The CMUnited-97 Robotic Soccer Team: Perception and Multiagent Control *

Manuela Veloso   Peter Stone   Kwun Han
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA 15213
{mmv,pstone,kwunh}@cs.cmu.edu
http://www.cs.cmu.edu/{˜mmv,˜pstone,˜kwunh}

Submitted to Autonomous Agents '98, October 1997

Abstract

Robotic soccer is a challenging research domain which involves multiple agents that need to collaborate in an adversarial environment to achieve specific objectives. In this paper, we describe CMUnited, the team of small robotic agents that we developed to enter the RoboCup-97 competition. We designed and built the robotic agents, devised the appropriate vision algorithm, and developed and implemented algorithms for strategic collaboration between the robots in an uncertain and dynamic environment. The robots can organize themselves in formations, hold specific roles, and pursue their goals. In game situations, they have demonstrated their collaborative behaviors on multiple occasions. The robots can also switch roles to maximize the overall performance of the team. We present an overview of the vision processing algorithm which successfully tracks multiple moving objects and predicts trajectories. The paper then focusses on the agent behaviors ranging from low-level individual behaviors to coordinated, strategic team behaviors. CMUnited won the RoboCup-97 small-robot competition at IJCAI-97 in Nagoya, Japan.

Content Areas: autonomous robots; multi-agent teams; coordinating perception, thought, and action; multi-agent communication, coordination, and collaboration; real-time performance.

*We thank Sorin Achim for developing and building the robots. This research is sponsored in part by the Defense Advanced Research Projects Agency (DARPA), and Rome Laboratory, Air Force Materiel Command, USAF, under agreement number F30602-95-1-0018 and in part by the Department of the Navy, Office of Naval Research under contract number N00014-95-1-0591. Views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied, of the Air Force, of the Department of the Navy, Office of Naval Research or the United States Government.

1 Introduction

Problem solving in complex domains often involves multiple agents, dynamic environments, and the need for learning from feedback and previous experience. Robotic soccer is an example of such complex tasks for which multiple agents need to collaborate in an adversarial environment to achieve specific objectives. Robotic soccer offers a challenging research domain to investigate a large spectrum of issues of relevance to the development of complete autonomous agents [7, 3].

The fast-paced nature of the domain necessitates real-time sensing coupled with quick behaving and decision making. The behaviors and decision making processes can range from the most simple reactive behaviors, such as moving directly towards the ball, to arbitrarily complex reasoning procedures that take into account the actions and perceived strategies of teammates and opponents. Opportunities, and indeed demands, for innovative and novel techniques abound.

We have been pursuing research in the robotic soccer domain within the RoboCup initiative [6], which, in 1997, included a simulator league and small-size and medium-size robot leagues. We have been doing research extensively in the simulator league, developing learning techniques and team strategies in simulation [12, 11]. Many of these team strategies were directly incorporated into the robotic system described here. We eventually hope also to transfer these learning techniques to the real system as we develop a complete Robotic Soccer architecture.

In this paper, we focus on presenting our team of small robotic agents, namely CMUnited-97, as a complete system with action, perception, and cognition capabilities. We developed the physical robots as actuators, a vision processing algorithm to perceive the world, and strategic reasoning for individual and collaborative behaviors.

The team is clearly not a perfect version of our multiple autonomous agents. We have developed previous versions of the team [1], and, as presented in the discussion and conclusion section, we are currently (and will continue) improving the team further. However, we believe that CMUnited-97 represents a major advance in our work and has several interesting contributions which we present in this paper:

- Reliable perception through the use and extension of a Kalman-Bucy filter. Sensing through our vision processing algorithm allows for (i) tracking of multiple moving objects; (ii) and prediction of object movement, particularly the ball, even when inevitable sharp trajectory changes occur.

- Multiagent strategic reasoning. Collaboration between robots is achieved through: (i) a flexible role-based approach by which the task space is decomposed and agents are assigned subtasks; (ii) a flexible team structure by which agents are organized in *formations*, homogeneous agents flexibly switch roles within formations, and agents switch formations dynamically; and (iii) alternative plans allowing for collaboration (e.g. passing or shooting), are controlled by pre-defined metrics for real-time evaluation.

