Protocols for Collaboration, Coordination and Dynamic Role Assignment in a Robot Team

Rosemary Emery, Kevin Sikorski and Tucker Balch

Abstract— **Creation of cooperative robot teams for complex tasks requires not only agents that can function well individually but also agents that can coordinate their actions. This paper presents several methods for collaboration and coordination in a team of soccer-playing robots. In our approach, fixed collaborative supporting behaviours allow for robots to aid each other and decrease interference. Coordinated dynamic role assignment then permits the robots to take advantage of their current location on the field. We present a robust protocol for dynamic role assignment based upon multi-threaded computer programming that mitigates the risk often associated with initiating a role change in a distributed system. This protocol is independent from the manner in which the decision to switch roles is made and would therefore support any approach to role assignment. The individual and supporting behaviours were tested at RoboCup 2001 in Seattle, Washington.**

I. Introduction

Robotic soccer is a domain that incorporates many of the core problems characteristic of mobile robotics, including localization, object recognition, motion planning and multi-robot coordination. Over the past two years, our lab has fielded the CMU Hammerheads, a middle-sized team in the RoboCup competition [1]. Based upon the team's performance in its first year of competition, we refined the team for its second competition this year.

Several hardware modifications have been made to the team since its debut at RoboCup 2000 in Melbourne, Australia, but the primitive behaviours of the team members remain the same. At the strategy level, however, the team is quite different. This year our focus was to combine older behaviour primitives with more sophisticated team coordination that allows the robots to interact with each other in ways that emphasize the strengths and minimize the weaknesses of the team. By moving beyond simple, specialized players and, instead, allowing robots to have roles that blend together elements of multiple positions, the team members are able to support each other and minimize interference.

A. Robotic Soccer

RoboCup is an international initiative designed to promote AI and robotics through a well defined domain that offers researchers the opportunity to apply a wide range of technology and solutions [2]. The original domain for the RoboCup Federation is that of robotic soccer. The CMU Hammerheads 2001 are members of the middle-sized league which consists of the largest robots currently competing. In this league each team has four members, each of which can be up 50cm in diameter and 80kgs. All robots carry their own sensors and are fully autonomous. Colour coding is used on the field to distinguish objects of importance. The ball is orange, the goals are blue and yellow, the field is green with white walls and field lines, and the robots are primarily black with magenta and cyan markers used to identify teammates. The objective, like that of real soccer, is to move the ball into the opponent's goal while defending your own goal.

R. Emery is with The Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, 15213. E-mail: remery@cs.cmu.edu. K. Sikorski is with the University of Washington, Seattle, WA, 98195. E-mail: kws@cs.washington.edu. T. Balch is with the Georgia Institute of Technology, Atlanta, GA, 30332. E-mail: tucker@cc.gatech.edu.

B. Collaboration and Coordination

In order to distinguish between the ways that robots can work together, we differentiate between collaboration and coordination. Collaboration occurs when multiple robots are working towards the same goal but do not explicitly coordinate their actions. Collaborative robots act as 'good teammates' by avoiding other players and broadcasting useful information to their teammates. Coordination involves more explicit protocols for deciding which robot will do what when. Coordination does not necessarily require communication, but it does require a robot to initiate a specific behaviour or set of behaviours should it recognize that a teammate is working towards a certain goal [3]. Dynamic role assignment is a type of coordinated behavior in which teammates entirely switch their roles under certain circumstances.

Developing collaboration and coordination in a robotic team is a difficult task. In soccer, robots must minimize their interferences, yet one robot must still get to the ball and transport it to the goal. This paper presents an approach that makes use of messages between the players to identify situations that require them to support or avoid another player rather than go after the ball themselves. The outcome of this strategy is that the players are able to better position themselves on the field than if each robot played as an individual. Dynamic role assignment, as an extension of this coordination, allows the players to switch their roles should their current locations place them at a disadvantage with respect to the game state. Switching roles involves risk as it can allow multiple players to take on the same role while no players fill some other role. Through the use of a protocol that requires locks and acknowledgements,

only one robot can initiate a role assignment and dictate the roles of its teammates, thus mitigating the risk.

In the remainder of this paper we will describe our research platform for robotic soccer and the individual player behaviours. A discussion of the team behaviours for collaboration and coordination and the limitations of this approach will then be presented. A protocol for dynamic role assignment that overcomes some of these problems is described in the last section.

II. Related Work

Ours is not the only robotic soccer team to exhibit coordinated behaviours. A high level of cooperation has been achieved among some teams in the simulation and small-size leagues of RoboCup which have the benefits of non-physical robots, and global vision and planning respectively. Our team, which competes in the middle-size league, faces the requirement that robots must be fully autonomous. Due to the nature of autonomous mobile robots, middle-sized league competitors initially focused on hardware problems and solving single mobile robot problems; however, more members of this league are starting to demonstrate a sophisticated level of teamwork.

