A Navigation System for Robot Soccer

Brett Browning, Gordon Wyeth and Ashley Tews

Computer Science and Electrical Engineering Department, University of Queensland
Brisbane, Australia 4072.

Abstract

This paper describes the navigation system used for the UQ RoboRoos robot soccer team, runner up in the 1998 world RoboCup championships. The navigation system has been developed to cope with the dynamic environment of robot soccer. Two interacting levels of the system called the schema and cognitive levels respectively, have been used to solve two orthogonal problems in spatial navigation: long-term planning and short-term motion control. A mechanism to compensate for the effects of sensor latency has been developed, overcoming the significant delay inherent in vision processing. This paper shows the performance of the system through the competition results, and the results of simulation.

1 Introduction

This paper presents the navigation and low-level control systems used by the UQ RoboRoos, a team of soccer playing robots. The architecture is capable of controlling the robots robustly in a dynamic environment with minimal computational power. Furthermore, the architecture actively compensates for sensor data that is old and out of date. The latter is a necessary requirement as the primary sensor for the robots, vision, introduces significant latency between events and their actual use in navigation.

The RoboRoos navigation system has been divided into two distinct components. A planning system, based on a coarse-grained occupancy grid [McKerrow, 1991], takes care of long-term planning problems, such as determining a path to the other side of the field without hitting any obstacles. Short-term planning problems, such as moment to moment control of individual wheel velocities, are handled by reactive control systems called *schemas* [Wyeth & Browning, 1998].

1.1 Robot Soccer

RoboCup games are the robot version of indoor soccer. The small-size league field consists of a table-tennis table, surrounded by white walls 10cm high. A middle-size league also exists where robots play on a field three times the size in every linear dimension. For the small-size league a standard orange golf ball is used in place of a soccer ball. The objective of the game is to score by hitting the ball into the opposition goal. In the small sized league, teams are allowed to use an overhead camera for recognising and tracking the robots and the ball thus simplifying the problem of determining where the robots are.

RoboCup competitions have been held since 1997. The aim of the competition is to provide a motivating force for research into intelligent robotics by providing a competitive but friendly environment with a challenging problem. [Kitano *et al.*, 1997] describes the major robot soccer intelligence issues as multi-agent cooperation, navigation in a dynamic environment, strategy learning, and skill in both movement and control of the ball. It is hoped that in the years to come the successes of RoboCup will spin-off contributions to the wider robotics community.

The RoboRoos robot soccer team consists of four homogeneous field robots and one specialised goalkeeper, all custom built to play soccer in the small-size league of RoboCup competitions. Figure 1 shows a picture of the team on a competition field.

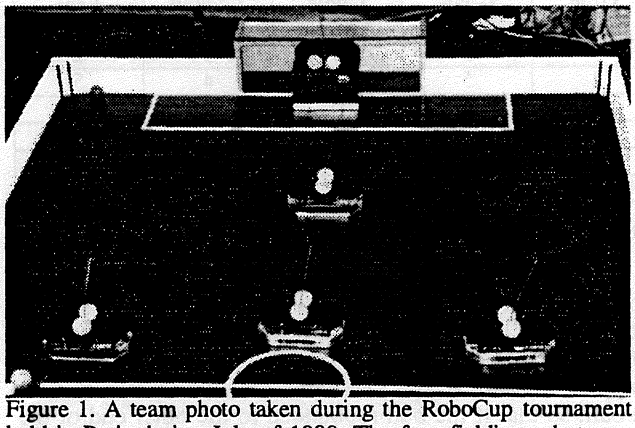

Figure 1. A team photo taken during the RoboCup tournament held in Paris during July of 1998. The four fielding robots are closest in the image with the goalkeeper at the rear.

1.2 Paper Outline

The following section (Section 2) provides an overview of the RoboRoos system and the place of the navigation components within it. Section 3 describes the navigation and kicking system components and the integration mechanisms employed. Section 4 presents results from simulation and competition game play showing the competency of the system and the problems currently within the system. Finally, Section 5 provides a summary

of the major results of the paper.