The combination of robust hardware, real-time vision, and intelligent control code represented a significant challenge which we were able to successfully meet. The work described in this paper is all fully implemented. Figure 1 shows a picture of our robotic agents. For the hardware description of our robots, see [13]. This paper is organized as follows: Section 2 presents the vision processing algorithm. In Section 3, we focus on the agent behaviors ranging from low-level individual behaviors, to coordinated, strategic, multiagent behaviors. Section 4 reports on our experiences using these robots in the RoboCup-97 robot competition and concludes.

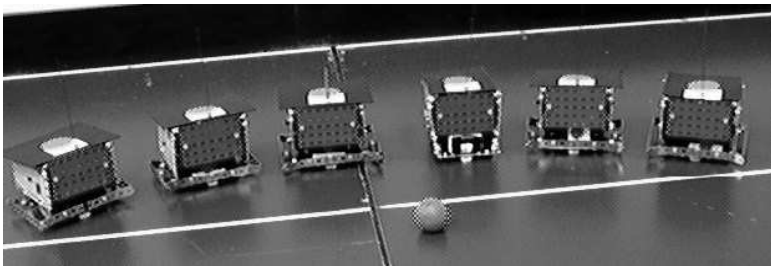

Figure 1: The CMUnited robot team that competed in RoboCup-97.

2 Real-Time Perception for Multiple Agents

The small-size robot league setup is viewed as an overall complete autonomous framework composed of the physical navigational robotic agents[1], a video camera over-looking the playing field connected to a centralized interface computer, and several clients as the minds of the small-size robot players. Figure 2 sketches the building blocks of the architecture.

The complete system is fully autonomous consisting of a well-defined and challenging processing cycle. The global vision algorithm perceives the dynamic

[1]For hardware details and specifications of the robots, please see [13].

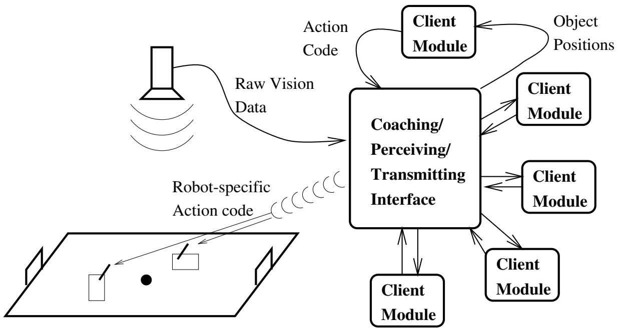

Figure 2: CMUnited Architecture with Global Perception and Distributed Reaction.

environment and processes the images, giving the positions of each robot and the ball. This information is sent to an off-board controller and distributed to the different agent algorithms. Each agent evaluates the world state and uses its strategic knowledge to decide what to do next. Actions are motion commands that are sent by the off-board controller through RF communication. Commands can be broadcast or sent directly to individual agents. Each robot has an identification binary code that is used on-board to detect commands intended for that robot. This complete system is fully implemented.

Although it may be possible to fit an on-board vision system onto robots of small size, in the interest of being able to quickly move on to strategic multiagent issues, we have opted for a global vision system. (In a future version of our robots, we may investigate distributed vision issues and incorporate on-board vision.)

The fact that perception is achieved by a video camera that over-looks the complete field offers an opportunity to get a global view of the world state. Although this setup may simplify the sharing of information among multiple agents, it presents a challenge for reliable and real-time processing of the movement of multiple moving objects—in our case, the ball, five agents on our team, and five agents in the opponent team.

This section focusses on presenting our vision processing algorithm whose accuracy makes it a major contribution towards the success of our team.

2.1 Detection

The vision requirements for robotic soccer have been examined by different researchers [8, 9]. Systems with on-board and off-board types have appeared in recent years. All have found that the reactiveness of soccer robots requires vision system with a high processing cycle time. However, due to the rich visual input, researchers have found that dedicated processors or even DSPs are often needed [2, 8].

The system we used at RoboCup-97 was surprisingly simple. A framegrabber with framerate transfer from a 3-CCD camera was used as the input. A relatively slow processor (166Mhz Pentium) was at the heart of the system, performing all computation.

The detection mechanism was kept as simple as possible. The RoboCup rules have specified well defined colors for different objects in the field and these were used as the major cue for object detection. The RoboCup rules specify a green color field with white markings at the side. Also, it specifies a yellow or blue colored ping-pong ball on the top of the robots, one color for each team. A single color patch on the robot is not enough to provide orientation information. Thus, an additional pink color patch was added to each robot. The ball is an orange golf ball. These colors can be differentiated in a straightforward manner in color-space.

