Using the Opponent Pass Modeling Method to Improve Defending Ability of a (Robo)Soccer Simulation Team

Jafar Habibi, Hamid Younesy, and Abbas Heydarnoori

habibi@sharif.edu, hyounesy@cs.sfu.ca, aheydarnoori@cs.uwaterloo.ca

Abstract. Modeling agents' behavior has always been a challenge in multiagent systems. In a competitive environment, predicting future behaviors of opponents helps to make plans to confront their actions properly. We have used the RoboCup soccer server environment [1] to design a coach, capable of analyzing simulated soccer games and making decisions to improve teammate players' behavior during the games. We will introduce our "Opponent Pass Modeling" method which makes a model of opponent team's passing behavior to guard opponent players and as a result, to improve the defending behavior of our team. We will also describe a new approach to evaluate coach algorithms using soccer server log-files and LogCoach tool.

1 Introduction

The Simulation League of RoboCup provides a standard competition platform where teams of software agents play against each other in a simulated soccer game [2]. In robotic soccer tournaments, a team of agents plays against another team of agents for a single, short (typically 10-minute) period. The opponents' behaviors are usually not observable prior to this game and so they must be analyzed and modeled during the game. A common challenge for agents in multiagent systems is trying to predict what other agents are going to do in the future. Such knowledge can help an agent determine which of its current action options are most likely to help it achieve its goals.

So far, there has been some good works done on opponent behavior modeling in robotic soccer environment. Ideal-model-based behavior outcome prediction (IMBBOP)[3] is a technique which predicts an agent's future actions in relation to the optimal behavior in its given situation. An online opponent model recognition method is introduced in [4, 5], in which the coach uses a number of pre-defined opponent models to provide the agents with proper team plans. More recently, some methods focus on unsupervised autonomous learning of the sequential behaviors of agents, from observations of their behavior [6].

In this paper, we will introduce the "Opponent Pass Modeling" method which uses statistical information about opponent players' successful passes to mark them during a game. We will also describe a new approach to test this method using soccer server log-files, by the means of some useful software tools designed

D. Polani et al. (Eds.): RoboCup 2003, LNAI 3020, pp. 543–550, 2004.

in *Sharif-Arvand* soccer simulation team [7] including LogCoach. As a platform for our coach algorithms, we have designed and implemented a coach, which is capable of gathering a rich collection of statistical information during soccer simulation games. This statistical information then can be used for doing analysis in a game to make decisions to improve players behavior [7, 8].

This article is organized as follows. Section 2 will describe the opponent pass modeling method from coach-side view which is more about gathering and analyzing statistical information and from players-side view which focuses on utilizing the results provided by coach. Section 3 introduces our testing approach using soccer server log-files with the tools implemented in Sharif-Arvand team. In section 4 we suggest some minor changes to adapt this method with the standard coach language. Section 5 presents conclusions and directions for future work.

2 Modeling Opponents' Passing Behavior

In a typical soccer simulation game, single player attacks will rarely result in scoring a goal; It is simply because of the fact that opponent team already benefits from players defending the area. So, players use passing techniques during an attack to keep the ball from opponents and make the situation ready for scoring. Figure 1 shows two sample effective methods applied by good teams during previous RoboCup competitions. The models have been recognized by observing different log-files of the games played by these teams. In each case, attacking players have used passes to open teammates to change the region of attention (where the defenders have crowded) making them free to position better for next passes. Numbers on the arrows indicate steps of scenario, and for simplicity, only the important movements have been shown in each step. It can be seen that if the defenders had marked the attackers properly they could have avoided surrendering a score. Marking, generally means guarding a player to prevent it from advancing the ball towards the goal, making an easy pass or getting the ball from a teammate.

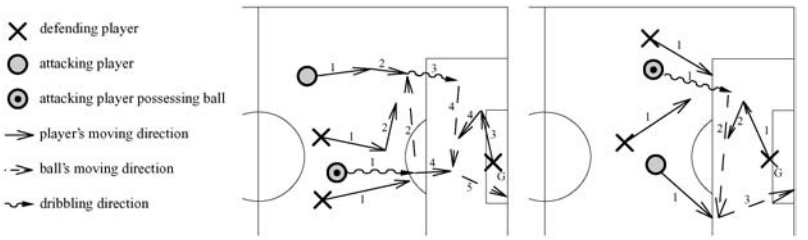

Fig. 1. Two samples of successful group attacks

To determine which players should be marked, we take advantage of the fact that teams are likely to show similar behaviors in similar cases. So, the players may use a similar sequence of actions when they are in similar situations. During

a game, coach considers the successful passes of opponent players to model their passing behavior. A successful pass can be defined as a ball transfer between two teammate players. Players will use the results later, to mark opponents during a game.

