,S4,S5,S7;
int lspeed,rspeed,speed,rotate,rmax,lmax;
int rtx[4];
/* Wait routine. */
int wait(int w)
        {
        for (i = 0; i<100; i++)
                {
                        for (j = 0; j<w ; j++)
                        {}
                }
        }
int run(int rr)
        {
                PBDATDIR -> En1 = rr;
                PBDATDIR -> En2 = rr;
                PBDATDIR -> En3 = rr;
        }
/* Move cursor*/
int Car_ret(int cr)
{
                                key_disp = cr;
                                Get_char();
                                PDDATDIR -> RS = 0;
```

University of Cape Town

```
                                PDDATDIR -> DATA = key_dispms;
                                wait(10);
                                PDDATDIR -> Lcd = 1;
                                wait(10);
                                PDDATDIR -> Lcd = 0;
                                wait(10);
                                PDDATDIR -> DATA = key_displs;
                                wait(10);
                                PDDATDIR -> Lcd = 1;
                                wait(10);
                                PDDATDIR -> Lcd = 0;
                                wait(10);
}
/* Split character for display */
void Get_char(void)
{
        key_dispms = 0;
        key_displs = 0;
        key_dispms = key_disp;
        key_dispms = key_dispms>>4;
        key_displs = key_disp;
        key_displs &= 0x0F;
}
/* Get Thousands, Hundreds, Tens and Units */
int Decimal(int dec)
{
        Thou = 0;
        Hund = 0;
        Ten = 0;
        Unit = 0;
        while (dec >= 1000)
        {
                Thou = Thou + 1;
                dec = dec - 1000;
        }
        while (dec >= 100)
        {
               Hund = Hund + 1;
               dec = dec - 100;
        }
        while (dec >= 10)
        {
               Ten = Ten + 1;
               dec = dec - 10;
        }
        Unit = dec;
}
/* Display keypad entry */
void Display_key(void)
{
                        wait(10);
                        PDDATDIR -> RS = 1;
```

University of Cape Town

```
                        wait(10);
                        PDDATDIR -> DATA = key_dispms;
                        wait(10);
                        PDDATDIR -> Lcd = 1;
                        wait(10);
                        PDDATDIR -> Lcd = 0;
                        wait(10);
                        PDDATDIR -> DATA = key_displs;
                        wait(10);
                        PDDATDIR -> Lcd = 1;
                        wait(10);
                        PDDATDIR -> Lcd = 0;
                        wait(10);
                        PDDATDIR -> RS = 0;
}
/* Clear display and home cursor */
void Clear(void)
        {
                wait(10);
                PDDATDIR -> RS = 0;
                wait(10);
                PDDATDIR -> DATA = 0x0;
                wait(10);
                PDDATDIR -> Lcd = 1;
                wait(10);
                PDDATDIR -> Lcd = 0;
                wait(10);
                PDDATDIR -> DATA = 0x1;
                wait(10);
                PDDATDIR -> Lcd = 1;
                wait(10);
                PDDATDIR -> Lcd = 0;
                wait(10);
        }
interrupt void Test1(void)
        {
        }
interrupt void GPT1_underflow(void)
        {
                *CMPR1 = Urn;
                *CMPR2 = Uln;
                *CMPR3 = Rdn;
                *EVIFRA |= 0x0200;
        }
interrupt void Test3(void)
        {
                if (S7 == 1)
                {
                        PCDATDIR -> Ex5 = 0;
                        S7 = 0;
                }
                else
```

University of Cape Town

```
                {
                        PCDATDIR -> Ex6 = 1;
                        S7 = 1;
                }
                rspeed = sen1;
                sen1 = 0;
                lspeed = sen4;
                sen4 = 0;
                if (Crn == 0)
                {
                        PBDATDIR -> En1 = 0;
                        Rrnn = 0;
                        Urnp = 0;
                }
                else PBDATDIR -> En1 = 1;
                if (Cln == 0)
                {
                        PBDATDIR -> En2 = 0;
                        Rlnn = 0;
                        Ulnp = 0;
                }
                else PBDATDIR -> En2 = 1;
                Rrn = (0.1*Crn) + (0.9*Rrnn);
                Rrnn = Rrn;
                Rln = (0.1*Cln) + (0.9*Rlnn);
                Rlnn = Rln;
                Ernn = Rrn - rspeed;
                Elnn = Rln - lspeed;
                Urnn = Urnp + (24*Ernn) - (17*Ernp);
                Ulnn = Ulnp + (24*Elnn) - (17*Elnp);
                if (Urnn > 512) Urnn = 512;
                if (Urnn < -511) Urnn = -511;
                if (Ulnn > 512) Ulnn = 512;
                if (Ulnn < -511) Ulnn = -511;
                Urn = Urnn + 512;
                Uln = Ulnn + 512;
                Urnp = Urnn;
                Ulnp = Ulnn;
                Ernp = Ernn;
                Elnp = Elnn;