The set of detected patches are unordered. The detected color patches on the tops of the robots are then matched by their distance. Knowing the constant distance between the team-color and the pink orientation patch, we were able to match patches that are this distance apart. Two distance-matched patches are marked as a robot.

Noise is inherent in all vision system. False detections in the current system are often of a magnitude of 100 spurious detections per frame. The system attempts to eliminate false detection using two different methods. First, color patches of size not matching the ones on the robots are discarded. This technique filters off most "salt and pepper" noise. Second, adding the distance matching mechanism describe above, all false detections are eliminated.

2.2 Data Association

Each of the robots is fitted with the same color tops and no attempts are made to differentiate them via color. Experience has shown that in order to differentiate 5 different robots by color, 5 different colors are needed. However, inevitable variations in lighting conditions over the area of the field are enough to make this detection mechanism unreliable.

Data association addresses the problem of retaining robot identification in subsequent frames. We devised an algorithm to retain association based on the spatial locations of the robots. During consecutive frames, association is maintained by searching for minimum distance criteria. Current robot positions are matched with the closest positions from the previous frame.

2.3 Tracking and Prediction

In the setting of a robot soccer game, the ability to detect merely the locations of objects on the field is often not enough. Like for real soccer players, it is often essential for robots to predict future locations of the ball (or even of the other players). We have used an Extended Kalman filter (EKF) for such a purpose[5]. The Kalman filter is very suitable for such a purpose since the detection of the ball's location is noisy.

The EKF is a recursive estimator for a possibly non-linear system. The goal of the filter is to estimate the state of a system. The state is usually denoted as an n-dimensional vector x. A set of equations is used to describe the behavior of the system, predicting the state of the system as:

$$x_{k+1} = f(x_k, u_k, w_k),$$

where $f(\cdot)$ is a non-linear function which represents the behavior of the non-linear system, u_k is the external input to the system and w_k is a zero-mean, Gaussian random variable with covariance matrix Q_k. w_k captures the noise in the system and any possible discrepancies between the physical system and the model. The subscript k denotes the value of a variable at time step k.

The system being modeled is being observed (measured). The observations can also be non-linear:

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

where z_k is the vector of observations and $h(\cdot)$ is the non-linear measurement function, and v_k is another zero-mean, Gaussian random variable with covariance matrix R_k. It captures any noise in the measurement process.

The EKF involves a two-step iterative process, namely *update* and *propagate*. The current best estimate of the system's state $\hat{x}$ and its error covariance is computed on each iteration. During the update step, the current observations are used to refine the current estimate and recompute the covariance. During the propagate step, the state and covariance of the system at the next time step are calculated using the system's equations. The process then iteratively repeats, alternating between the update and the propagate steps.

Through a careful adjustment of the filter parameters modelling the system, we were able to achieve successful tracking and, in particular prediction of the ball trajectory, even when sharp bounces occur.

Our vision processing approach worked perfectly during the RoboCup-97 games. We were able to detect and track 11 objects (5 teammates, 5 opponents and a ball). The prediction provided by the EKF allowed the goalkeeper to look ahead in time and predict the best defending position. During the game, no goals were suffered due to miscalculation of the predicted ball position.

3 Multiagent Strategy Control

We achieve multiagent strategy through the combination of accurate individual and collaborative behaviors. Agents reason through the use of persistent reactive behaviors that are developed to aim at reaching team objectives.

3.1 Single-agent Behaviors

In order to be able to successfully collaborate, agents require robust basic skills. These skills include the ability to go to a given place on the field, the ability to direct the ball in a given direction, and the ability to intercept a moving ball. All of these skills must be executed while avoiding obstacles such as the walls and other robots.

The navigational movement control is done via closed-loop reactive control. The control strategy follows a modified version of a simple Braitenburg vehicle [4]. The Braitenburg *love vehicle* defines a reactive control mechanism that directs a differentially driven robot to a certain destination point (goal). A similar behavior is required in the system; however, the love vehicle's control mechanism is too simplistic and, in some start configurations, tends to converge to the goal very slowly. We devised a modified set of reactive control formulae that allows for effective adjustment of the control trajectory:

$$\begin{aligned} translational\ velocity &= \alpha \cdot \sin\theta \\ rotational\ velocity &= \beta \cdot \cos(\theta) \end{aligned}$$

where θ is the direction of the target relative to the robot, α and β are the base translational and rotational velocities, respectively. This set of control formulae differs from the love vehicle in that it takes into account the orientation of the robot with respect to the goal and explicitly adds rotational control.

