Robust Team-Play in Highly Uncertain Environments

(Short Paper)

Henry Work
Bowdoin College
8650 College Station
Brunswick, ME 04011
hwork@bowdoin.edu

Eric Chown
Bowdoin College
8650 College Station
Brunswick, ME 04011
echown@bowdoin.edu

Tucker Hermans
thermans@bowdoin.edu

Jesse Butterfield
jbutterf@cs.brown.edu

ABSTRACT

Effective teamwork in highly dynamic environments requires a delicate balance between giving agents the autonomy to act and react on their own and restricting that autonomy so that the agents do not work at cross purposes. In this article we describe the problems involved in coordinating behavior based upon a highly dynamic object, a soccer ball, for agents with sensing and communications limitations. We then present a system for coping with these problems and examine its success in light of its performance at the RoboCup 2007 championship.

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]: Robotics—*performance measures*

General Terms

Algorithms, Design, Performance

Keywords

RoboCup, cooperation, localization

1. INTRODUCTION

Soccer presents a challenging environment for multi-agent systems. Agents must cooperate to defeat a group of opposing agents. The soccer ball's movement is highly unpredictable: it is constantly being propelled by different players; it is affected by spin and uneven surfaces; and occasionally it "teleports" when moved by referees. Control of the ball is crucial to success in soccer, so it is of paramount importance for the agents on a team to share information and coordinate their roles. At the same time, however, agents must also have the freedom to react quickly without waiting to hear from their teammates.

In soccer teams try and maximize their possession of the ball. It follows that in order to gain possession, it is strategically important for an agent to get to the ball as quickly as possible [3]. We define *chase time* as the estimated time it will take a robot to get to the soccer ball. Minimizing *chase time* is our goal, and can be achieved in a variety of ways. Most obviously, a robot's walk speed can be optimized. However, In the Standard Platform League, where teams use equivalent hardware, machine learning has been applied successfully to this problem to the point where most of the top teams move at virtually identical speeds. To win at robot soccer, our team has two core beliefs: one, that every agent must have current, accurate knowledge of where the soccer ball is on the field at all times; and two, that only one agent approach the ball at any given time.

We begin this article with a discussion of RoboCup and the Standard Platform League. Second, we discuss our use of a hybrid ball model. Next, we present our coordination system, developed for use by the Northern Bites in the Standard Platform league. Finally, we finish with results from competition.

Cite as: Robust Team-Play in Highly Uncertain Environments (Short Paper), Work, H., Chown, E., Hermans, T, and Butterfield, J., *Proc. of 7th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2008)*, Padgham, Parkes, Müller and Parsons (eds.), May, 12-16., 2008, Estoril, Portugal, pp. XXX-XXX.

2. THE ROBOCUP DOMAIN

The system we describe is a vital component for a competitor in the RoboCup Standard Platform League (formerly the Four-Legged League). The robot platform is the Sony Aibo. Although an inexpensive and fairly robust robot, the Aibo presents major sensing challenges.

For example, accurate distance estimation is difficult due to the robots' low resolution CMOS cameras (208x160 pixels). With narrow viewing angles (57 degree horizontal, 45 degree vertical), the robots must spend a significant amount of their time searching for the ball (especially when the ball is close to the robot). There are also significant blurring issues with the low shutter speed of the Aibo's camera, as well as a documented chromatic distortion problem [2].

As with vision, accurately modeling odometry is not an easy feat. For example, there is no standard carpet surface within the Standard Platform League, and thus odometry varies from location to location. Further, during matches, robots frequently push one another, thereby shifting their positions. Odometry in our domain is challenging, but complements vision as the second, necessary half of the problem. Despite these sensing challenges with the game and the platform, accurate self-localization is possible using standard techniques such as Monte Carlo methods or Kalman Fil-

ters [3, 5, 7]. Nonetheless, crucial localization errors impact games on a regular basis.