2 The UQ RoboRoos System

This section describes the RoboRoos architecture surrounding the navigation system. For a complete description see [Wyeth *et al.*, 1998] and [Tews & Wyeth, 1998].

2.1 Overview

Figure 2 shows a block diagram of the RoboRoos system. The hardware components of the system consist of a colour CCD camera, the off-board Pentium II 233MHz computer with a PCI based framegrabber card, the custom built communications hardware and the five custom built robots. The software processing is distributed over the off-board PC and the microcontrollers housed in each of the five robots.

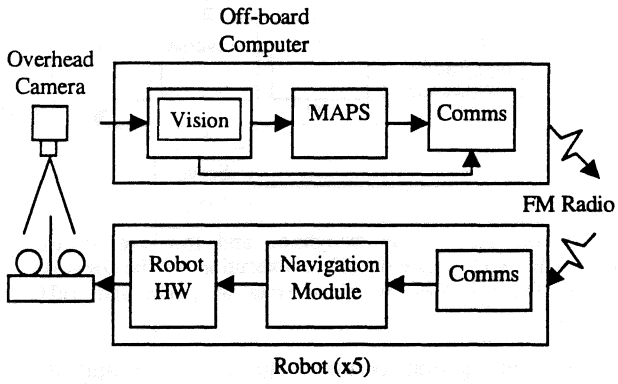

Figure 2. Overview of the UQ RoboRoos system. Data enters the system via the overhead colour camera. Visual processing and multi-agent planning (MAPS) are performed on an off-board computer. The results are broadcast to the five robots on the field using the FM communications network. The robots are responsible for carrying out the spatial navigation commands and low-level motor control.

The PC is responsible for processing the shared vision resource and for performing the high-level strategic game planning using MAPS (Multi-Agent Planning System) for the team. Spatial navigation and local control are performed on-board by each robot. A broadcast RF communications network is used to update each robot with the results of the vision processing and global planning. Although the processing for vision and MAPS is shared, each robot can be viewed as a single autonomous system with sensory information and navigation goals provided by the vision and MAPS systems, respectively.

2.2 Vision Processing

The vision system comprises of a camera, a frame grabber and a computer to identify and track the ten robots and the ball at a frame rate of 25Hz. Each image of 376x217 pixels is grabbed into the PC by the framegrabber card. Frames are placed into a rotating buffer so that the next frame can be grabbed while the current frame is being processed. This approach increases overall throughput, but does not decrease the delay between frame capture time and the time the data is used by the robots.

Recognition of the robot locations occurs in two stages. The first step is to search for the black cover of each robot. Since each robot is identical when viewed from above, the system tracks the robots positions across frames. Using this information, the predicted locations of the robots in the current frame are used as the start points for finding the black covers. Each black cover is found using a region growing algorithm based on a luminance comparison is performed. Pixels are added to a region if they contact the region and they have an average RGB intensity less than a user specified threshold. If a pixel does not have enough neighbouring pixels in the region it is rejected thus ensuring solid blobs are identified instead of long thin objects. Regions of suitable size and shape then pass into the second stage of processing.

Once a suitable region is found the algorithm attempts to locate the two markers on top of the robot. Again, a region growing algorithm is employed. In this instance, the pixels are added if their average RGB intensity is greater than the same threshold used earlier. When two objects that fit the size and shape of the markers are found the heading of the robot is calculated using the centroid of each marker region. The markers are differentiated by comparing the average RGB vector of each marker region to two prototype vectors using an Euclidean distance metric in RGB space. This comparison determines the front of the robot.

From a navigation standpoint, vision processing errors can be classified into fatal and non-fatal errors. Fatal errors are when robot identities swap permanently. Such errors are very difficult to recover from without user intervention. Non-fatal errors occur with short-term fluctuations in individual robot location and heading. The latter is a significant problem. Given relatively small size and separation of the markers, heading information is limited in resolution to approximately ±15°. Moreover, the heading value is prone to significantly wander in this range. These errors need to be overcome by the navigation system for reliable movement to occur.

2.3 MAPS - Global Planning

The global planner, called MAPS, generates strategic sub-goals for robot movement and determines what kind of action is required. In essence, the global planner provides the link between the overall goal of the team, to win, and the goals for navigation used by each robot.