The CS Freiburg 2000 and ART 2000 teams both use fixed goalies while allowing their other players to move between roles of main attacker, helping attacker and defender [4], [5]. These teams use utility functions to calculate which roles the players should assume. Each member of the CS Freiburg team transmits the values of utility functions for assuming each role to all other players. A player can switch roles if it has the highest utility value for that role and the robot currently in that role also wishes to switch, thus minimizing the probability that two players take on the same role at the same time. ART 2000 uses two utility functions, the values of which are also transmitted to all players. Robots take on roles based upon the value of these utility functions relative to the other players with a double threshold used to prevent oscillation between roles. The RMIT United 2000 team uses heuristics based upon information gathered from a world model maintained by each player to assign player roles [6]. Bonuses for the player currently in a given role and constant heuristic evaluation minimize the possibility of two players assuming the same role. Finally, each member of the Italian Golem team evaluates a Q-function for each possible action to determine which behaviour it should activate [7]. These values are broadcast to teammates and the player with the highest Q-value is allowed to execute its associated action. Thus, interference over the ball is minimized as the player with the ball will have priority in executing actions regarding ball manipulation.

There are elements of these teams that are similar to the CMU Hammerheads' strategy. Our breakdown of the three non-goalie players is similar to CS Freiburg and ART 2000 with a main forward, supporting player and a halfback. Unlike these other teams, however, we have unique support of the goalie by the halfback. In part this is because our goalie is itself unique. The CMU Hammerhead goalie will aggressively leave the penalty box in order to clear a ball away from the goal unlike those of the other teams.

It is important to realize that the problem of deciding that a role swap should occur is separate from the problem of executing the swap reliably. One important contribution of this work is a protocol that ensures role swaps are made reliably. The dynamic role assignment incorporated in our team involves a protocol for generic role assignment that could be initiated through utility function evaluation in a fashion similar to these other teams, but has extra safeguards beyond hysteresis to ensure that no role is held by two players at the same time. We, however, use case-based reasoning for determining when roles should be changed rather than utility functions.

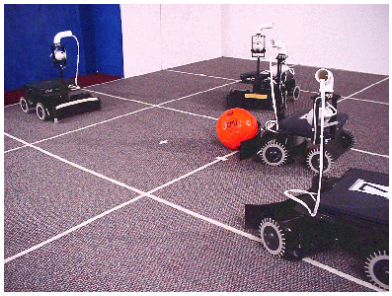

Fig. 1. The CMU Hammerhead 2001 Team.

III. Research Platform

The CMU Hammerhead 2001 team is comprised of four robots (see Figure 1). The robots were constructed at our lab as part of a project to build inexpensive, autonomous robots for the study of multirobot systems operating in dynamic and uncertain environments. The underlying mechanical platform is a commercially available non-holonomic robot and passive trailer [8]. Mounted on the trailer is a Pentium laptop which runs the robot's control systems. A shaft extending above the drive unit provides a mount for a digital camera, and a slip ring attachment ensures that the two-wheeled drive unit can rotate freely under the trailer. Each robot uses colour segmentation of the images it receives through its camera to identify approximate range and orientation to objects in its world [9]. The drive unit has a microcontroller that controls the motors and provides odometry and bump sensing. Mounted to the front of the drive unit is a shallow scoop that allows the robot better control over the ball. Communication between robots is provided by wireless Ethernet using PCMCIA cards on the robots and a wireless access point.

Robots are controlled using the Clay library of TeamBots [10], [11]. TeamBots is a Java-based collection of ap-

plication programs and packages for multi-agent robotics research. The Clay library is a group of Java classes which can be easily combined to create motor schema-based control systems [12], which are generally represented as finite-state machines. In this method each state corresponds to a set of activated behaviours, or a behavioural assemblage, for accomplishing a task. Perceptual nodes take information from the robot's sensors such as the location of objects of interest and obstacles. These nodes are embedded in motor schemas to produce vectors representing the desired trajectory of the robot. Behavioural assemblages, in turn, are formed by combining one or more motor schemas through weighted linear superposition. This behaviour-based approach to robot control is described fully in [10], [12].

The benefits of behaviour-based control in robotic soccer is that it allows the control system to quickly locate and acquire the ball in a highly dynamic environment. A planning approach would require a fast planner that is capable of dealing with the constraints of our non-holonomic robot. The success of the motor schema-based control systems in generating paths for our robots in generalized pushing tasks (including ball dribbling) is described in [13].