3.1.1 Ball Handling

If a robot is to accurately direct the ball towards a target position, it must be able to approach the ball from a specified direction. Using the ball prediction from the vision system, the robot aims at a point on the far side of the target position. The robots are equipped with two methods of doing so:

Ball Collection: Moving behind a ball and knocking it towards the target.

Ball Interception: Waiting for the ball to cross its path and then intercepting the moving ball towards the target.

When using the ball collection behavior, the robot considers a line from the target position to the ball's current or predicted position, depending on whether or not the ball is moving. The robot then plans a path to a point on the line and behind the ball such that it does not hit the ball on the way and such that it ends up facing the target position. Finally, the robot accelerates to the target. Figure 3 illustrates this behavior.

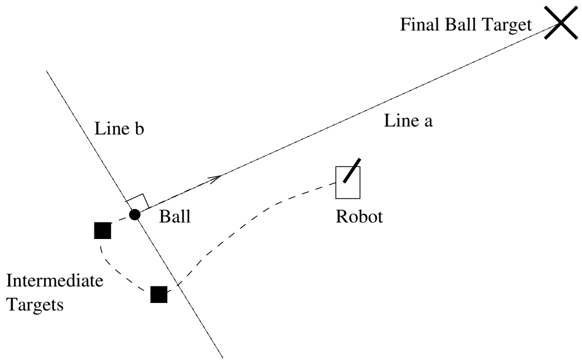

Figure 3: **Ball Collection–**The robot computes the line from the ball to the target (line a) as well as the line through the ball and perpendicular to this line (line b). Whenever the robot is on the same side of line b as the target, it aims for an intermediate target to the side of the ball so that it avoids hitting the ball away from the target. Otherwise, the robot aims for a point directly behind the ball along line a. Once there, it accelerates towards the target.

When using the ball interception behavior (Figure 4), on the other hand, the robot considers a line from *itself* to the target position and determines where the ball's path will intersect this line. The robot then positions itself along this line so that it will be able to accelerate to the point of intersection at the same time that the ball arrives.

In practice, the robot chooses from between its two ball handling routines based on whether the ball will eventually cross its path at a point such that the

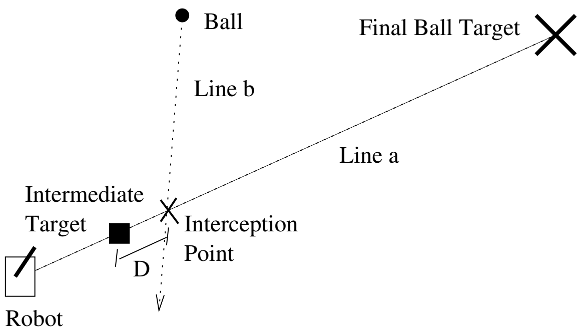

Figure 4: **Ball Interception–**The robot computes the intersection of the line between itself and the target position (line a) and the ball's line of trajectory (line b). The robot then positions itself at a fixed distance (*D*) behind the intersection point, either moving forwards or backwards to get there. Knowing the time *T* required to accelerate from a stopped position to distance *D*, and also knowing the ball's velocity, the robot accelerates towards the final target when the ball is time *T* away from the interception point.

robot could intercept it towards the goal. Thus, the robot gives precedence to the ball interception routine, only using ball collection when necessary. When using ball collection, it actually aims at the ball's predicted location a fixed time in the future so as to eventually position itself in a place from which it can intercept the ball towards the target.

3.1.2 Obstacle Avoidance

In the robotic soccer field, there are often obstacles often between the robot and its goal location. Our robots try to avoid collisions by planning a path around the obstacles. Due to the highly dynamic nature of this domain, our obstacle avoidance algorithm uses closed-loop control by which the robots continually replan their goal positions around obstacles. In the event that an obstacle blocks the direct path to the goal location, the robot aims to one side of the obstacle until it is in a position such that it can move directly to its original goal. Rather than planning the entire path to the goal location at once, the robot just looks ahead to the first obstacle in its way under the assumption that other robots are continually moving around. Using the reactive control described above, the robot continually reevaluates its target position. For an illustration, see Figure 5.