The other major hurdle facing coordination behavior in the RoboCup domain is poor communication. The competition environments are well known for being high in packet loss and packet lag time [6]. Packet delays on the order of seconds (and occasionally minutes) are not unusual due to the amount of wireless interference at competition venues. Communication systems must be specifically designed to withstand less-than-ideal conditions (and not just in the lab). These conditions necessarily impact the sort of dynamic role switching inherent in soccer.

3. SHARING BALL INFORMATION

The ball localization problem requires a solution that combines local and global data. Most teams in our league have agreed on the necessity of sharing information between agents, but have differed in approaches. One approach is to maintain two completely different ball models, based on particle filters, and switch between the two depending on the quality of local data [5]. Others consider teammate information for merging and pruning after arriving at their own ball estimates [7]. Our approach is similar to other teams that use teammate information as input to the ball filter, but only in the absence of more reliable local information [3].

Reliable ball estimates start with reliable self-localization estimates. Our system is an implementation of an Extended Kalman Filter [1], or EKF, which is popular in the Standard Platform League [3, 7, 4].

Communication is the next component of our localization system. In our implementation, we use a broadcast UDP framework. Each agent broadcasts information at about six packets a second out to its fellow agents (at competition, only 3-4 may be received due to packet-loss). The broadcast rate was deduced through experimentation: sending more packets worsened our overall packet-loss, while sending fewer made our response to teammates unacceptably slow. Further, to keep our communication as fast and robust as possible, we keep overhead extremely low.

Sensing error and communication lag must play an important role in the architecture of a system that cooperatively tracks a moving object across multiple agents. High sensing error relayed from other agents can potentially ruin accurate locally-derived data. Alternately, if the environment operates at a faster pace than the communications network, then individual agents cannot depend on shared information being up-to-date. Unfortunately, both of these problems exist in our competition environment. In our lab, with a highly tuned vision calibration and a low latency, low packet-loss wireless network, our robots reliably perform very well. At competition, however, new fields and lighting conditions (bad for odometry and vision, respectively) can often greatly reduce the accuracy of localization. Typical competition setup also involves severe WLAN interference problems. Therefore, our system necessarily needs to move away from a full sharing model.

In our system, each agent will either ignore or integrate shared ball data depending on the quality of its locally derived data. In robot soccer terms, this translates to: 'If I see the ball, I'll ignore my teammates' and 'If I haven't seen the ball recently, I'll listen to my teammates.' The primary qualifier for deciding when to use information reported by fellow agents is a buffered time threshold of three frames, to handle the frequent case when an agent loses the ball only momentarily (typically due to ball being occluded). Another qualifier is the uncertainty of information relayed by other agents; again in soccer terms: 'If I haven't seen a ball recently, and my teammate has, only trust my teammate if it is confident.' This ball uncertainty is calculated by the EKF and shared in every packet. If shared information is used, it is fed into the localization system as if it were locally generated.

It is important to note here that the information is being broadcast by each agent regardless of whether a particular agent decides to use it. Also, note that this approach depends heavily on the quality of each agent's sensors; if an agent's sensors are reporting falsely, its fellow agents' information will not help. This trade-off we believe is necessary due to the fast-paced game and unreliable communications.

The system we describe results in a group of n agents with n distinct ideas of the ball at any given time t. The goal in terms of implementation is to make these estimates, while distinct, as consistently accurate as possible. Our system works like this: if at t all n agents see the ball, then none use shared information, but if none of the n agents see the ball, then ball estimates are unaffected and no shared information is used. In the majority of situations, this system provides a strong cycle of information: agents who see the ball improve the ball knowledge of those agents that do not see the ball.