2.1 Coach-Side Analysis

A coach client has the advantage of receiving noise free data about the field from soccer server. So it is the best agent to centrally collect and analyze information in a game. In a typical layered design for coach, the lower layers are responsible for communicating with the soccer server, storing the received data in proper data structures and recognizing the skills done by players. For example, skills like possessing the ball or shooting the ball can be detected by comparing the position and the speed of the ball and players during a sequence of cycles; or for detecting passes which are a more complex skill, we use the lower detected skills like the current possessor of the ball and the last player who kicked the ball; if they both belong to the same team the ball has been passed, otherwise it has been intercepted. Then, based on these primary information in the lower layers, more intelligent algorithms for the coach have been designed.

In our method, the coach uses information about successful passes of opponent players to build a simple model for their passing behavior. As it can be seen in figure 2(a), the coach stores the passes that opponents have done in the defense area as well as initial coordinates of the receivers of the passes. The defense area is defined as a portion of the field near our own goal and is about one third of the field length. This information is then sent to the players in proper cycles. Here is a simple pseudo code showing this part of the method:

```
for each successful pass of opponent {
  if source [or destination] of the pass is inside defense area {
    T1 = PassStartTime
    T2 = PassEndTime
    P = PassReceiverPlayer
    SuccessfulPassList.Store(BallPos(T1), BallPos(T2), PlayerPos(P, T1))
  }
}
```

2.2 Players-Side Utilization

Now it is players' turn to use the information recently received from coach. The basic idea is to mark the opponent who is the most probable to receive a pass from the player possessing the ball. So, when an opponent player possesses the ball in our defense area we check whether the current position of that player is near to one of the BallPos_1s in the list received from coach; i.e. the starting point of the passes. If so, the player may send a similar pass and we should mark the player who will probably receive the pass. As shown in figure 2(b) we choose the nearest player to the PlayerPos corresponding to that pass; i.e. the nearest

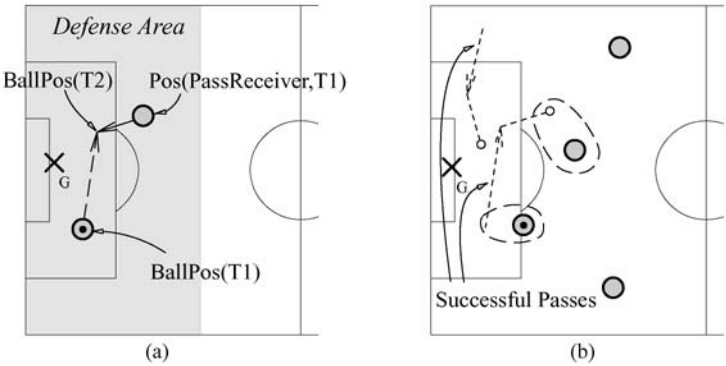

Fig. 2. (a) Coach collects information about successful passes (b) Players decide about the opponent to be marked

plater to the guessed pass destination. Now, one of our players (e.g. the nearest player) should mark the selected opponent player. Pseudo code below shows the described algorithm in more details:

```
if BallOwner is an opponent and BallPos is inside defense area {
  POpp = BallOwner
  for each item in the successful-pass list (i)
    if IsNear(Pos(POpp),BallPos_1[i])) {
      for all opponent players (P[j])
        if IsNearest(Pos(P[j]), PlayerPos[i])
          MarkOpponentPlayer(j)
    }
}
```

Lots of optimizations can be done to improve the collected information and the decision of the players when implementing this algorithm. In case more than one BallPos_1 is nominated, the opponent player which has less defense players near will be chosen. Also, in addition to original coordinates of successful passes, we store the symmetric coordinates related to the fields' x-axis; because of the fact that, teams usually have symmetric formations which causes the players to play symmetrically. We can also compute values for passes according to their regional importance or consequential effect in the game (e.g. surrendering a goal). In players-side, we only consider opponents who can receive the predicted pass in a point near the previous successful pass (near BallPos(T2)). Note that this method does not rely on the individual players (their uniform number). So, players' substitutions will not affect the result.