Even with obstacle avoidance in place, the robots can occasionally get stuck against other robots or against the wall. Particularly if opponent robots do not use obstacle avoidance, collisions are inevitable. When unable to move, our robots

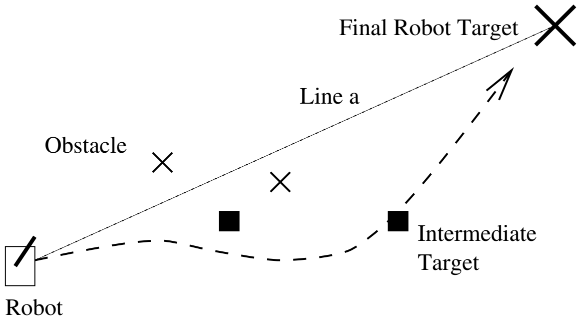

Figure 5: **Obstacle Avoidance–**The robot starts by trying to go straight towards its final target along line a. When it comes across an obstacle within a certain distance of itself and of line a, it aims at an intermediate target to the side, and slightly beyond the obstacle. The robot goes around the obstacle the short way, unless it is at the edge of the field. Using reactive control, the robot continually recomputes line a until the obstacle is no longer in its path. As it comes across further obstacles, it aims at additional intermediate targets until it obtains an unobstructed path to the final target.

identify the source of the problem as the closest obstacle and "unstick" themselves by moving away. Once free, normal control resumes.

3.2 Multiagent Behaviors

Although the single-agent behaviors are very effective when just a single robot is on the field, if all five robots were simultaneously chasing the ball and trying to shoot it at the goal, chaos would result. In order to achieve coordinated multiagent behavior, we organize the five robots into a flexible team structure.

The team structure, or *formation*, defines a set of roles, or *positions* with associated behaviors. The robots are then dynamically mapped into the positions. Each robot is equipped with the knowledge required to play any position in each of several formations.

The positions indicate the areas of the field which the robots should move to in the default situation. There are also different *active modes* which determine when a given robot should move to the ball or do something else instead. Finally, the robot with the ball chooses whether to shoot or pass to a teammate using a passing evaluation function.

These high-level, multiagent behaviors were originally developed in simulation and then transferred over to the robot-control code. Only the run-time passing evaluation function was redefined. Further details, particularly about the flexible team structures, are available in [10].

3.2.1 Positions, Formations, and Active Modes

Positions are defined as flexible regions within which the player attempts to move towards the ball. For example, a robot playing the "right-wing" (or "right forward") position remains on the right side of the field near the opponents' goal until the ball comes towards it. Positions are classified as defender/midfielder/forward based on the locations of these regions. They are also given behavior specifications in terms of which other positions should be considered as potential pass-receivers (see Section 3.2.2).

At any given time each of the robots plays a particular position on the field. However, each robot has all of the knowledge necessary to play any position. Therefore the robots can—and do—switch positions on the fly. For example, robots A and B switch positions when robot A chases the ball into the region of robot B. Then robot A continues chasing the ball, and robot B moves to the position vacated by A.

The pre-defined positions known to all players are collected into formations, which are also commonly known. An example of a formation is the collection of positions consisting of the goalkeeper, one defender, one midfielder, and two attackers. Another possible formation consists of the goalkeeper, two defenders and two attackers. For illustration, see Figure 6.

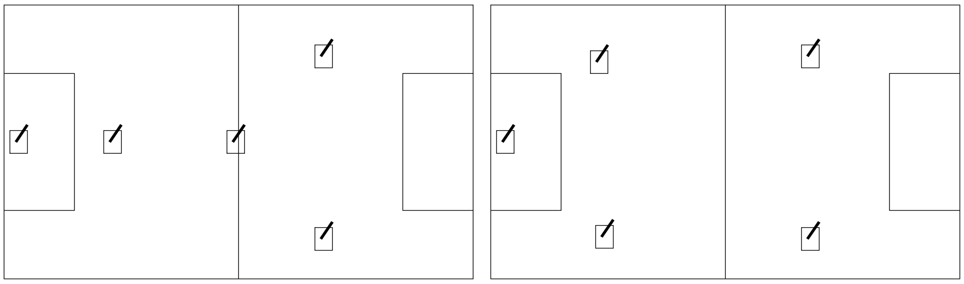

Figure 6: Two different defined formations. Notice that several of the positions are reused between the two formations.

