UQ RoboRoos 2004: Getting Smarter

David Ball and Gordon Wyeth

School of Information Technology and Electrical Engineering
The University of Queensland
Brisbane, Australia
www.itee.uq.edu.au/~dball/roboroos/
{dball, wyeth}@itee.uq.edu.au

Abstract. This paper describes the University of Queensland's robot soccer team, the RoboRoos, that placed second at RoboCup 2003. This paper describes the mechanical systems, including the omni-directional drive system, the cross-bow kicker and the dribbling mechanism, and briefly describes the electrical and electronic systems. The paper explains the new software systems of the RoboRoos, which were the chief reason for the improvements in the team's performance in 2003. The paper describes the 70 frames per second vision system, the MAPS team coordination system, the action execution system, the navigation system and the motion control system. The paper concludes with thoughts for the future of the RoboRoos.

Introduction

The University of Queensland's RoboRoos [1], [2], [3], [4] team is one of the longest standing teams in the small-size league of RoboCup [5] having competed annually since 1998. During these years the performance of the team has been successful, and many research areas explored especially in the areas of multi-robot coordination and navigation in highly dynamic environments.

The 2002 RoboCup competition saw the first time that the RoboRoos team did not make it through the round robin section. The problems that the team experienced were deemed to be largely due to a build up of fixes and partial re-writes of many key modules of software. The team decided that the chief change for 2003 would be a complete review of all code, with complete re-writes of the low and medium level intelligence modules, optimisation of the robot's motion control and significant optimisation of the vision code. Debugging was greatly assisted by the addition of a playback mode that can recreate all intelligence decisions at each time step.

The improvements from the new software were significant. Despite only limited changes to the hardware of the robots that had failed to pass the round robin in 2002, the RoboRoos were able to reach the final of the competition, with a commanding 63 – 5 goal difference for the entire competition, and a 10 – 0 win over CMU in the semi final. In addition, the RoboRoos won the shooting challenge event, scoring 20 goals in 3 minutes.

System Overview

The RoboRoos system is a layered set of subsystems that each perform different tasks. The flow diagram in Figure 1 shows how the system is laid out. An overview of the system will be given by following the flow of the information from the camera to the robots actuators (motors).

Two overhead digital FireWire cameras capture global images of the field. The vision system processes the images to identify and locate the robots and the ball. The state of the field is sent to the Multi-Agent Planning System (MAPS). MAPS is the highest level planner in the RoboRoos system, responsible for distributing the goal of the team (to score the most goals) amongst the individual robots. MAPS coordinates the RoboRoos by selecting an action and an action location for each robot. Some example actions include KICK and DEFEND. The MAPS actions and action parameters are passed to the Action Execution system (AES).

The MAPS actions are interpreted by the AES system. Each action has a set of appropriate parameters and a notion of the overall desired robot motion. The Navigation module attempts to achieve the desired motion behaviour while avoiding obstacles. The Navigation module determines the immediate desired heading and distance for the Motion System. The Motion system accelerates and decelerates the robot to the desired heading and distance by creating force limited trajectories. This ensures wheel slip is kept to a minimum.

The 2003 system was run on two networked 1.6GHz Centrino Laptops. For RoboCup 2003, the system latency was approximately 40 milliseconds. This is broken down as follows. FireWire Image Transportation (14ms) + Vision (14ms) + Network (1ms) + Intelligence (5ms) + Wireless (6ms) = Total (40 ms).