IV. Individual Player Behaviours

The CMU Hammerhead 2001 team is made up of four positions or roles: a goalie, halfback, floater and forward. While these are fixed roles, they are complementary and overlap with each other. This section describes the basic behaviours of each role. Sections V, VI and VII will describe the higher level team strategies created through collaboration, coordination and dynamic role assignment.

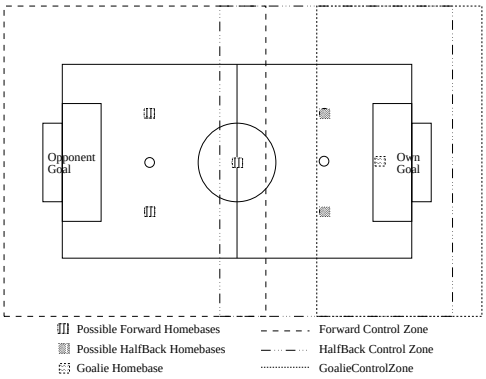

Fig. 2. Player Zones of Control. This diagram shows the zones of control, or danger zones, for the forward, halfback and goalie and each role's homebase. The floater will have a zone of control and homebase equal to that of the forward or halfback depending upon which of these two sub-roles it is currently playing. The control zones extend outside of the field of play in order to take into account errors in localization.

At its core, each position has roughly the same behaviour: 1) search for the ball by rotating in place and track it once it has been located; and 2) should the ball enter the role's area of control (its danger zone as shown in Figure 2), then move into a ball acquiring behaviour that lines the robot up behind the ball facing the opponents' goal. Once the robot has control of the ball, behaviour between the positions varies slightly. The goalie will push the ball to the boundary of its danger zone and then return back to its homebase. The halfback will also push the ball to the boundary of its danger zone but, should it still retain good control of the ball (i.e. the ball is in its scoop), then it will continue to push the ball to the opposing goal, returning home only when it loses control of the ball. The floater and forward will try to push the ball into the goal so long as it is in their danger zone which consists of the opponents' side of the field. If a robot is unable to locate the ball after searching for a specified length of time it will return to its homebase. For the goalie this is always a spot in the centre of the goal just in front of the goal line, for the other roles it is one of several positions dictated by the robot's current location on the field and those of its teammates (this will be further discussed in Section V-B). Roles also differ in the speed with which a player will move. The goalie moves the fastest as it needs to 'kick' the ball out of its danger zone. The halfback moves slightly slower while acquiring the ball in order to insure that it is well lined-up behind it, but will speed up once the ball is in its scoop. The forward moves the slowest during ball acquisition and dribbling in order to ensure that it has tight control of the ball. These core behaviours are then further modified by team collaboration and coordination.

We emphasize defense in our strategy because our goalie is somewhat handicapped by its non-holonomic nature. While it might seem counter-intuitive to give precedence to defensive players, when faced with a strong team one does not want the defensive players to yield control of the ball to a forward but instead wants the defensive players, with their greater speed, to move the ball out of the defensive zone. In order to play defensively we allow the floater to act as a forward when the ball is in the offensive zone but as a halfback if the ball is in the defensive zone or if the true halfback has control of the ball and is outside of the defensive zone.

V. Collaborative Behaviours

A. Teammate Avoidance

In order to prevent interference between team members, each player has the ability to avoid its teammates incorporated into its basic behaviours. This avoidance is implemented two ways. The first type of avoidance is a linear repulsion from another player's location, reported via **position** messages (see Figure 3). This type of avoidance is used in less critical behaviours such as going to a homebase. The second type of avoidance is a motor-schema that swirls the robot around the player it is avoiding. This is used in behaviours in which the robot still needs to make progress towards a goal like the ball. The avoidance of

teammates in this fashion is a form of collaboration. A hierarchical scheme that modifies the degree to which one player will avoid another player is a coordinated behaviour and will be discussed in Section VI-A.

position	**have − ball**	**goalie**
$time : 1500$	$time : 1800$	$time : 650$
$robotNum : 3$	$robotNum : 0$	$robotNum : 2$
$position : (x, y)(r, theta)$	$haveball : true$	$behaviour : leaving$
$heading : (x, y)(r, theta)$	$role : floater$	$side : right$
	$time : 3900$	$time : 6300$
	$robotNum : 0$	$robotNum : 2$
	$haveball : false$	$behaviour : entering$
	$role : floater$	$side : centre$

Fig. 3. Example Collaboration and Coordination Messages. The first message is the **position** message which is sent out by every player to indicate its current position. The second message type is the **have-ball** message which is sent when a player is performing an acquire behaviour and is within 1m of the ball. The final type of message is a **goalie** message which is sent by the goalie when it leaves or enters the penalty box while moving towards the ball.