As is the case for position-switches, the robots switch formations based on pre-determined conditions. For example, if the team is losing with very not much time left in the game, the robots would switch to a more offensive formation. On the other hand, if winning, they might choose a defensive formation. The precise conditions for switching positions and formations are decided upon in advance, in what we call a "locker-room agreement," [10] in order to eliminate the need for complex on-line negotiation protocols.

Although the default action of each robot is to go to its position and face the ball, there are three *active modes* from which the robot must choose. The default position-holding behavior occurs when the robot is in an *inactive* state. However, when the ball is nearby, the robot changes into an *active* state. In the active state, the robot moves towards the ball, attempting either to pass it to a teammate or to shoot it towards the goal based on an evaluation function that takes into account teammate and opponent positions (see Section 3.2.2). A robot that is the intended receiver of a pass moves into the *auxiliary* state in which it tries to intercept a moving ball towards the goal. Our current decision function sets the robot that is closest to the ball into the active state; the intended receiver robot (if any) into the auxiliary state; and all other robots into the inactive state.

3.2.2 Run-time Evaluation of Collaborative Opportunities

One of CMUnited-97's main features is the robots' ability to collaborate by passing the ball. When in active mode, the robots use an evaluation function that takes into account teammate and opponent positions to determine whether to pass the ball or whether to shoot. In particular, as part of the formation definition, each position has a set of positions to which it considers passing. For example, a defender might consider passing to any forward or midfielder, while a forward would consider passing to other forwards, but not backwards to a midfielder or defender.

For each such position that is occupied by a teammate, the robot evaluates the pass to that position as well as evaluating its own shot. To evaluate each possible pass, the robot computes the *obstruction-free-index* of the two line segments that the ball must traverse if the receiver is to shoot the ball (lines b and c in Figure 7). In the case of a shot, only one line segment must be considered (line a). The *value* of each possible pass or shot is the product of the relevant obstruction-free-indices. Robots can be biased towards passing or shooting by further multiplying the values by a factor determined by the relative proximities of the active robot and the potential receivers to the goal. The robot chooses the pass or shot with the maximum value. The obstruction-free-index of line segment l is computed by the following algorithm (variable names correspond to those in Figure 7):

1. *obstruction-free-index* = 1.
2. For each opponent O:
   - Compute the distance x from O to l and the distance y along l to l's origin, i.e. the end at which the ball will be kicked by the robot (See Figure 7).

- Define constants *min-dist* and *max-denominator*. Opponents farther than *min-dist* from l are not considered. When discounting *obstruction-free-index* in the next step, the y distance is never considered to be larger than *max-denominator*. For example, in Figure 7, the opponent near the goal would be evaluated with $y =$ *max-denominator*, rather than its actual distance from the ball. The reasoning is that beyond distance *max-denominator*, the opponent has enough time to block the ball: the extra distance is no longer useful.
- if $x <$ *min-dist* and $x < y$,
  obstruction-free-index *= $x/MIN($*max-demoninator*,$y)$.

3. return *obstruction-free-index*.

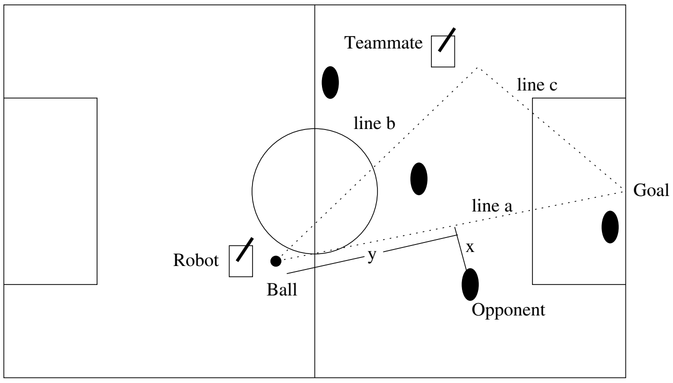

Figure 7: **Pass Evaluation—**To evaluate a pass to a teammate, the robot considers how open the paths are from the ball to the teammate (line b) and from the teammate to the goal (line c). When evaluating shots, it considers the line from the ball to the goal (line a). For each opponent and each line segment, it computes the opponent's distance to the segment (x) and along the segment to the origin. The smaller x is and the larger y is, the easier it would be for the opponent to intercept the ball. Note that some opponents would cause discounts in the values of passes along more than one segment.