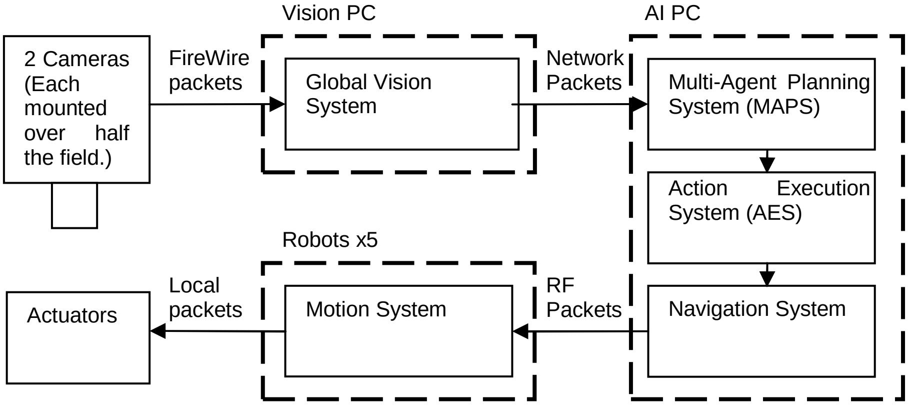

Fig. 1. Overview of the entire RoboRoo system.

Mechanical Design

The 2002 mechanical design features an omni-directional drive system, a powerful crossbow kicker and a dribbler all in a compact and robust design. All RoboRoos are of the same mechanical design, each having a mass of 1.5 kilograms. The RoboRoos are constructed using aluminium that gives them a strong but light frame. A polycarbonate cover protects the internals of the robots. The robots have a low centre of mass, achieved by placing the kicker DC motor, the majority of the batteries and the drive system on the 2 mm clearance plane at the base of the robot. The low centre of mass reduces weight transfer between wheels as the robot accelerates. Consistent weight transfer leads to less slip of the wheels across the playing surface.

Omni-Directional Drive

For maximum agility the RoboRoo robots have omni-directional drive. The RoboRoos omni-directional drive is implemented using three DC motors each with two castor wheels. Each motor is mounted at 120 degrees from each other. The wheels are placed at the extremity of the robot to minimise rocking. The wheels have a moulded rubber surface for good grip with the surface. A photo of the RoboRoos drive system is shown in Figure 2.

The DC motors are the MiniMotor 6 Volt 2224's (4.2 Watt) with mounted 512 line encoder. This motor has a recommended continuous speed up to 8000RPM and continuous torque up to 5mNm. The gear ratio from the motors to the drive shafts is 4.2:1. When running at the peak recommended performance, the drive system is capable of an acceleration of 2.38m/s/s and a maximum velocity of 2.88m/s.

To be competitive the robots are run at higher accelerations. This is generally between $3.0\mathrm{m/s}^2$ and $4.0\mathrm{m/s}^2$. Exceeding the recommended continuous torque is acceptable provided that thermal limits are monitored. To overload the motors a 9.6 volt supply is used to drive the 6 volt motors.

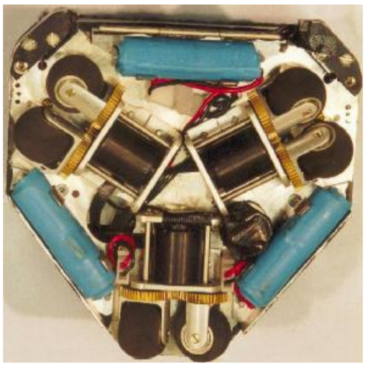

Fig. 2. Picture of the omni-directional drive system.

Crossbow Kicker

The RoboRoo robots feature a powerful kicking mechanism that is able to project the golf ball at up to 5.0m/s. It also has the ability to kick at 2.0m/s for passes. The kicking mechanism for the RoboRoos is of a crossbow design and uses an elastic cord to store the mechanical energy. The kicking mechanism, while mechanically complex, uses only one DC motor to both retract and release the kicker. The kicker also uses the 6 Volt 2224 MiniMotor. The plate that strikes the golf ball is the same mass as the golf ball. This gives maximum efficiency of energy transfer. More details of the kicker can be found in [6].

Ball Control Mechanism

The RoboRoos ball control device is a rotating rubber cylinder that applies backspin to the golf ball when in contact. A 6 Volt 2224 MiniMotor drives the shaft onto which the rubber is vulcanised. A 10:1 gearbox is used between the motor and the shaft.