MAPS is based on the use of potential fields. Cooperation results in an emergent fashion without explicit communication being required. While it is possible for each robot to run its own copy of the MAPS algorithm, MAPS has been implemented on the PC for ease of debugging. When new sensory information is available, the MAPS system builds a new potential field. The potential field incorporates sensory knowledge with the overall goals and strategies of the team. MAPS evaluates the potential field to determine where the players should move to, who should kick the ball, and where the ball should be kicked. The result is a parameterised command list for the team.

There are four commands used in the current system; *HALT, KICK, GOTO* and *KICKOFF*. Each parameter has a x and y parameter associated with it. The HALT command is used to stop the robots when they need to be moved by hand and before kick-offs. The

KICK command is used to tell a player to kick a ball and the <x,y> parameters indicate where to kick it to. At any one time there is only one kicking player. This mechanism prevents conflicts. The GOTO command tells a player to move to the location (x, y) in coarse-grained grid coordinates. Finally, the KICKOFF command is reserved for game restarts. The <x> parameter is user specified and represents the angle at which the ball will be kicked. The commands and their parameters are sent to the robots at each processed frame. It is the responsibility of the robots to attempt to perform each command.

2.4 Communications

The communications network is based on a broadcast system with a single transmitter (the PC) and five receivers (the robots). The PC transmits packet information each frame via the single channel FM transmitter connected to the spare serial port. The robots receive these packets via on-board dedicated single channel FM receivers. All receivers and transmitters have been custom built around the Radiometrix SIL series transmitters and receivers. The boards are designed such that the transmitter and receivers can be interchanged between 418MHz carrier modules and 433MHz carrier modules to prevent carrier wave conflicts with other teams. Each packet contains an estimate of the time since the frame was captured, position and heading information for the 10 robots on the field and the list of planner commands for each of the robots.

2.5 Robot Hardware

The mechanical design of the field robots has been optimised for speed, acceleration and cornering ability, while maintaining as broad as possible a profile for contacting the ball. The wheels are arranged in a wheelchair arrangement, with Teflon skids at the front and rear of the robots providing the third point of contact. The soft rubber tyres provide a high coefficient of friction, minimising slip at the tyre contact points. Each fielding robot is capable of accelerating at up to $3.6ms^{-2}$ with a top speed of $3.4ms^{-1}$. Current playing speeds are around $1ms^{-1}$. The goalkeeper robot, which uses four motors rather than two, offers greater linear acceleration ($8.1ms^{-2}$) at the expense of turning capabilities.

The electronics for the fielding robots and the goalkeeping robots are identical and are housed on two boards. The main board has the Motorola 68332 processor and associated memory, as well as the power electronics for supply of the subsystems and the control of the motors. The RF communication electronics are housed on a separate shielded board. Power is provided by custom built NiCd batteries located inside the robot frame.

All on-board processing is performed by the 68332 microcontroller. The processor is also responsible for generating the PWM waveforms that drive the robots and sampling the two encoders used to determine the velocity of each wheel. Radio communications is received by the microcontroller via the in built asynchronous serial port. In the current system, message packets from the PC are buffered using an interrupt routine.

3 Navigation System

Figure 3 shows the block diagram of the navigation and low-level control subsystems. Each robot maintains two threads of execution. The cognitive thread, formed by the navigation and kicking modules, is synchronised with the vision and global planning systems by virtue of the incoming message packets. Effectively, the cognitive thread operates at the frame rate of 25Hz. The schema thread is processed at 1kHz along with the servo control subsystem.

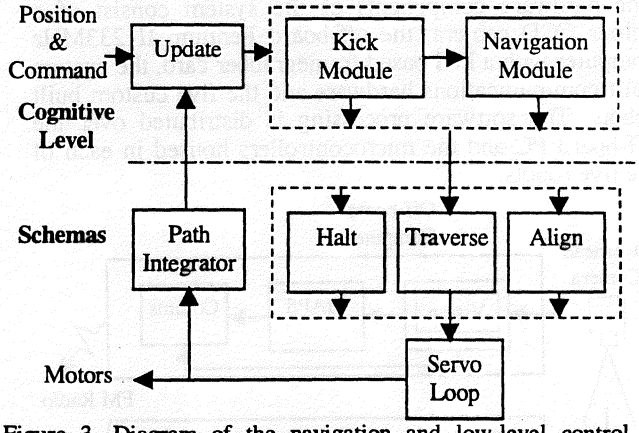

Figure 3. Diagram of the navigation and low-level control systems. The two separate threads of execution are the cognitive level and the schemas, which operate at 25Hz and 1kHz respectively.