B. Homebase Selection

The notion of returning to one's homebase when a player cannot see the ball is important to the way in which our team plays. The selection of the homebase allows us to exploit the robotic soccer rule which has the referee placing the ball on a penalty spot when game progress is not being made. The nine penalty spots are located along the centre line and 2m in front of each of the goals. If one of our robots is in this location or can see this location then we increase our chance of gaining control of the ball. As mentioned earlier, while the goalie's homebase is a fixed position, the other players are free to select from a variety of positions.

The halfback will select a homebase halfway down the defensive zone, either on the left or right side of the field. The side is selected by choosing the side opposite to that last used by the goalie for entering the penalty box in order to minimize interference between the halfback and goalie. This information is made available to the halfback through the **goalie** messages sent when the goalie enters and leaves the penalty box. The forward has the option of three homebase locations, the centre of the field, the far left penalty spot or the far right penalty spot (see Figure 2) and chooses the one closest to its current location. The floater will either select a homebase in the defensive zone or offensive zone based on its current sub-role. While acting as a halfback, it will select the halfback homebase farthest away from the current location of the halfback, and while acting as a forward it will select the forward homebase closest to its current location that is not also closest to the forward's current location. A robot attempts to go to the homebase closest to its current position because the referee will replace the ball on the closest penalty spot to the ball's current location.

VI. Player Coordination

This section discusses coordination strategies that enable two or more robots to act together.

A. Hierarchical Teammate Avoidance

The performance of last year's CMU Hammerhead team showed that a behaviour-based approach which has every player avoiding every other player equally, can result in repeated collisions with neither robot making progress. Instead, a hierarchical avoidance scheme permits the more important player to continue to make good progress while the other players avoid it. There are similarities between this and the dominance hierarchy described in [14]. Our team gives precedence to defensive players, and so the goalie is always at the top of the hierarchy and all players will avoid it. The floater will also avoid the halfback, and the forward will avoid the halfback and floater in addition to the goalie. Avoidance of teammates is always increased in non-essential behaviours such as returning to a homebase and is minimized in an essential behaviour such as acquiring the ball.

B. Ball Claiming

To further prevent teammate interference, a player that is within 1m of the ball and is in a ball acquiring behaviour can claim the ball using a **have-ball** message. If a lower-ranked player has control over the ball and a higher-ranked player then claims it, the lower-ranked player loses its claim on the ball. Again, this is done in a hierarchical fashion with the goalie having priority, followed by the halfback, floater and forward respectively. The purpose of the ball-claiming is two-fold: firstly, all players will avoid the teammate that currently has the ball in all behaviours, regardless of that team member's position in the hierarchy; and secondly, knowledge that a certain teammate has control of the ball is used to move the other players into supporting behaviours.

C. Teammate Support

We implemented two coordination strategies: goalie support and offensive support. For both of these support strategies, one of the players sends out appropriate messages and expects another robot to adjust its behaviour appropriately. Special **goalie** messages are used to indicate when the goalie is out of the penalty box, while offensive support is initiated by the **have-ball** messages which are also used to help with teammate avoidance.

As mentioned earlier, by necessity the CMU Hammerhead goalie is aggressive. It will leave the penalty box in order to clear a ball away from the goal. Once this is done, the goalie must turn around and go back into position in front of the goal. While the goalie is returning to the goal it cannot see the ball and so there is the danger of an opponent being able to score. In order to minimize the time dur-

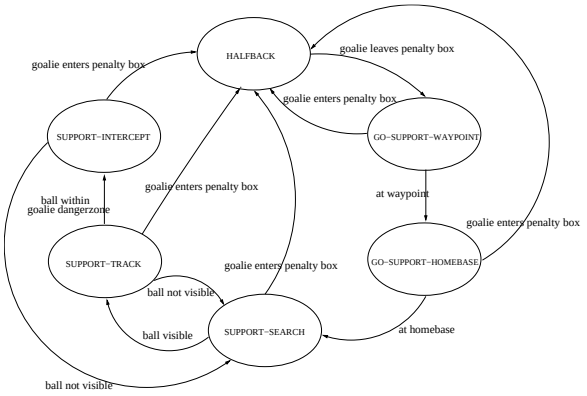

Fig. 4. Goalie Support FSM. The halfback moves between its halfback role and that of goalie support based upon **goalie** messages.

ing which our goal is unprotected, we implemented support of the goalie by the halfback. When the goalie leaves the penalty box it sends a **goalie** message indicating that it is leaving and the side of the goal from which it is leaving. Upon receipt of this message, the halfback will go into the goal from the opposite side and then move through the penalty box to take up a position just in front of the penalty box. Once in this location it will search for and track the ball and, should the ball come into the goalie's danger zone, intercept the ball as shown by the finite state machine in Figure 4. When the goalie has cleared the ball away from the goal (or given up) it returns to its homebase and, as it re-enters the penalty box, it sends a **goalie** message indicating that it is entering the penalty box and from which side it is entering. The halfback will then move to a homebase position that is on the opposite side of the field from the side indicated by this message. This is designed to minimize interference with the goalie.