One feature of the RoboRoos ball control mechanism is a 21mm gap cut in the centre of the bar. When the ball is located in this gap the RoboRoo has some control over the sideways motion of the ball. When the ball is located in this center gap their drive system motion becomes ball centric, that is, the robot rotates around the ball rather than its own geometric centre when rotating. It has another benefit as the crossbow kicker gives a more accurate and powerful kick to a ball located in the centre of the ball control mechanism.

Electrical Design

The core electrical and electronic design of the robots has not changed in the last three years of competitions, apart from a change of communications module and an increase in battery voltage from 7.2 V to 9.6 V. The electrical systems have been remarkably reliable, and have run without failure over three RoboCup competitions as well as long periods of testing and numerous demonstrations of the robots.

Main CPU Board

The RoboRoos Main CPU board uses Motorola's MC68332 32 bit microcontroller. This microcontroller executes the low level motor control loop. The major benefit of using this microcontroller is its built-in Time Processor Unit (TPU). The TPU has 16 channels (I/O pins) that can be configured independently for a variety of tasks, including quadrature decoding (QDEC), input capture (IC) and Pulse Width Modulation (PWM) generation. Except for the kicker mechanism, the motors have encoders for fast and immediate local feedback. The QDEC function removes the need for either the central CPU to process the incoming encoder signals or a dedicated FPGA / CPU for the task. The central CPU board has an external 256 kilobytes of FLASH and 128 kilobytes of RAM.

Batteries

The RoboRoo robots are powered by NiMH (Nickel Metal Hydride) 1000mAHr 9.6V batteries. The batteries are split into 4 sets of two cells each. Most of the cells are located on the base of the robot, in the same plane as the drive system. The robots are currently able to last 15 minutes on one set of batteries.

Wireless Communication System

As the environment is highly dynamic it is important that the RF communication system has low latency and low error rate. The RoboRoos use the 900Mhz spread spectrum RF transceiver modules from Innomedia that have a high data rate of 250 kbps.

Unfortunately the RF modules have a complex custom communication interface of multiple synchronous serial channels for transmit mode, receive mode and initialisation mode. The interface board uses a PIC16F873 microcontroller to hide this complex communication interface and allows the central CPU to communicate using standard UART serial. This makes easy conversion from the commonly used RadioMetrix modules. For our communication packet of 70 bytes the latency is approximately 6ms. Error rate are always low, for example at RoboCup 2002 and 2003 they were always under 5% at the full 250kpbs.

This is a high level of performance compared to the commonly used 418/433Mhz RadioMetrix RF modules by F180 league teams. Communications were found to be reliable (under 5% error) only at rates of 19.2kbps or less. This is approximately 13 times slower that the Innomedia system.

The serial port on a standard PC is limited to 115.2kbaud. To prevent this from being the limiting factor in this system the transmitter communicates with the PC’s USB port. The other benefit of using the PC’s USB port is that it can supply the power for the transmitter.

Vision

The vision system PC is connected to two Basler A301fc FireWire cameras that are capable of grabbing 640 x 480 images @ 80Hz. The images are grabbed in parallel but processed serially. The camera’s RGGB data format is colour classified using lookup tables. Classification determines whether a pixel is one, none or a combination of black, white, green, blue, yellow or orange. Calibration of the colour thresholding is a two stage process and uses a ‘painting’ process shown in Figure 3. First examples of the colours are selected by selecting small regions on the raw image. The pixels in these regions are displayed in the UV colour space, and the user then paints the colour regions in the UV colour space. The initial classification is performed at half resolution, skipping every second pixel and every second line. This reduces the processing time of this stage by a factor of four.

The next stage is to remove the field border. The field border is removed by region growing from a white pixel in each corner. This process is made more efficient by defining four rectangular corner areas to be ignored. To remove field lines and noise from the classification process, pixels of “not green” are thinned according to a neighbourhood threshold of seven. The remaining areas of “not green” are then grown in to a set of blobs for further local processing.