The interaction between the cognitive and the schema threads forms the mechanism that allows the robot to make long term plans while still maintaining a fast response time for motor control. The navigation module effects its long-term plan by arbitrating between the schemas. The meshing between schemas is such that the active schema can be instantaneously interchanged causing a smooth transition in the robot's behaviour.

Schemas generate motion through the motion control subsystem. This module consists of a servo control loop that is responsible for maintaining the forward and rotational velocity of the robot at the levels specified by the active schema. This lowest level of control is implemented using linear PI control algorithms. The control loop uses the encoder feedback to maintain the robot's velocity. The PI control loop parameters have been calibrated to obtain a stable but fast response time to step and ramp inputs.

3.1 Schemas

Three schemas are used on the robot to control the setting of desired wheel velocities. A *Halt* schema is used to halt the robot when HALT commands are received. This schema reduces the robot's velocity until the robot has been stopped.

During normal movement, the robot executes either the *Traverse* or *Align* schemas. The Traverse and Align schemas provide translational and rotational motion, respectively. Each schema has a length parameter associated with it that allows the schema to terminate safely in the event of loss of communications. For example, if the robot is facing towards the top of the field and wishes to face the goal, the Align schema will be

instantiated with –90° length parameter. The robot will then turn, accelerating and decelerating in a fashion suitable for a 90° turn. The length of the turn is determined by encoder feedback. Should new information come through during the turn, the length parameter may be modified or the schema switched out in a smooth transition. In the event of no new information the turn will terminate smoothly. The schemas also have variable acceleration and velocity parameters. This means, for example, that the Traverse operation can be easily modified for an aggressive kick operation by increasing the acceleration and allowed velocity.

3.2 Cognitive Thread

Cognitive processing on the robot takes one of two forms. When a robot is given a KICK command, a special purpose kicking routine takes over. At any other time, when the robot is not halted, the spatial navigation system based on occupancy grids is used.

3.2.1 General Navigation

General spatial navigation uses a coarse-grained occupancy grid system [McKerrow, 1991]. The robot builds an internal map of the field, consisting of a 30x15 grid, using the latest sensory data. The dimensions of the map are limited by processing power and communications bandwidth. The size was chosen so that each robot occupies at least one cell maximally.

The map is built in two stages. Obstacles are first inserted into the map by placing large values in the grid positions around the obstacles reported locations. All objects foreign to the robot, including the ball, are treated as obstacles. Additionally, the defence box defended by the RoboRoos goalkeeper and the cells adjacent to the surrounding wall are also treated as obstacles.

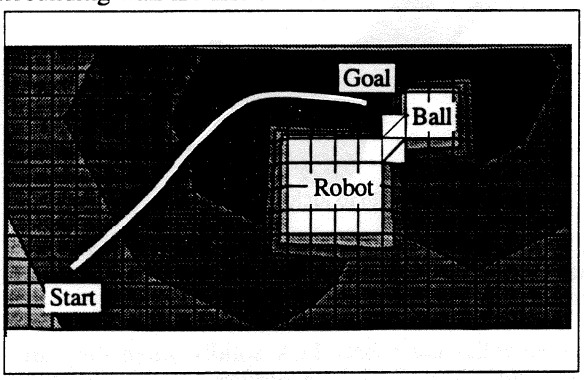

Figure 4. An contour plot showing an example flood fill and the resulting path. The colours represent collections of similar valued cells with darker colours representing the lower value.