The progress of the halfback through the penalty box takes less than the 10 seconds that are permitted for a second defender to be in this defense area. Overall this scheme leaves the goal more protected than the goalie can do alone. Figure 5 shows a sequence of the halfback supporting the goalie during game play.

In 2000, in order to have full coverage of the field, the team had two forwards, each of which were responsible for the entire offensive zone. This scheme ensured that at least one of the two forwards would attempt to acquire the ball; however, it also resulted in interference between them. This year, in addition to the hierarchical scheme for avoiding teammates, a supporting behaviour was used that places the offensive player without the ball at a more advantageous location.

When an offensive player gains control of the ball, a teammate will move to a location that is just behind and to one side of it. This support behaviour is initialized by the receipt of a **have-ball** message from a valid player. As the player with the ball moves up the field, so does this supporting player. If the initial player should lose control of the ball then the supporting player is in an ideal position to regain the team's control of the ball. The position of this supporting player also helps prevent opponents from getting to the ball and helps block them if they should gain control of the ball. The dedicated forward is able to support either the floater or halfback, and the floater can support the forward (as shown in Figure 6).

This offensive support was successful in minimizing interference between the forward and floater and in advantageously placing the robots on the field. One benefit, was that the supporting player, who frequently could not see the ball as it was blocked from view, still maintained a good position on the field instead of returning to its homebase position as it would have done in this situation the previous year.

D. Limitations without Dynamic Role Assignment

The halfback's behaviour and offensive support govern how the robots interact as the ball moves up the field; if a halfback manages to gain control of the ball it will continue to move towards the goal until it loses control of it, thus acting as a pseudo-forward. It will be supported by the forward while it has control of the ball and the floater will drop back to play defense. Otherwise, if either the forward or floater has control of the ball they will be supported by the other player. One problem with this approach; however, is that if the halfback loses control over the ball when it is close to the opponent's goal, it will give up and move back towards the defensive zone. The floater, which has been staying in the defensive zone, must then move up the field in order to support the forward or to get the ball. This results in wasted travel time and leaves the defense vulnerable. Instead, if the players can switch between their roles of halfback, forward and floater then they can take advantage of their current locations on the field.

VII. Dynamic Role Assignment

A. The High-Level Swap Protocol

Initiating a role change in a distributed system such as robotic soccer involves risk. For example, if two robots simultaneously decide to initiate conflicting role assignments, it is possible for the team to be left with two attackers, instead of one attacker and one defender. It is, therefore easy to draw parallels between between dynamic role assignment and multi-threaded computer programming. In the latter, mutually exclusive locks, or mutexes, are frequently used to ensure that two threads will not manipulate a data region at the same time. Our method of role swapping borrows from this approach [15], [16].

A.1 The Process: Locking

In order to successfully complete a role reassignment, the initiating robot must:

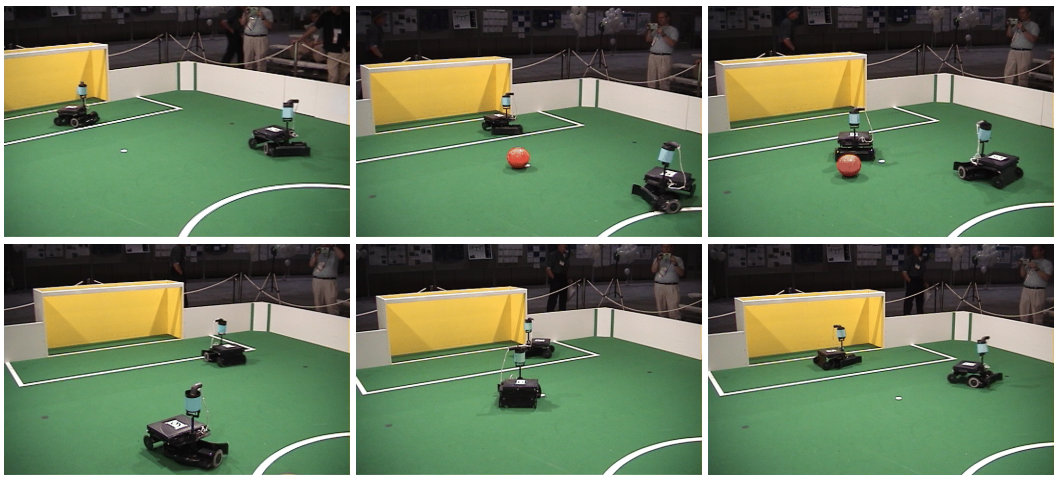

Fig. 5. Goalie Support. The halfback (player on right) supports the goalie (player on left) when the goalie leaves the penalty box in order to clear the ball away from the goal.