3 Using LogCoach for Evaluation

Using results of games is not necessarily a good way to evaluate most coach algorithms. It is because of the fact that decisions made by a coach, are usually

high level commands which need to be executed by players. So, even if coach makes decisions properly, they won't be effective unless players can utilize them during a game, which will require well programmed players in skills, tactics, etc. For example, considering our pass-behavior modeling algorithm, even if players correctly predict opponents who will receive passes, they need a good marking skill to mark opponents. If for any reason, a player fails to mark an opponent properly, not only it will not be able to stop the attack, but also one of the defenders uselessly will be hanging around.

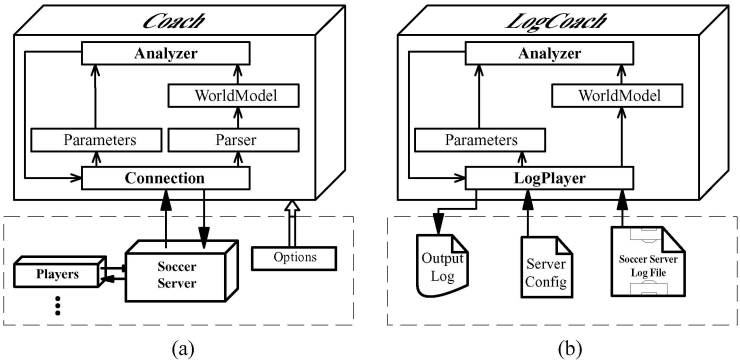

Fig. 3. (a) Layered structured of Sharif-Arvand coach (b) LogCoach structure

So, we have designed and implemented LogCoach tool. Figure 3 presents the structure of Sharif-Arvand online coach client and LogCoach. In the online coach, connection layer is responsible for gathering data by communicating with soccer server. While, in the LogCoach, LogPlayer module uses soccer server output log-files to provide the the game data and stores the output as graphical information which can be viewed using a visualizer tool called CoachDebugger later. So, rather than running real soccer simulation games, we can use stored log-files as the input data, without any changes required in other layers.

Using the LogCoach instead of coaching actual games, offers some benefits and some limitations. Instead of running a complete soccer simulation game a log-file can be processed in a few seconds; so it increases speed of development and testing. Also, combined with the CoachDebugger it is a great help for testing algorithms while enabling us to analyze the coach decisions by observation. An other main advantage of using logfiles is that exactly the same input data is available between the changes of the algorithms allowing it to see if new code segments really improve the algorithm. However, without a real game, it is impossible to have active behavior like interacting with the players or adapting the parameters according to the result of the previous decisions in a game.

To test and evaluate the correctness and effectiveness of our opponent pass modeling algorithm, we developed a simple testing module using the LogCoach. The idea is basically to find out how many opponent passes could be avoided

during a game, and more precisely, how many goals could have been avoided if players had used the opponent pass modeling algorithm to mark opponent players. Actually, we put our players' predicting algorithm in the test module and use the coach's statistical information to predict players who will receive passes:

```
When an opponent possesses the ball in our defense area {
  Predict the player who will receive the pass
  Store the predicted player and other game status information
  GuessesCount++
}

For each successful pass of an opponent in our defense area {
  if the pass receiver is the predicted one
    CorrectGuesses++
}
```

In the end the $\frac{CorrectGuesses}{GuessesCount} \times 100$ will be the percentage of correct guesses during the game. To test the algorithm we used over 40 log files most of which were from the last two RoboCup official games. We chose the games whose total scored goals were more than 10 and we considered the defeated team as our team for which the coach was used. The result is presented in Table 1. As you can see the results are satisfactory; even in the worst case, about half of the opponents' passes in our defense area could be predicted and probably avoided. Although it does not mean that avoiding the passes could completely stop the attacks, but they sure could deviate the opponents' main plan.

Table 1. Evaluation results based on correct pass predictions

	Game Result	GuessCount	CorrectGuess	Correctness
Worst Case	1-11	23	10	43%
Best Case	0-23	34	31	91%
Average (rounded)	0-14	26	19	74%

Doing some changes in the evaluation function, the results were even more interesting. This time we just considered the passes which consequently resulted in a goal.

```
When opponent scores a goal {
  P1 = Player who scored the goal
  P2 = The last player who sent a pass to P1
  GoalGuessesCount ++
  if we had already predicted P1 to receive a pass from P2
    GoalCorrectGuesses++
}
```

Table 2 shows results of the new evaluation approach. Correctness column shows the percentage of the goals which could be avoided if the scorer was marked before receiving the last pass. Again we insist that this does not mean that the defending team could survive from all the successfully predicted attacks, specially because of their weakness in skills and tactics comparing with the opponents players.