Blobs are parsed for areas of orange, blue and yellow pixels. Once found, the area around these pixels are re-thresholded at full resolution, and region segmented. Each coloured region is defined as an object and has its second moment of area calculated to determine its “stripiness”. Objects are identified as robots provided they have sufficient pixels and are sufficiently circular. To prevent concave robot configurations from being re-thresholded twice, each coloured circle is painted after it has been examined.

The area around own colour markers is region segmented for white objects. The direction of the robot is found by averaging the direction of the stripeiest white object, and the angle from the centre of the coloured circle to the centre of this object. The identity of each robot is determined by binary examination of the remaining white markers. The ball is located by examining orange pixels in an image that has not been thinned.

Finally Tsai’s algorithm [7] is used to lens correct the robot and ball locations.

At RoboCup 2003 the vision system processed 70 frames per second on a 1.6GHz Pentium M based laptop. It had a detection rate of 99% and no false positives. The vision system has an accuracy of approximately 5mm.

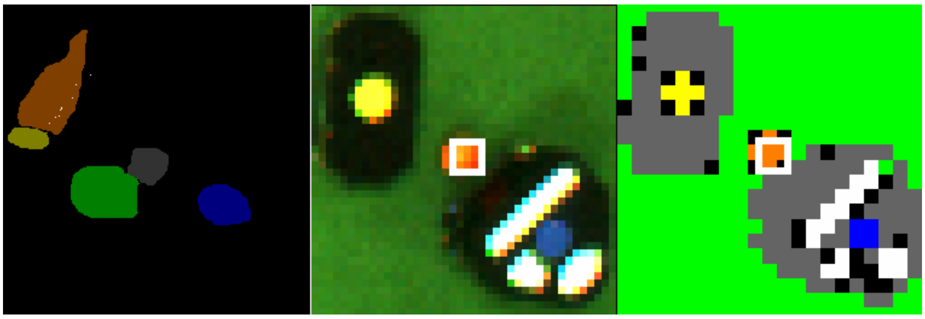

Fig. 3. The RoboRoos Vision System thresholding process. From left to right, (1) the paint window where the colour regions are defined, (2) the raw image from the camera, (3) the final colour thresholded image in half resolution. The black pixels represent unknown coloured pixels. Note that the areas around the makers are re-thresholded at full resolution later in the vision pipeline.

Intelligence System

The intelligence system is responsible for coordinating and determining the motion of the five robots. The inputs to the intelligence system are the packets from vision and the commands from the Referee Box, the output is the motion packets sent to the robots over the RF link. The intelligence system has three primary modes; Real, Simulator, Playback. All modes use the same intelligence system cores. This includes the Playback mode which recreates ALL of the intelligence decisions based on a binary log of the core state. This greatly assisted in debugging. The Real and Simulation and modes respond to commands from the Referee Box. All graphics are rendered using the OpenGL Utility Toolkit (GLUT). Total intelligence system time on a 1.6GHz Pentium M based laptop is 5 milliseconds.

Multi-Agent Planning System (MAPS)

The Multi-Agent Planning System (MAPS) is the highest level planner in the system, responsible for distributing the overall goal of the team (to score more goals than the opposition) amongst the individual robots. MAPS is responsible for the multi-robot coordination and cooperation by selecting an action and an action location for each robot. MAPS determines the team's actions based on the current world model, the team goal and the currently available actions.

MAPS uses potential fields as the mechanism for determining action selection and action location. The potential fields can model the suitability of an action for the different agents, or be used to find a suitable action location. MAPS has a library of potential field functions and abstractions, where each field is a two dimensional array of values. A more positive value represents a more desirable action for an agent, or, in the case of determining action location, a more desirable location for that action. The following four types of potential fields are examples of the type of fields used.