1. **LOCK:** Request 'locks' from each of the other robots.
2. **ACKLOCK:** Receive acknowledgement of locks from each robot.
3. **SWAP:** Issue the role reassignment command to each robot.
4. **ACKSWAP:** Receive acknowledgement of the reassignment from each robot.

It should be noted that during the entire decision process, all robots will continue to execute the behaviours appropriate for their current role. Steps 1 and 2 of the role assignment process are analogous to establishing control of a mutex on a dataspace used by two distributed program threads. In step one, an initiating robot makes it known that it wants to try to reassign roles, and 'locks' the other robots out from also trying at the same time. The initiating robot must also wait for an acknowledgement from the other robots. If all the robots do not respond in a certain period of time, the initiating robot simply aborts its reassignment attempt as it is likely that another robot is trying to reassign roles at the same time.

The robot that is being locked must also take steps to avoid dead-lock. For example, if the initiating robot crashes halfway through the protocol a dead-lock could occur. In order to prevent this, a timer is set each time a dynamic role reassignment message is received. If the timer is allowed to expire before progress is made towards the role reassignment, then the robot is unlocked, and it is then allowed to be locked and reassigned by other robots.

Note that the reassignment protocol does not need to explicitly resolve situations in which multiple robots try to initiate role swaps at the same time (resource contention). Our protocol forces all robots to unlock and wait a short time before attempting another reassignment. It is possible that the dead-lock condition will be reached again; however, given that RoboCup is a substantially dynamic domain involving mobile opponents and a moving ball, this is unlikely to occur many times before the the conditions leading to the resource contention change.

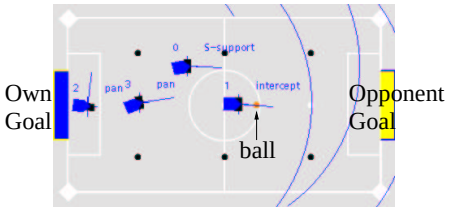

Fig. 6. Offensive Support. This simulation snap-shot shows the forward (player 0) in a supportive role during offensive play. The floater (player 1) has control of the ball.

A.2 The Process: Reassignment

Once it is known that all the other robots are willing to exchange roles, the initiator assigns and transmits each robot's new role. It again must wait for acknowledgement of these messages. If one or more robots do not respond in a pre-determined period of time, then the initiating robot must take action to remedy the situation. Two simple approaches are to either resend the role assignment to the robots that are not responding, or to 'roll-back' the assignments to the previous configuration. Each of these has its advantages and disadvantages. Resending the unacknowledged role may help if the last message was lost due to interference or network congestion; however, if it was lost because of failed networking hardware, then this will not help. 'Rolling-back' to the previous assignments is safer in the case of failed hardware: if the robot is still functional, but unable to communicate, then the resulting team will still have the correct number of attackers and defenders.

A.3 Sequencing

Network packets can be delayed for a number of reasons. During testing, it was observed that a message gener-

ated by a first, failed attempt at reassignment could arrive very late, in fact, late enough to be received while processing a second reassignment request. For example, if an *acknowledge-lock* message was received late, it could signal that the sender was ready to undergo a role change, when in reality it was not ready and instead had been locked by another robot.

In order to eliminate the possibility of interference from past delayed messages, a sequence identifier was added to each dynamic role assignment message. This is a counter that identifies the reassignment attempt to which the current message belongs. In order to process dynamic reassignment messages, the sequence number associated with a *lock-request* message is stored when the message arrives. Until either the role reassignment is complete or the lock times out, all other dynamic reassignment messages must arrive from the locking robot, and have the same sequence number. Using this method, even if two robots are using the same sequence number, their messages will not interfere with each other.

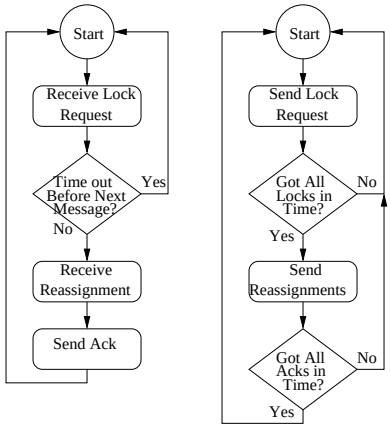

Fig. 7. Protocol for Dynamic Role Assignment. The left chart is followed by robots receiving roles. The right chart is followed by the robot initiating the reassignment.