                *EVIFRB |= 0x0001;
        }
interrupt void Test4(void)
        {
        }
interrupt void Test5(void)
        {
        if ((SCICTL2 -> txrdy) == 1)
        {
        }
```

University of Cape Town

```
        if ((SCIRXST -> rxrdy) == 1)
        {
                rtx[4] = rtx[3];
                rtx[3] = rtx[2];
                rtx[2] = rtx[1];
                rtx[1] = rtx[0];
                rtx[0] = *SCIRXBUF;
                if ((rtx[4] == 255) && (rtx[3] == 5) && (rtx[2] == rtx[0]))
                {
                        if (rtx[2] == 15)
                        {
                                Cln = (rtx[1])-128;
                                Cln = Cln*5;
                                Cln = Cln>>5;
                        }
                        if (rtx[2] == 33)
                        {
                                Crn = (rtx[1])-128;
                                Crn = Crn*5;
                                Crn = Crn>>5;
                        }
                        if (rtx[2] == 204)
                        {
                                Rdn = rtx[1]*4;
                                if (rtx[1] == 128) PBDATDIR -> En3 = 0;
                                else PBDATDIR -> En3 = 1;
                        }
                        if (rtx[2] == 240) run(rtx[1]);
                }
        }
        if ((SCIRXST -> rxerror) == 1)
        {
                SCICTL1 -> swreset = 0;
                SCICTL1 -> swreset = 1;
        }
}
interrupt void XINT2(void)
                {
                        if ((PADATDIR -> Sensor3) != S1)
                                {
                                        if (Urnn >= 0) sen1++;
                                        else sen1--;
                                        S1 = (PADATDIR -> Sensor3);
                                }
                        if ((PDDATDIR -> Sensor6) != S2)
                                {
                                        if (Ulnn >= 0) sen4++;
                                        else sen4--;
                                        S2 = (PDDATDIR -> Sensor6);
                                }
                        *XINT2CR |= 0x8000;
                        *XINT1CR |= 0x8000;
```

University of Cape Town

```
}
void main(void)
{
PDDATDIR -> DATADIR = 0xFC;
PCDATDIR -> DATADIR = 0x21;
PBDATDIR -> DATADIR = 0x75;
PADATDIR -> DATADIR = 0x41;
asm("   clrc INTM");
*OCRB = 0x031F;
*OCRA = 0x0547;
*IMR = 0x0036;
*IFR = 0xFFFF;
*XINT1CR = 0x0007;
*XINT2CR = 0x0007;
*XINT1CR |= 0x8000;
*XINT2CR |= 0x8000;
*SCICCR = 0x07;
SCICTL1 -> rxena = 0;
SCICTL1 -> txena = 0;
SCICTL1 -> swreset = 1;
SCICTL1 -> rxerrintena = 1;
*SCIHBAUD = 0x00;
*SCILBAUD = 0x81;
*SCIPRI = 0x78;
SCICTL2 -> txintena = 0;
SCICTL2 -> rxintena = 0;
*EVIMRA = 0x0201;
*EVIMRB = 0x0001;
*DBTCON = 0x06E0;
*ACTR = 0x0999;
*COMCON = 0x8300;
*T1PR = 1024;
*T1CON = 0xA802;
*T2PR = 0x1E84;
*T2CON = 0xB606;
*T1CON = 0xA842;
*T2CON = 0xB646;
PDDATDIR -> RS = 0;
PDDATDIR -> Lcd = 0;
PCDATDIR -> Ex5 = 1;
S7 = 1;
S1=0;
S2=0;
sen1=0;
sen4=0;
Urnp = 0;
Ulnp = 0;
Urn=512;
Uln=512;
Cln = 0;
Crn = 0;
Rrnn = 0;
```

University of Cape Town

```
                Rlnn = 0;
                Rdn=512;
                wait(100);
                rtx[0]=0;
                rtx[1]=0;
                rtx[2]=0;
                rtx[3]=0;
                rtx[4]=0;
                SCICTL1 -> rxena = 1;
                SCICTL2 -> rxintena = 1;

/*debug stuff*/
                rmax = 0;
                lmax = 0;

/* Set 8 bit mode */
                PDDATDIR -> RS = 0;
                for (in = 0; in<3; in++)
                {
                PDDATDIR -> DATA = 0x3;
                wait(5);
                PDDATDIR -> Lcd = 1;
                wait(30);
                PDDATDIR -> Lcd = 0;
                wait(10);
                }
/* Set 2 line 4 bit mode */

                PDDATDIR -> DATA = 0x2;
                wait(10);
                PDDATDIR -> Lcd = 1;
                wait(10);
                PDDATDIR -> Lcd = 0;
                wait(10);
                PDDATDIR -> DATA = 0x2;
                wait(10);
                PDDATDIR -> Lcd = 1;
                wait(10);
                PDDATDIR -> Lcd = 0;
                wait(10);
                PDDATDIR -> DATA = 0x8;
                wait(10);
                PDDATDIR -> Lcd = 1;
                wait(10);
                PDDATDIR -> Lcd = 0;
                wait(10);
/* cursor travel */
                PDDATDIR -> DATA = 0x0;
                wait(10);
                PDDATDIR -> Lcd = 1;
                wait(10);
                PDDATDIR -> Lcd = 0;
```

University of Cape Town