Once the obstacles have been processed, the algorithm attempts to plan a path to the desired goal location (not to be confused with the soccer goal). Paths are planned using a flood fill algorithm. Each non-obstacle cell in the grid is set to the approximate Euclidean distance of the shortest path to the goal from that cell. The distance value in each cell is calculated in an iterative fashion beginning with the goal cell, which is assigned the value of zero. Each unmarked cell adjacent to the goal cell is set to an approximate of the distance between it and the goal cell. Cells adjacent to the newly processed cells are then calculated by adding the distance between the adjacent cells to that in the already calculated cell. Figure 4 shows a typical flood fill. Note that the flood fill algorithm successfully avoids getting the robot trapped behind other robots.

Once the flood fill algorithm has completed, the navigation module selects a schema to run until the next iteration. Examining the cell within which the robot is located and the cells adjacent to it determines which schema is selected. If the robot is at the goal cell, where the cell value is zero, the Halt schema is chosen. If not, the adjacent cell with the smallest value is chosen as the next cell for the robot to enter. In the cases where there are multiple cells with the same value, the cell in the current direction of travel is chosen. The direction to the chosen adjacent cell is calculated and used to determine which schema should execute. If the chosen direction is different from the current direction of travel by more then a set threshold, the Align schema is chosen. In the current system, the alignment threshold is set to approximately 10°. Alignment angles are currently restricted to multiples of 45°, in other words, North, North-East, East and so on.

3.2.2 Kicking

The kicking routines are responsible for getting the robot behind the ball and kicking it towards the commanded location. The kick location is specified by the <x, y> parameter generated by the MAPS algorithm. The actual kicking action is performed by the robots head butting the ball, as they have no specialised kicking mechanisms.

Kicking the ball can be broken into four possible scenarios. The first scenario occurs when the robot is behind the ball in the correct location to kick it. The robot triggers a Traverse schema over a suitable predefined length (currently set to 36cm). In order to give the ball a good "thump", the linear acceleration and maximum velocity of the robot are increased closer to the physical limits (currently at $3ms^{-2}$ and $3ms^{-1}$, respectively). The second case occurs when the robot is in the kick zone, but is incorrectly aligned to kick it to the desired location. Here the robot triggers an Align schema to change the heading by the appropriate amount.

The last two cases depend on the motion of the ball. An estimate of the ball's velocity, calculated from the previous two reported values, is used to predict where the kick zone is located. If the robot is already in the kick zone, it will wait for the ball to reach it. While the robot waits, it triggers an Align schema to aim in the correct direction. Once the ball reaches the kick zone, the above cases apply. If the ball is stationary or not moving towards an appropriate spot for the robot to kick it, the navigation system is invoked. The kick routine specifies the desired destination for the navigate system based on the predicted location of the kick zone. Once the robot reaches this location, whether by achieving the navigational goal or by some change in the ball's motion, processing occurs as defined above.

3.3 Sensor Latency and Path Integration

The delay of visual processing, combined with the delay in communicating the information to the robots creates a sensor latency problem. The information processed by the robots is out of date by approximately 120ms. There is

significant variation in the data age however.

Sensor latency creates a significant problem for effective navigation. Sensor latency affects both position and heading values. For a robot travelling at a modest $1ms^{-1}$, it will have moved 120mm during the time it takes to capture the image to receiving the information. Given that each cell is approximately 45mmx45mm large, the sensor latency corresponds to an error in location that may be as large as 2 grid coordinates. As [Elfes, 1987] point out, occupancy grids are highly sensitive to errors in localisation. Inaccuracies in heading information built up through sensor latency and non-fatal vision errors causes the Align schema to be constantly triggered. The effect causes the robot to twitch on the spot as if it does not know where to go!

The solution developed for the problem utilises path integration. Path integration is the ability to remember where the robot has been in the recent time frame. The robot's location information is then updated using the age of the data and the path integration information. The rotational and translational velocities of the robot since the frame was captured are integrated by summation. The result is added to the position and orientation information from the vision sensor. Given a reasonably accurate estimate of the frame age, the system works effectively and removes all traces of the robots twitching.