B. Implementation

This protocol for dynamic role assignment has been implemented in TeamBots and tested in simulation using case-based reasoning for deciding when to initiate a swap, and assignment roll-back if acknowledgements are not received from all team members during the reassignment phase.

It is sub-optimal for the halfback to travel back to the defensive zone and the floater move out of that zone if the halfback loses control of the ball near the opponent's goal. Instead, if the halfback can then become a forward, it could take advantage of its close position to the ball and goal, and the floater, by becoming a halfback, could take advantage of its position in the defensive zone. There would, therefore, be no need for players to move around the field in this situation. We implemented this exact role switching policy, initiated by the halfback when it loses tight control of the ball while in the upper half of the offensive zone. The identification of this situation is done by the halfback using its current position and that of the ball on the field. When it identifies this situation, it follows the procedure shown in Figure 7 to initiate a swap. In the current implementation, the forward will always become a floater and the original floater a halfback; however, this could be further refined so that the team member closest to the halfback at the time of a swap becomes the new forward. Figure 8 shows dynamic role assignment during simulated game play.

A more complicated procedure for determining when to initialize role re-assignments and what those new roles should be (such as the utility functions discussed in Section II) can be used with our protocol and this remains an area open to the team for further refinement in the upcoming years.

VIII. LIMITATIONS

One limitation of the outlined collaboration and coordination schemes lie in their dependence on communication. Should communication stop functioning, individual players would not receive the **position** messages that allow them to avoid their teammates. If no **goalie** or **have-ball** messages are received then goalie and offensive support will not be initialized. The team strategy, however, is based upon players that functioned in the previous year without the more complex team behaviours and, therefore, will still be able to work towards achieving their goals even if communication fails. The dependence upon communication, however, can be reduced through visual detection of teammates. Using the colour markers that indicate teammates, players could locate and maintain a model of where their teammates are located in relationship to the ball and penalty box. They could then use the model and hierarchical information to decide if another robot has control of the ball and if the goalie is out of the penalty box. This would allow them to both avoid their teammates and move into appropriate supporting behaviours. Dynamic role assignment will not function in the absence of communication as outlined in this paper. Due to its design; however, our protocol can recover should communication be lost during game play.

A second limitation is that of the quality of the localization of each player. If player's are poorly localized relative to each other then their **position** messages will not be interpreted correctly. This was seen during extended game play where one player would avoid the **position** sent by a badly-localized robot and as a result, collide with that robot. Poor localization can also lead to the goalie not being able to recognize when it is out of the penalty box or properly located in front of the goal. Again, visual information can be used in the team behaviours, either instead of

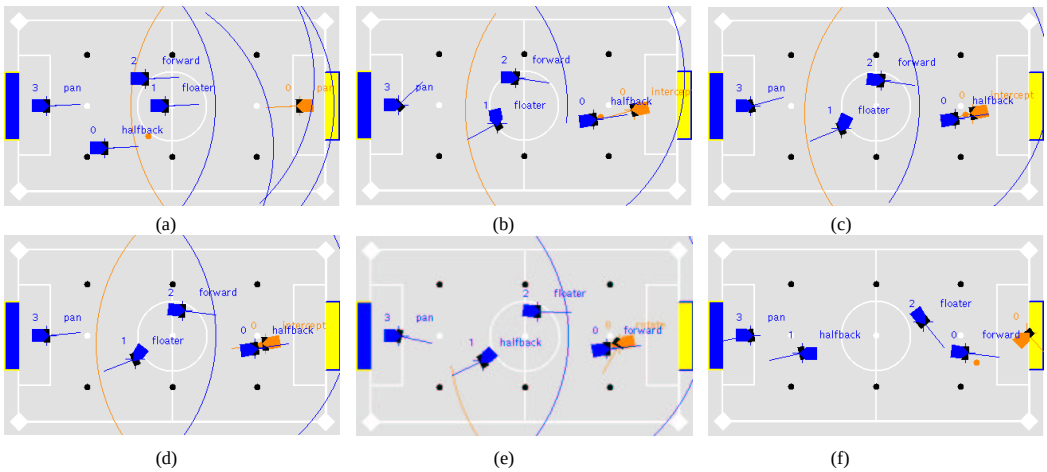

Fig. 8. Dynamic Role Assignment. The halfback, player 0, gains control of the ball in (b) and moves up the field while the forward moves into a supporting position and the floater drops back to play defense ((c) and (d)). When the opposing goalie takes the ball from the halfback in (e), the halfback initiates a swap and switches to be the forward, while the original forward (player 2) becomes the floater and the floater (player 1) becomes the halfback. Play then continues in (f).

or augmenting communicated information, to reduce these problems.