4. ROBUST COORDINATION

The hybrid ball model creates a system where each robot has an independent notion of the ball's location. Any cooperative behavior in this environment needs to work despite disagreements among team members about crucial factors such as who is closest to the ball. The discussed communication issues and the speed of the game serve to further limit the sorts of decision-making processes available for dynamically-assigning specific roles to enable a team to function smoothly. Following other teams, we believe it is not reasonable to implement a negotiation strategy or to have one robot make decisions unilaterally [5, 3]. The time required to assign roles and the need for quick responses trumps any possible advantages of a centralized approach.

With our primary goal of minimizing *chase time* to the ball, waiting even half a second for orders can impact possession. To be maximally effective, a robot cannot wait for a reply from a teammate on whether the robot should chase a ball coming toward it. Each agent must be able to switch its role independently as necessary, while using the best information about its teammates' whereabouts. With these issues in mind our system is based upon several principles. First, we want it to work well even in the face of poor communication. Second, it must be highly responsive to cope with the dynamic environment of soccer. Finally, even though individual agents must be able to switch roles on the fly without confirmation from their teammates, it is mandatory that agents keep their mutual interference to a minimum.

Our general soccer strategy is for one, and only one, robot to chase the ball at any given time (and that any robot, save the *goalie*, can become *chaser*). Our cooperative system is comprised of four roles: *chaser*, *attacker*, *defender*, and *goalie*. *Chase time* is calculated using odometry and assumes no delays (obstruction, losing the ball, etc). Each agent independently calculates its own *chase time* every frame and

compares it to the last reported *chase time* of all other agents. The agent with the lowest *chase time* should be the *chaser* and therefore maximize the chances of our team gaining possession. To reflect our overall aims of scoring a goal, we reduce the assumed *chase time* in special circumstances – such as an agent lined up behind the ball going towards the opponent's goal. Lowering the *chase time* improves the likelihood that an agent not closest to the ball, but still in the best position to score, will become *chaser*.

A team of mobile agents should be able to independently make decisions and have their overall decision be consistent. For example, in a situation where the ball is 50 cm from one agent, 150 cm from another, and 300 cm from a third, choosing who chases the ball may be easy. However, when three robots are roughly equidistant to the ball, the decision making process becomes more difficult. Information lag and system noise incurred from sensor data may cause the agents' divergent world models to bring about different conclusions as to which robot has the shortest *chase time*. To combat this error while maintaining our non-negotiated system, we use a tiebreaker, which draws its inspiration from real sports, where certain players "call off" others (i.e. a goalie can call off a defender). For our system we use player numbers (each robot has a unique number 1-4), such that in a tie-break situation the higher player number can call off a lower player number.

If Robot A believes it might be the fastest to the ball (it is within a small threshold ϵ of the minimum *chase time* for all the robots), it will start pursuing and calling the ball ("I got it!"). Robot A will continue to chase the ball until a higher ranked robot calls it off, and/or any lower ranked robots receiving Robot A's call discontinue chasing. Thus Robot A is committed to chasing the ball. It cannot stop chasing and calling the ball until its *chase time* is outside a larger threshold δ (that is $\delta > \epsilon$). This prevents hysteresis, where robots oscillate back and forth between roles due to small changes in the data. There is a third important threshold, λ, which ensures that a robot should stop listening to a higher ranked teammate if its *chase time* is less than that teammate's *chase time* by the value of λ. This ensures that when there is a discrepancy between local information and communicated information the robot relies on its local information. When things rapidly change on the field, a robot must not wait for a message from the current *chaser* with higher rank before it acts. In summary Robot A will chase the ball if:

1. $chasetime(A) - min(chasetime(A, B)) < \epsilon$ and no higher robot is calling off A, or
2. $chasetime(A) - min(chasetime(A, B)) < \delta$ and it was already chasing and no higher robot is calling off A, or
3. $chasetime(A) < chasetime(B) - \lambda$ where the Robot B is the higher ranked robot calling off A

Each threshold controls a feature of the robots' cooperation. Increasing ϵ makes it more likely that a robot who is farther from the ball will ultimately end up being *chaser*, but makes it less likely that no robot will be *chaser*. δ controls how willing robots are to switch roles. Increasing the δ value decreases how often the robots will switch roles, which can leave the wrong robot chasing the ball, but protects against robots oscillating back and forth about who should chase. λ controls how much a robot should rely on local information. Increasing λ makes it less likely two robots will chase the ball, but slows down reactions to a ball suddenly being closer to a non-*chaser* robot.