Thus the obstruction-free-index reflects how easily an opponent could intercept the pass or the subsequent shot. The closer the opponent is to the line and the farther it is from the ball, the better chance it has of intercepting the ball.

3.2.3 The Goalkeeper

The goalkeeper robot has both special hardware and special software. Thus, it does not switch positions or active modes like the others. The goalkeeper's physical frame is distinct from that of the other robots in that it is as long as allowed under the RoboCup-97 rules (18cm) so as to block as much of the goal as possible. The goalkeeper's role is to prevent the ball from entering the goal. It stays parallel to and close to the goal, aiming always to be directly even with the ball's lateral coordinate on the field.

Ideally, simply staying even with the ball would guarantee that the ball would never get past the goalkeeper. However, since the robots cannot accelerate as fast as the ball can, it would be possible to defeat such a behavior. Therefore, the goalkeeper continually monitors the ball's trajectory. In some cases it moves to the ball's predicted destination point ahead of time.

The decision of when to move to the predicted ball position is both crucial and difficult, as illustrated in Figure 8. Our current decision function is as follows:

1. The goalkeeper always stays in front of the goal. If the following steps indicate that it should move beyond the goal, it stays at the closest edge of the goal.
2. If all of the following conditions are true, the goalkeeper moves to the ball's predicted location (dotted in Figure 8):
   - The ball is moving faster than a minimum threshold speed;
   - The ball is not in Zone Z of Figure 8 (on either side of the field).
   - The ball is moving towards a point either within the goal, or within a minimum distance from the goal.
3. Otherwise, the goalkeeper stays even with the ball's y coordinate (see Figure 8).

4 Discussion and Conclusion

CMUnited-97 successfully demonstrated the feasibility and effectiveness of teams of multiagent robotic systems. Within this paradigm, one of the major challenges was to "close the loop," i.e., to integrate all the different modules, ranging from perception to strategic multiagent reasoning. CMUnited is an example of a fully implemented multiagent system in which the loop is closed. In addition, we were implemented interesting strategic behaviors, including agent collaboration and real-time evaluation of alternative actions.

It is generally very difficult to accumulate significant scientific results to test teams of robots. Realistically, extended runs are prohibited by battery limitations and the difficulty of keeping many robots operational concurrently. Furthermore, we only had the resources to build a single team of five robots, with one spare

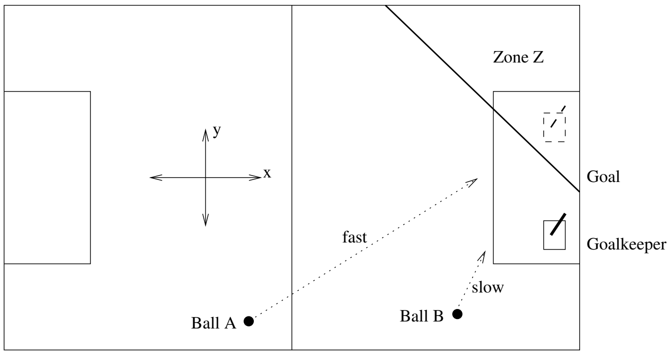

Figure 8: **Goalkeeping—**Ideally, the goalkeeper should always be even with the ball's y coordinate. However, since the robot cannot accelerate as quickly as the ball can move, it must sometimes move to the ball's *predicted* location. Such a case is illustrated by ball A: The goalkeeper should move to the dotted position. On the other hand, ball B indicates a situation in which the goalkeeper should *not* move to the ball's predicted location: were the goalie to move to the dotted position, an opponent could easily intercept the ball into the goal. Thus, we needed to create a decision function to choose between following the ball's y coordinate and moving to the ball's predicted location.

so far. Therefore, we offer a restricted evaluation of CMUnited based on the results of four effective 10-minute games that were played at RoboCup-97. We also include anecdotal evidence of the multiagent capabilities of the CMUnited-97 robotic soccer team.

The CMUnited-97 robot team played games against robot teams from Nara Institute of Science and Technology (NAIST), Japan; University of Paris VI, France (team name "MICROB"); and University of Girona, Spain. The results of the games are given in Table 1. In total, CMUnited-97 scored thirteen goals, allowing only one against. The one goal against was scored by the CMUnited goalkeeper against itself, though under an attacking situation from France. We refined the goalkeeper's goal behavior, as presented in Section 3.2.3, following the observation of our goalkeeper's error.