- Basefield: This field represents favourable regions of the physical environment. The regions of the field that will always be favourable to the goals of the team are the most positive.
- Object Regions: These fields model physical objects on the field by representing an area of effect around an object.
- Clear Path: This is an abstract feature that represents clear paths to objects or locations. It biases regions that offer a line-of-sight path to the point in question.
- Distance: This is another abstract feature that represents the distance from objects, thus favouring action locations close to the robot or to a goal location.

It is the overlaying of multiple fields that gives an abstract goal-biased terrain map that provides the information to determine the robots' sub-goals. By tuning the strengths and shapes of the component fields, MAPS can be tuned to give peak performance for a specific goal, or to act against specific opponent strategies.

MAPS uses potential fields to determine which robot will acquire the ball (the BallPlayer) as shown in Figure 4. By using potential fields many important factors that influence the selection of the BallPlayer can be considered. These include:

- the distance from the ball (Distance),
- the robots behind the ball relative to the opponents goal are they are able to run onto the ball in a better motion (Basefield),
- the effect of opponents between the ball and our robots (Clear Path),
- the last selected player to reduce jitter (Object Regions).

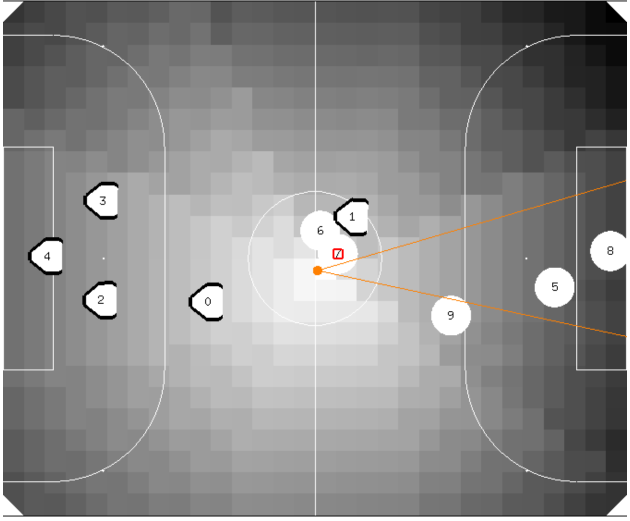

Fig. 4. The RoboRoos system determines our BallPlayer using potential fields. In this situation robot 0 would be selected as our BallPlayer even though robot 1 is closer. This is due to accounting the effect of the opponent robots in the way of robot 1. Other effects not apparent in this view is a biasing of the previous BallPlayer (to reduce jitter) and a biasing (magnitude and direction) of the ball's velocity.

Action Execution System (AES)

The Action Execution System is responsible for selecting the immediate robot behaviour. The field state and the MAPS assigned action are used as resources to determine this behaviour. The AES system fills in the details of the assigned actions from MAPS by decomposing it into a series of small tasks that can be performed using simple behaviours. The AES then decides on the immediate task to be achieved.

Each behaviour has a set of associated parameters and desired robot motion. Parameters associated with a robot include accelerations, application specific actuator state and desired repulsive strength of obstacles. The most important and complex action that the AES determines behaviours for are the BallPlayers actions which include shooting, dribbling and passing. For example if the robot is given a dribble action it may first have to acquire the ball, and if the robot is given a kick command but it must first ensure the robot is in an appropriate pose.

Navigation System (NAV)

The navigation system is responsible for robustly avoiding obstacles while still allowing the robot to achieve its desired pose. The AES system provides the desired pose for the robot and the relative avoidance strength for each obstacle to the NAV system. The NAV system determines the desired direction and distance for the robot to travel in. Obstacles may be physical objects such as the robots and the field boundaries. They may also be virtual obstacles that can be used to keep a robot out of a particular area (for example our defence zone).