```
                wait(10);
                PDDATDIR -> DATA = 0x6;
                wait(10);
                PDDATDIR -> Lcd = 1;
                wait(10);
                PDDATDIR -> Lcd = 0;
                wait(10);
/* Display On */
                PDDATDIR -> DATA = 0x0;
                wait(10);
                PDDATDIR -> Lcd = 1;
                wait(10);
                PDDATDIR -> Lcd = 0;
                wait(10);
                PDDATDIR -> DATA = 0xC;
                wait(10);
                PDDATDIR -> Lcd = 1;
                wait(10);
                PDDATDIR -> Lcd = 0;
                wait(10);
                Clear();

                for(;;)
                        {
                                Car_ret(139);
                                key_dispms = 3;
                                Decimal(abs(lspeed));
                                key_displs = Thou;
                                Display_key();
                                key_displs = Hund;
                                Display_key();
                                key_displs = Ten;
                                Display_key();
                                key_displs = Unit;
                                Display_key();
                                Car_ret(203);
                                key_dispms = 3;
                                Decimal(abs(rspeed));
                                key_displs = Thou;
                                Display_key();
                                key_displs = Hund;
                                Display_key();
                                key_displs = Ten;
                                Display_key();
                                key_displs = Unit;
                                Display_key();
                        }
                }
```

University of Cape Town

D.2 Header Files

D.2.1 porta.h

```
/* Port A */
  volatile struct porta {
                        unsigned Other          :2;
                        unsigned Xint1          :1;
                        unsigned Sensor1        :1;
                        unsigned Sensor2        :1;
                        unsigned Sensor3        :1;
                        unsigned PWM1           :1;
                        unsigned Ex1            :1;
                        unsigned DATADIR        :8;
       };
  volatile struct porta *PADATDIR = (volatile struct porta *)    0x7098;
```

D.2.2 portb.h

```
/* Port B */
  volatile struct portb {
                        unsigned PWM3           :1;
                        unsigned Ex2            :1;
                        unsigned PWM5           :1;
                        unsigned Ex3            :1;
                        unsigned En1            :1;
                        unsigned En2            :1;
                        unsigned En3            :1;
                        unsigned Ex4            :1;
                        unsigned DATADIR        :8;
       };
  volatile struct portb *PBDATDIR = (volatile struct portb *)    0x709A;
```

D.2.3 portc.h

```
/* Port C */
  volatile struct portc {
                        unsigned Ex5            :1;
                        unsigned OTHER          :4;
                        unsigned SPISTE         :1;
                        unsigned Sensor4        :1;
                        unsigned Sensor5        :1;
                        unsigned DATADIR        :8;
       };
  volatile struct portc *PCDATDIR = (volatile struct portc *)    0x709C;
```

University of Cape Town

D.2.4 portd.h

```
/* Port D */
  volatile struct portd {
                        unsigned Sensor6        :1;
                        unsigned Xint2          :1;
                        unsigned Lcd            :1;
                        unsigned RS             :1;
                        unsigned DATA           :4;
                        unsigned DATADIR        :8;
        };
  volatile struct portd *PDDATDIR = (volatile struct portd *)    0x709E;
```

D.2.5 serial.h

```
/* Serial com*/
        volatile struct serial1 {
                        unsigned rxena                  :1;
                        unsigned txena                  :1;
                        unsigned sleep                  :1;
                        unsigned txwake                 :1;
                        unsigned res1                   :1;
                        unsigned swreset                :1;
                        unsigned rxerrintena            :1;
                        unsigned res2                   :1;

        };
        volatile struct serial2 {
                        unsigned txintena               :1;
                        unsigned rxintena               :1;
                        unsigned res1                   :4;
                        unsigned txempty                :1;
                        unsigned txrdy                  :1;

        };
        volatile struct serialrx {
                        unsigned res2                   :1;
                        unsigned rxwake                 :1;
                        unsigned pe                     :1;
                        unsigned oe                     :1;
                        unsigned fe                     :1;
                        unsigned brkdt                  :1;
                        unsigned rxrdy                  :1;
                        unsigned rxerror                :1;

        };
        volatile struct serial1 *SCICTL1 = (volatile struct serial1 *)          0x7051;
        volatile struct serial2 *SCICTL2 = (volatile struct serial2 *)          0x7054;
        volatile struct serialrx *SCIRXST = (volatile struct serialrx *)        0x7055;
```

University of Cape Town

Appendix E

Accompanying CD-ROM

This text is accompanied by a CD-ROM. Before navigating the CD-ROM, the user should first read the README file in the root directory, which will contain any last minute information. The CD-ROM will contain the following information:

- **Source Code**
  All source code that was produced can be found in the "Sources" directory.
- **Thesis document**
  This document, in both postscript form and pdf form, can be found in the "Manuscript" directory.

University of Cape Town