As the matches proceeded, spectators noticed many of the team behaviors described in Section 3.2. The robots switched positions during the games, and there were several successful passes. The most impressive goal of the tournament was the result of a 3-way passing play: one robot passed to a second, which passed to a third, which shot the ball into the goal.

In general, the robots' behaviors were visually appealing and entertaining to

Opponent	Score
NAIST	5-0
MICROB	3-1
U. of Girona	2-0
NAIST (finals)	3-0
TOTAL	13-1

Table 1: The scores of CMUnited's games in the small robot league of RoboCup-97. CMUnited-97 won all four games.

the spectators. Several people attained a first-hand appreciation for the difficulty of the task as we let them try controlling a single robot with a joystick program that we developed. All of these people (several children and a few adults) found it quite difficult to maneuver a single robot well enough to direct a ball into an open goal. These people in particular were impressed with the facility with which the robots were able to pass, score, and defend.

We are aware that many issues are clearly open for further research and development. We are currently systematically identifying them and addressing them towards our next team version. In particular, we are planning on enhancing the robot's behaviors by using machine learning techniques. We are currently developing techniques to accumulate and analyze real robot data.

References

[1] Sorin Achim, Peter Stone, and Manuela Veloso. Building a dedicated robotic soccer system. In *Proceedings of the IROS-96 Workshop on RoboCup*, November 1996.

[2] M. Asada, S. Noda, S. Tawaratumida, and K. Hosoda. Purposive behavior acquisition for a real robot by vision-based reinforcement learning. *Machine Learning*, 23:279–303, 1996.

[3] Minoru Asada, Yasuo Kuniyoshi, Alexi Drogoul, Hajime Asama, Maja Mataric, Dominique Duhaut, Peter Stone, and Hiroaki Kitano. The robocup physical agent challenge: Phase-i. *To appear in Applied Artificial Intelligence (AAI) Journal*, 1998.

[4] V. Braitenburg. *Vehicles – experiments in synthetic psychology*. MIT Press, 1984.

[5] Kwun Han and Manuela Veloso. Physical model based multi-objects tracking and prediction in robosoccer. In *Working Note of the AAAI 1997 Fall Symposium*. AAAI, MIT Press, 1997.

[6] Hiroaki Kitano, Yasuo Kuniyoshi, Itsuki Noda, Minoru Asada, Hitoshi Matsubara, and Ei-Ichi Osawa. Robocup: A challenge problem for ai. *AI Magazine*, 18(1):73–85, Spring 1997.

[7] Hiroaki Kitano, Milind Tambe, Peter Stone, Manuela Veloso, Silvia Coradeschi, Eiichi Osawa, Hitoshi Matsubara, Itsuki Noda, and Minoru Asada. The robocup synthetic agent challenge 97. In *Proceedings of the Fifteenth International Joint Conference on Artificial Intelligence*, San Francisco, CA, 1997. Morgan Kaufman.

[8] Michael K. Sahota, Alan K. Mackworth, Rod A. Barman, and Stewart J. Kingdon. Real-time control of soccer-playing robots using off-board vision: the dynamite testbed. In *IEEE International Conference on Systems, Man, and Cybernetics*, pages 3690–3663, 1995.

[9] Randy Sargent, Bill Bailey, Carl Witty, and Anne Wright. Dynamic object capture using fast vision tracking. *AI Magazine*, 18(1):65–72, Spring 1997.

[10] Peter Stone and Manuela Veloso. Task decomposition and dynamic role assignment for real-time strategic teamwork. In *submitted to the Third International Conference on Multi-Agent Systems*, November 1997. Draft available by request or from http://www.cs.cmu.edu/ pstone/pstone-papers.html.

[11] Peter Stone and Manuela Veloso. Using decision tree confidence factors for multiagent control. In *Proceedings of the First International Workshop on RoboCup*, Nagoya,Japan, August 1997.

[12] Peter Stone and Manuela Veloso. A layered approach to learning client behaviors in the robocup soccer server. *To appear in Applied Artificial Intelligence (AAI) Journal*, 1998.

[13] Manuela Veloso, Peter Stone, Kwun Han, and Sorin Achim. Cmunited: A team of robotic soccer agents collaborating in an adversarial environment. In *Proceedings of the First International Workshop on RoboCup*, Nagoya,Japan, August 1997.