NAV is a reactive navigation system that is appropriate to the highly dynamic environment. The different influences in the environment are represented using multiple schemas. Schema theory is a behaviour-based approach whereby overall robot behaviour results from the interaction of many simple schemas operating in parallel. In this navigation system, multiple schemas are represented by polar mappings that are centred on the robot. The Goal Direction Map represents the attraction of the desired goal location and the Obstacle Map represents the repulsion of the obstacles around the robot. The Motor Heading Map determined by a piecewise substraction of the Obstacle Map from the Goal Direction Map. The largest peak forms the desired navigation direction. The sequence of navigation frames in Figure 5 shows the RoboRoos navigation system in action.

The Navigation System has shown itself a capable system for achieving desired motion behaviour while competently avoiding obstacles in the highly dynamic environment. This is because:

- The complex problem of navigation is broken down into simple schemas that represent the different navigational influences.
- Reaction to environment change is fast as navigation is based on reaction, not on a plan.
- Navigation maintains competence even with poor behaviour selection by AES due to robust avoidance of obstacles.

The deficiencies of a reactive navigation system, local minima and non path optimal generation are generally not apparent. Local minima are not apparent as the global state is highly dynamic, meaning that problem states do not exist for long. Optimal path generation is wasteful, as the generated path will not exist for long, again due to the highly dynamic nature of the environment.

Motion Control System

The primary function of the motion control module is to ensure that the robot complies as closely as possible with the Navigation System requests without causing the robot to lose traction with the surface. The Navigation System provides a desired direction of travel and a total length that remains to the goal location. The motion control software seeks to provide constant force acting from the wheel to the ground. Typically, this force is somewhat less than the normal force applied by the robot to the ground so that good traction is achieved. Consequently, the robot limits straight line acceleration and rotational acceleration, and must adjust speeds when manoeuvring so that the wheels can generate sufficient force.

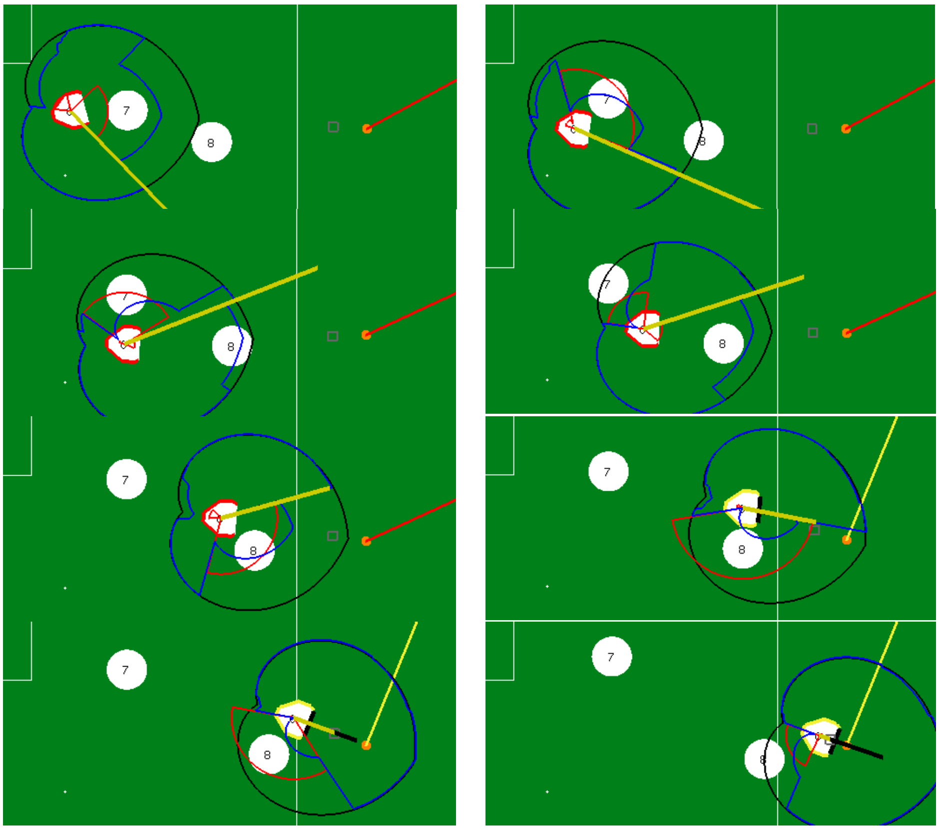

Fig. 5. Eight frames showing the RoboRoos navigation system in action. The black map is the Goal Direction map, the red map is the obstacle map and the blue map is the resultant map. The Yellow line that projects from the robot represents the resultant heading direction.