4 Results and Discussion

This section shows some of the results for the navigation and kicking system. The trails of movement have been generated using the RoboRoos simulator. The simulator models the full kinematics of the system and the ball dynamics during collisions (ball friction is also modelled). The simulation results show a high correlation with the observed on-field behaviour. On-field behaviour has been analysed from video footage of the robots during competition.

4.1 Navigation Performance

Figures 5 shows an example of the typical navigation performance around stationary and dynamic obstacles, respectively. Clearly evident is the ability of the system to plan paths around obstacles. Although the generated path is not optimal, it is still reasonably efficient as the robot is allowed to build up greater speed along straight sections.

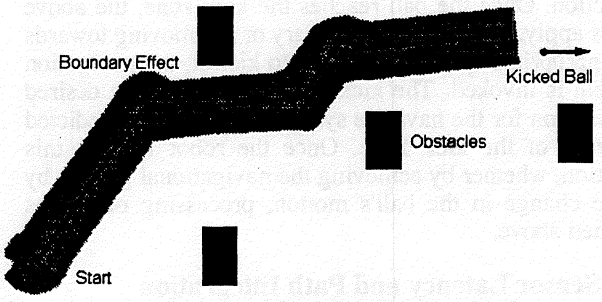

Figure 5. A typical navigation performance taken from the simulator. The real robots exhibit almost identical behaviour to the simulator. The lumps at the corners are the edges of the robot as it rotates.

The points labelled as boundary effects show the drawback of using the coarse-grained occupancy grid system. When the robot is moving close to the edge of a cell, it may wander into an adjacent cell location. The resulting abrupt changes heading direction generated by the path-planning algorithm severely affect the navigational efficiency of the robot. Locations near obstacles and regions where the gradient of the map changes direction tend to produce problematic boundary effects. Boundary effects significantly affect navigation performance by disrupting the robots momentum and causing the robot to take much longer to reach its destination.

The boundary effects are caused by the lack of resolution in position information available to the navigation system. Higher resolution position information would enable the robots to stay closer to the centre of the cell. In addition, the robot could look ahead into the next cell and prevent any potential boundary effect when it strays near cell edges.

4.2 Kicking Performance

In most situations, for balls travelling at slow velocities compared to the robot, the kicking mechanism described earlier works reliably. In practice, kicking accuracy is usually maintained to within $\pm 10^{\circ}$ based on observation. The main cause of error in the kicking angle is due to the poor resolution of heading information produced by the vision system. Figure 6 shows a kick in progress with the different kick cases labelled.

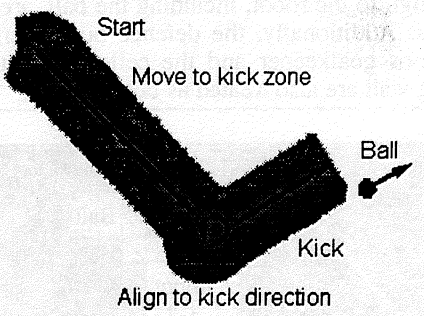

Figure 6. An example of the kick routine taken from the simulator. The robot kicked the ball into the open goal mouth located to the right of the image.

Generally, the robots kick solidly when they are behind the ball in a good field position to kick the ball. The cases where the robot is on the wrong side of the ball for kicking cause the worst performance. When the ball is moving too quickly relative to the robot's speed kicking performance also deteriorates rapidly. The main cause for these failures is the significant delay in getting into position caused by the navigation system when close to obstacles. Improvements to the general navigation system will have a direct flow on effect to kicking performance in these situations.

4.3 Game Results

In real competition, the UQ RoboRoos robot team has proven to be highly successful. After only three months of development, the team won the second place title in the

1998 world RoboCup championships. During games, the system showed remarkable defensive capabilities and decisiveness when attacking the goal.

Developments to the vision system since the competition have increased the frame rate from 8Hz to 25Hz. The higher frame rate has improved the standard of play by allowing the robots to move at higher speeds. The robots now move at up to $1ms^{-1}$ during general navigation and kick at up to $3ms^{-1}$. During the RoboCup competition the robots were limited to speeds under $0.5ms^{-1}$ during normal movement and kicked at only $2ms^{-1}$.