Table 2. Evaluation results based on prediction of passes resulted in a goal

	Game Result	GoalGuessesCount	GoalCorrectGuesses	Correctness
Worst Case	1-11	11	5	45%
Best Case	0-23	23	19	83%
Average	0-14	14	9	67%

4 Adapting for the Standard Coach Language

Standard Coach Language is a general language for communication between a coach and players during a soccer simulation game. The idea is that a coach uses a unified standard language to communicate with teammate players, so it can be used for coaching any team that can understand this language.

The syntax of the standard coach language allows the coach to define rules, regions, constraints and instructions [1]. Our opponent pass modeling method can be easily adapted to be used with a standard language based coach. For each successful pass, it defines regions around the starting point of the pass and starting point of the pass receiver. Then it defines a rule for the teammate players to mark the opponent inside a region, related to where the ball exists. Here is an example showing usage of the standard coach language for this method (based on soccer server version 8.0).

```
(definer "Reg1" (arc (pt X_FROM Y_FROM) 0 R 0 360))
(definer "Reg2" (arc (pt X_PASS_REC Y_PASS_REC) 0 R 0 360))

(definerule "rule1" model
((and (time >= INITIAL_TIME) (bpos "Reg1") (bowner opp {X}))
(do our {X} (markl "reg2"))) )
```

where X_FROM & Y_FROM will be coordinates of starting point of the pass, X_PASS_REC & Y_PASS_REC will be initial position of the pass receiver. R and INITIAL_TIME are constant values (e.g. 5 and 1000) indicating the radius of the regions and the initial time to be waited before executing the rule (a minimum time, coach needs to collect more data). "markl" is an instruction defined in the standard language, tells the players to mark the line from the ball to a specific region.

5 Conclusions and Further Works

In this paper, we described a new method to model the passing behavior of opponent teams, based on statistical information about previous successful passes.

We saw that the model then can be used by players to guard opponents properly and as a result to improve their defense ability. We presented a new approach to evaluate coach algorithms using soccer server log-files instead of running actual games. We also showed how the opponent pass modeling method can be adapted to be used with standard coach language.

We are currently working on using this method to decide about team formation. For example if coach observes situations in which there are more opponent players to be marked than the number of defenders it may decide to improve the defense line by changing number or placement of defenders in the formation.

References

1. Chen, M., et al.: RoboCup Soccer Server User Manual for Soccer Server Version 7.07 and later. *http://sserver.sourceforge.net/docs/*, last visited: Jan 2003
2. Kitano, H., et al.: The RoboCup Synthetic Agent Challenge. In Proceedings of the 15th International Joint Conference on Artificial Intelligence (1997)
3. Stone, P., Riley, P., Veloso, M.: Defining and Using Ideal Teammate and Opponent Models. In Proceedings of the Twelfth Innovative Applications of Artificial Intelligence Conference (2000) 1040-1045
4. Riley, P., Veloso, M.: Coaching a Simulated Soccer Team by Opponent Model Recognition. In Proceedings of the Fifth International Conference on Autonomous Agents (2001) 155-156
5. Riley, P., Veloso, M.: Recognizing Probabilistic Opponent Movement Models. RoboCup 2001: Robot Soccer World Cup V, Springer-Verlag New York (2002) 453-458
6. Kaminka, K., et al.: Learning The Sequential Coordinated Behavior of Teams from Observations. In Proceedings of the RoboCup-2002 Symposium (2002)
7. Habibi, J., et al.: Sharif-Arvand Simulation Team. RoboCup 2000: Robot Soccer WorldCup IV, Springer-Verlag New York (2001) 433-436
8. Habibi, J., et al.: Coaching a Soccer Simulation Team in RoboCup Environment. In The 1st EurAsian Conference on Advances in Information and Communication Technology (EurAsia-ICT 2002), Springer-Verlag (2002) 117-226
9. Drucker, C., et al: As Time Goes by: Using Time Series Based Decision Tree Induction to Analyze The Behavior of Opponent Players. RoboCup 2001: Robot Soccer World Cup V, Springer-Verlag New York (2002) 325-330