Maintaining good traction with the ground means that the motion control system can adequately perform its other essential role: keeping track of the robot's position between external sensor updates. In the highly dynamic environment significant rotation and translation can occur between external sensor updates. By accounting self motion between updates and during sensor delays, the motion control system greatly enhances the accuracy of the robot's interactions with the environment. It is critically important to time and aim kicks correctly. By using the immediate feedback from local sensors, the motion control software can execute timing critical functions with a precision beyond the other software modules.

An interesting function in the motion system software is that the point of reference can be changed for navigation commands. Typically, rotation takes place about the centre of the robot. Under direction from the AES, the motion control system can switch the rotation centre to an arbitrary point. This is applied to ball control. When the ball is directly in front of the robot, in the dribblers centre gap as detected by the local ball sensor, the motion control system can switch to rotating about the ball's centre. This provides better control of the ball during dribbling operations.

Conclusion and Future Work

The mechanical and electrical systems of the RoboRoos have remain mostly unchanged since the 2002 competition, with all of the teams efforts focussed on the software at all levels. Changes to the vision system have lead to great improvements in the frame rate and the accuracy of recognition. Changes in the MAPS multi-agent planning system have improved player selection, and introduced new attacking and defensive strategies. The action execution system has been greatly simplified with great improvements particularly in ball acquisition. The changes to the motion control system have improved the overall responsiveness of the robots, and significantly reduced the error in trajectory execution. The use of a playback system to view logs of games has greatly aided in debugging.

While each of these changes has contributed in only a small way, the overall improvement in the integrity of the system has caused a huge improvement in the performance of the robots. The system is now sufficiently reliable to look at new AI issues. One of the limitations of the RoboRoos system is its inability to adapt to a team’s playing style. With the adoption of the “hands-off” playing conditions in the Small Size League, it is no longer possible to enter strategy data to the robots. One of the prime directions for the RoboRoos team is to investigate methods for adapting play to defeat specific opponent tactics and strategies. These methods will be tested in the 2004 RoboCup tournament.

References

1. Wyeth, G., B. Browning, and A.D. Tews. *UQ RoboRoos: Preliminary Design of a Robot Soccer Team*. in *Lecture Notes in AI: RoboCup '98, 1604*. 1999.
2. Wyeth, G., A.D. Tews, and B. Browning. *UQ RoboRoos: Kicking on to 2000*. in *RoboCup-2000: Robot Soccer World Cup IV*. 2001: Springer-Verlag.
3. Wyeth, G., et al. *UQ RoboRoos: Achieving Power and Agility in a Small Size Robot*. in *RoboCup 2001*. 2002: Springer-Verlag.
4. Ball, D., et al. *UQ RoboRoos 2003: The Complete Rewrite*. in *RoboCup2002*. 2003: Springer-Verlag (Submitted).
5. Kitano, H., et al. *RoboCup: The Robot World Cup Initiative*. in *IJCAI-95 Workshop on Entertainment and AI/ALife*. 1995.
6. Ball, D., G. Wyeth, and D. Cusack. *Design of a Team of Soccer Playing Robots*. in *Autonomous Minirobots in Research and Edutainment (AMiRE)*. 2003. Brisbane, Australia.
7. Tsai, R.Y., *A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses.* IEEE Journal of Robotics and Automation, 1987. **3**: p. 323-344.