The improvements to the system after the RoboCup championships allowed the RoboRoos to reach the final stages of the Pacific Rim Series held in Singapore during November. Bugs in the vision system when exposed to low-lighting levels, a condition not experienced in Paris or during testing, led to a significant increase in fatal errors. Once the vision system lost track of the robots, it was unable to recover resulting in poor game performances compared to testing prior to the competition. The robots only performed reliably for short periods of play after game restarts. During the periods of reliable operation, however, the team played attacking soccer and scored goals in a convincing fashion. The sensitivity of the system to visual errors is a problem that will be examined during the preparation for RoboCup'99 to be held in Stockholm in August of 1999.

The on-field performance shown by the robots when reliable vision information is available highlights the success of the navigation architecture. Robots move into position without hitting opposition players in a relatively efficient manner. When kicking opportunities present themselves the robots attack the ball aggressively often improving the field position of the team. Although the paths generated by the navigation algorithm are not optimal, the robots efficiency of movement is still quite high. The major weaknesses in the system result from the limitations of the schemas chosen and the boundary effects discussed earlier. During normal play, it is difficult to detect most occurrences of the boundary effect problem due to the dynamic nature of the game. However, the problem still occurs making the robots slower to move into attacking positions.

When navigating turns on a planned path, the robot's forward momentum is severely disrupted. Although the interaction between the Traverse and Align schemas is seamless in terms of the velocity commands sent to the motion control subsystem, the effects on the momentum of the robot are not. All turns are performed with the same linear and rotational accelerations. Choosing the level of linear deceleration for turning requires balancing the need for high deceleration during large turns with the need to maintain the robot's linear momentum during small changes in heading direction.

Combining the Traverse and Align schemas into one *Curve* schema that modifies the robot's translational and rotational velocities, with minimal effects on the robot's forward momentum, offers one possible approach. The Curve schema could smoothly transfer the robot's momentum from translational to rotational and vice versa. The resulting smoother motion would not only improve navigational efficiency but would also reduce the amount of wheel slip that occurs. Reducing the amount of slippage experienced by each tyre improves the life of the tyres and improves the quality of the encoder feedback information. The latter would improve the reliability of the path integration system. Preliminary investigations into the development of a Curve schema are currently under way.

5 Conclusions

This paper has presented a two layered approach to the development of a navigational system for use in a dynamic environment. Each of the two layers, which form two separate threads of execution, is dedicated to different parts of the navigation problem. The slower cognitive thread solves the long-term planning problems of general navigation while the faster schema threads solve the problem of motion control with need for fast response times. The cognitive thread also arbitrates between schemas by selecting which schema has sole control over the robot's movement. A novel solution to the problem of sensor latency, based on the use of a path integration mechanism, has also been presented.

The success of the UQ RoboRoos robot soccer team against world class opposition shows the validity of the architecture and provides the best mechanism for the evaluation of the underlying approach.

References

[McKerrow, 1991] Phillip J. McKerrow. *Introduction to Robotics*. Addison Wesley, 1991.

[Elfes, 1987] Alberto Elfes. Sonar-Based Real-World Mapping and Navigation. *IEEE Journal of Robotics and Automation*, Vol. RA-3, No. 3, June, 1987.

[Kitano *et al.*, 1997] Hiroaki Kitano, Milind Tambe, Peter Stone, Manuela Veloso, Silvia Coradeschi, Eiichi Osawa, Hitoshi Matsubara, Itsuki Noda, and Minoru Asada. The RoboCup synthetic agent challenge 87. *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, pp 24-29, San Francisco, CA, 1997.

[Wyeth & Browning, 1998] Gordon F. Wyeth and Brett Browning. Cognitive Models of Spatial Navigation from a Robot Builder's Perspective. *Adaptive Behaviour*, MIT press, Volume 6, Issue 3/4, pp.509-534.

[Tews & Wyeth, 1998] Ashley Tews and Gordon F. Wyeth. Using Centralised Control and Potential Fields for Multi-robot Cooperation in Robotic Soccer. *Proceedings of Pacific Rim International Workshop on Multi-Agents (PRIMA), PRICAI '98*, November 22-27, Singapore, Editor: Toru Ishida, pp. 176-190.