After deciding on which robot should become the *chaser*, the remaining field players must decide which robot is to become the *defender* and which should become the supporting *attacker*. The decision making process for this issue is the same as in determining the *chaser*, only the determining metric is *distance to own goal* instead of *chase time*. We structure the tie-breaking thresholds about defense to ensure there is always a *defender* when communicated data deteriorates.

To summarize, our goal was to build a robust system for dynamic role selection to work within a domain, where all of the agents have theoretically equivalent skills and thus can take on any role at any given time. We have taken specific steps to combat three major problems inherent in dynamic role switching with a decentralized decision making process. These problems are:

1. *hysteresis* - where robots switch roles back and forth in a cycle
2. *deadlock* - where robots apparently have equal claim on an important role, and
3. *information lag* - where information about other robots lags behind what a robot can directly sense.

Probably the most crucial element of our solution is the idea of error buffering. For example, when a robot determines if it should be the *chaser* it does not use a strict threshold algorithm, but thresholds based on the fact that error is built into the system. The error being buffered can also be temporal as a robot waits for updates from its teammates. It might seem as if all of these hedges could easily lead to poor behavior such as swarming, but what typically happens in a real game is that the robots move into an initial formation and the formation itself (on the basis of the relative spacing between robots) ensures that most decisions can be made cleanly. The only breakdowns of our team play in competition have occurred in the case of a catastrophic loss of communication. During such times coordinated team play is nearly impossible unless every agent can localize its teammates independently (something no team in our league can do). In our system agents continually monitor how long it has been since they have received any information from their teammates. If a robot determines that it is no longer communicating with teammates default behaviors are performed. Such default behaviors cannot be robot specific, (e.g. player 1 is the *chaser*) since some robots may not be functioning. For this reason our default behaviors amount to taking a defensive position unless the ball comes within a fixed range.

5. RESULTS

RoboCup provides some obvious metrics to measure the total quality of the team: wins, losses, goals scored, and goals against. However, there are many factors beyond the team play described in this article that impact whether or not goals are scored. For instance the robots' walk speed, ability to effectively kick, effectiveness of goal keeping, etc. In addition there are wide variations in quality among RoboCup

Table 1: Analysis of how often our team got to the ball first on different out-of-bounds situations. The first number in each column is how many times our team got to the ball first, the second number is the total number of times the ball went out of bounds.

Opponent	Sideline	Endline	Total
Team A	18 / 30	11 / 15	29 / 45
Team B	17 / 27	7 / 8	24 / 35
Team C	18 / 31	14 / 16	32 / 47
Totals	53 / 88	32 / 39	85 / 127

teams. Teams that are brand new to the league, for example, cannot be expected to provide strong competition. With those things in mind we have created several statistics to help judge the effectiveness of our coordination strategy.

We have identified two key types of events relevant to the notion of ball control that can be easily extracted by watching tapes of the competition. The first is a side effect of the rules. When the ball is knocked out of bounds in RoboCup it is replaced on the field according to which team knocked it out and whether it was knocked out over a sideline or an endline. It is clearly an advantage to be the first team to the ball after it is replaced. Since at the highest levels of competition there is not much difference in robot speed, the likelihood of getting to the ball first is dependent mainly on positioning – which comes as a direct result of coordinated team play. In order to avoid skewing our results we have limited our analysis to games in the final three rounds of the tournament where only the top competitors remained. The results are summarized in Table 1. Overall we got to the ball first more than twice as often as our opponents when the ball was replaced after going out-of-bounds. When the out-of-bounds occurred over an endline this shot up to a ratio better than four to one.