IX. Conclusions

Individual behaviours are not sufficient for good performance in robotic soccer and instead collaboration and coordination are necessary. Hierarchical teammate avoidance and support minimize interference between players and place them at advantageous locations on the field. Without the ability to dynamically switch their roles; however, the team members cannot always take best advantage of the game situation and so performance is limited. A protocol for dynamic role assignment was developed in order to overcome this limitation of the team.

Acknowledgements

The authors thank Ashley Stroupe and Steve Stancliff for their contributions to the team. Finally we thank Manuela Veloso for her ongoing encouragement and advice. This research was sponsored by the United States Army under Grant No DABT63-99-1-0013 and by a Natural Sciences and Engineering Research Council of Canada Fellowship. The content of this publication does not necessarily reflect the position or the policy of the sponsors and no official endorsement should be inferred.

References

[1] R. Emery, T. Balch, R. Shern, K. Sikorski, and A. Stroupe, "CMU Hammerheads team description," in *RoboCup 2000: Robot Soccer World Cup VI*, P. Stone, T. Balch, and G. Kraetzschmar, Eds., Melbourne, Australia, August 2000, Springer.

[2] H. Kitano, M. Asada, Y. Kuniyoshi, I. Noda, and E. Osawa, "Robocup: The robot world cup initiative," in *Proceedings of the First International Conference on Autonomous Agents (Agents - 97)*, Los Angeles, California, February 1997.

[3] R. Simmons, S. Singh, D. Hershberger, J. Ramos, and M. F. Smith, III, "First results in the coordination of heterogeneous robots for large-scale assembly," in *Proceedings of the ISER '00 Seventh International Symposium on Experimental Robotics*, December 2000.

[4] T. Weigel, W. Auerbach, M. Dietl, B. Dümler, J.-S. Gutmann, K. Marko, K. Müller, B. Nebel, B.Szerbakowski, and M. Theil, "CS Freiburg: Doing the right thing as a group," in *RoboCup 2000: Robot Soccer World Cup VI*, P. Stone, T. Balch, and G. Kraetzschmar, Eds., Melbourne, Australia, August 2000, Springer.

[5] G. Adorni, A. Bonarini, G. Clemente, D. Nardi, E. Pagello, and M. Piaggio, "ART'00 - Azzurra robot team for the year 2000," in *RoboCup 2000: Robot Soccer World Cup VI*, P. Stone, T. Balch, and G. Kraetzschmar, Eds., Melbourne, Australia, August 2000, Springer.

[6] J. Brusey, M. Makies, L. Padgham, B. Woodvine, and K. Fantone, "RMIT United," in *RoboCup 2000: Robot Soccer World Cup VI*, P. Stone, T. Balch, and G. Kraetzschmar, Eds., Melbourne, Australia, August 2000, Springer.

[7] P. de Pascalis, M. Ferraresso, M. Lorenzetti, A. Modolo, M. Peluso, R. Polessel, R. Rosati, N. Scattolin, A. Speranzon, and W. Zanette, "Golem team in middle-sized robots league," in *RoboCup 2000: Robot Soccer World Cup VI*, P. Stone, T. Balch, and G. Kraetzschmar, Eds., Melbourne, Australia, August 2000, Springer.

[8] Probotics, "The cye robot," http://www.probotics.com, 2001.

[9] J. Bruce, T. Balch, and M. Veloso, "Fast and cheap color vision on commodity hardware," in *Workshop on Interactive Robotics and Entertainment*, Pittsburgh, PA, April 2000, pp. 11–15.

[10] T. Balch, "Clay: Integrating motor schemas and reinforcement learning," Tech. Rep. College of Computing Tech Report GIT-CC-97-11, Georgia Institute of Technology, March 1997.

[11] T. Balch, *Behavioral Diversity in Learning Robot Teams*, Ph.D. thesis, College of Computing, Georgia Institute of Technology, December 1998.

[12] R. Arkin, "Motor schema-based mobile robot navigation," *International Journal of Robotics Research*, vol. 8, no. 4, pp. 92–112, 1989.

[13] R. Emery and T. Balch, "Behavior-based control of a non-holonomic robot in pushing tasks," in *Proceedings of International Conference on Robotics and Automation*, Seoul, Korea, May 2001.

[14] D. Goldberg and M. J. Mataric, "Interference as a tool for designing and evaluating multi-robot controllers," in *Proceedings of AAAI-97*, Providence, Rhode Island, July 1997.

[15] A. Silberchatz, H. Korth, and S. Sudarshan, *Database System Concepts*, McGraw Hill, 3rd edition, 1997.

[16] E. W. Dijkstra, "The structure of the "THE" multiprogramming system," *Communications of the ACM*, vol. 11, no. 5, pp. 341–346, May 1968.