Another quality metric is based upon style of play in RoboCup. The top teams in RoboCup generally advance the ball by first having a robot trap the ball under its chin and then, when control has been gained, kicking it down the field. If our goal is to get to the ball more quickly, it stands to reason that more successful teams will attempt more traps. We have analyzed how often each team attempted to trap (and how often the traps were successful). Since robot speeds are relatively equal, this is another case where positioning should be the deciding factor. Once again we will limit our results to matches played from the quarterfinals on. The results are summarized in Table 2. The results show that our team attempted traps more than twice as often as our opponents. We also trapped successfully at a similar ratio to our opponents. The successful trap ration reflects the fact that many of the basic soccer skills in RoboCup are close to being optimized for the top teams.

We attribute our team's ability to get to the ball faster than our opponents as being the primary factor enabling us to win the RoboCup championship.

Table 2: Number of traps attempted, and successes, by our team and our opponents by game. The first two columns represent the figures for our team, the next two columns for our opponents.

Opponent	Attempts	Success	Attempts	Success
Team A	93	67	49	32
Team B	123	71	39	19
Team C	100	67	57	45
Total	316	205	145	96

6. CONCLUDING REMARKS

It might seem like many of the problems outlined in this article will evaporate as robot sensing and communication improves. Experience in robotics has shown that this is not the case, that improvements in sensing and communications either simply move the problem down to a finer level of detail, or lead to increasingly difficult problems being tackled where the relative limitations go right back into effect. Further, it is a fundamental property of the real world that things break down at least occasionally. The system presented here works to balance an agent's ability to act autonomously against its need to work with its teammates. It also strives to balance performance in optimal laboratory conditions against the ability to handle more adverse real-world conditions.

7. ACKNOWLEDGMENTS

This work was partially sponsored by the NSF under Grant No. 0092605. The authors gratefully acknowledge that support. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation (NSF).

8. REFERENCES

[1] R. Kalman. A new approach to linear filtering and prediction problems. *Transactions of the ASME - Journal of Basic Engineering*, 82:35–45, 1960.

[2] W. Nisticò and T. Röfer. Improving percept reliability in the sony four-legged robot league. In *RoboCup 2006: Robot Soccer World Cup X*. Springer Berlin, 2006.

[3] M. Quinlan, S. Nicklin, K. Hong, N. Henderson, S. Young, T. Moor, R. Fisher, P. Douangboupha, S. Chalup, R. Middleton, and R. King. The 2005 nubots team report. Technical report, University of Newcastle, 2005.

[4] T. Rofer, J. Brose, E. Carls, J. Carstens, D. Gohring, M. Juengel, T. Laue, T. Oberlies, S. Oesau, M. Risler, M. Spranger, C. Werner, and J. Zimmer. The german national robocup team. Technical report, Germany, 2006.

[5] T. Röfer, H. Burkhard, O. von Stryk, U. Schwiegelshohn, T. Laue, M. Weber, M. Juengel, D. Gohring, J. Hoffmann, B. Altmeyer, T. Krause, M. Spranger, R. Brunn, M. Dassler, M. Kunz, T. Oberlies, M. Risler, M. Hebbel, W. Nistico, S. Czarnetzki, T. Kerkhof, M. Meyer, C. Rohde, B. Schmitz, M. Wachter, T. Wegner, and C. Zarges. German team: Robocup 2005. Technical report, Germany, 2005.

[6] M. Roth, D. Vail, and M. Veloso. A world model for multi-robot teams with communication. In *IROS-2003*, 2003.

[7] M. Veloso, P. Rybeski, S. Chernova, C. McMillen, F. v. J. Fasola, D. Vail, A. Trevor, S. Hauert, and R. Espinoza. Cmdash'05: Team report. Technical report, Carnegie Mellon University, 2005.