WrightEagle 2D Soccer Simulation Team Description 2012

Aijun Bai, Haochong Zhang, Guanghui Lu, Miao Jiang and Xiaoping Chen

Department of Computer Science,
University of Science and Technology of China,
baj@mail.ustc.edu.cn, xpchen@ustc.edu.cn

Abstract. WrightEagle 2D soccer simulation team is a 2D soccer simulation team which has been participating in annual RoboCup competitions since 1999 and won 3 champions and 4 runners-up in the past 7 years. In this paper, we briefly present our current research efforts and some newly introduced techniques since the last competition.

1 Introduction

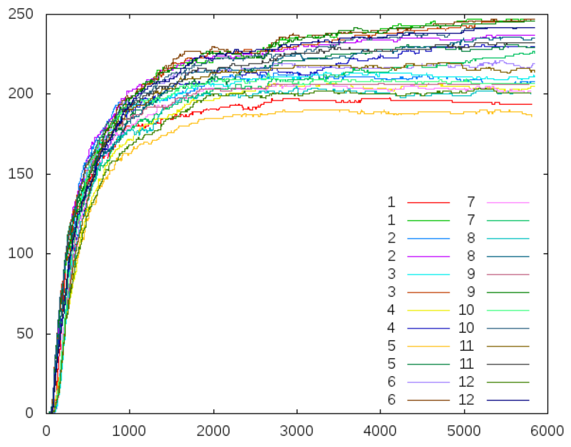
WrightEagle 2D soccer simulation team, which was established in 1998 as the first branch of WrightEagle RoboCup team developed by Multi-agent Systems Lab. of USTC, has been participating in annual competitions of RoboCup since 1999. Recent years, we have won the champion of RoboCup 2011, 2009 and 2006, the runner-up of RoboCup 2010, 2008, 2007 and 2005.

We take RoboCup soccer simulation 2D as a typical problem of multi-agent systems, and our long-term goal is to do research in decision-making and other challenging projects in artificial intelligence [3]. This year, we developed some new techniques for both the low-level skills and the high level decision-making model in our new team WE2011, based on our research efforts [5, 9, 10, 8, 7, 13, 12, 2, 11]. In this paper, we present a brief description of some of our progress mentioned above.

In 2011, we also released the newest version (3.0.0) of our team's base code WrightEagleBASE to the public as an open-source software which can be freely accessed from our team's website.[1] We hope that our released software will be helpful to a new team who wants to participate in the RoboCup event and/or start a research of multi-agent systems.

The reminder of this paper is organised as follows. Section 2 introduces the Opponent Formation Detection System of our team, and empirically evaluates its performance. Section 3 presents the Monte Carlo Localization techniques used in our team with some empirical results. Section 4 briefly describes the newly developed Multi-step Positioning and Passing Behaviours. Finally, the paper is concluded in Section 5.

Fig. 1. Opponent formation detection process



2 Opponent Formation Detection System

Detection of the opponent's formation type takes an important role in opponent modeling, affecting our team's global strategy substantially. An opponent formation detection system (OFDS) was implemented to tackle this issue.

In OFDS, we use Delaunay Triangulation [1] to model the opponent's formation, and categorize the formation space into several predefined types, such as 433, 442, 4231, etc. Each type of the formation is initialized by some default samples based on statistic data taken from some typical logfiles. As the game progresses, all types of the formation are constantly inserted more samples based on the current state of the field. The update procedure also computes the difference for each sample inserted. Then the type of formation, which has the minimal cumulative differences currently, will be selected as the opponent's formation. It is worth pointing out that the opponent's formation types in attack and defense situation are treated separately, and updated in different time in our team.

To evaluate OFDS, we ran our team against Helios10, which uses formation 4231 for both attack and defense situation by default, for 250 games independently. The empirical results for each player (including our online coach denoted by player 12) are shown in Figure 1, where x axis represents the cycle number during a game, and y axis represents the number of games which OFDS correctly detected the opponent formation (i.e. 4231). Notice that the detection processes of both formation types in attack and defense situation for each player are depicted in Figure 1 without distinguish. Different players may have different probabilities of correct detection depending on the precision of their internal state information of the opponents. For example, the online coach always has the most precise world state, but the goalie who stands closely to the goal most

[1] `http://www.wrighteagle.org/2d/`

Table 1. Average error of the player's own expected state

x(m)	y(m)	v_x(m/s)	v_y(m/s)	d_b(Deg)	d_n(Deg)
0.047	0.046	0.0014	0.0013	0.44	1.5e-6

of the time can only observe precisely the nearby area. It can be seen from Figure 1 that OFDS converged after about 2000 cycles from the beginning of a match, and resulted at about 0.8 the average probability of correct detection summarized in all players (including online coach).

3 Monte Carlo Localization

In RoboCup 2D domain, the player must overcome the difficulty that it can only receive local and noisy observations, to obtain a precise enough estimation of the environment's current state. In our team, the player estimates the current state from its *belief* [6]. A belief b is a probability distribution over state space, with $b(s)$ denoting the probability that the environment is actually in state s. We assume conditional independence between individual objects, then the belief $b(\boldsymbol{s})$ can be expressed as

$$b(\boldsymbol{s}) = \prod_{0 \leq i \leq 22} b_i(\boldsymbol{s}[i]), \tag{1}$$

where $\boldsymbol{s}$ is the full state vector, $\boldsymbol{s}[i]$ is the partial state vector for object i, and $b_i(\boldsymbol{s}[i])$ is the marginal distribution for $\boldsymbol{s}[i]$. A set of m_i weighted samples (also known as particles) are then used to approximate b_i as:

$$b_i(\boldsymbol{s}[i]) \approx \{\boldsymbol{x}_{ij}, w_{ij}\}_{j=1\ldots m_i}, \tag{2}$$

where $\boldsymbol{x}_{ij}$ is a sampled state for object i, and w_{ij} represents the approximated probability that object i is in state $\boldsymbol{x}_{ij}$ (obviously $\sum_{1 \leq j \leq m_i} w_{ij} = 1$).

In the beginning of each cycle, these samples are updated by Monte Carlo procedures using the domain's *motion model* and *sensor model* [4]. In our team, the player assumes that all other players share a same predefined behavior model: they will execute a random kick if the ball is kickable for them, or a random walk otherwise. Finally, the environment's current state $\boldsymbol{s}$ is estimated as:

$$\boldsymbol{s}[i] = \sum_{1 \leq j \leq m_i} w_{ij} \boldsymbol{x}_{ij}. \tag{3}$$

Based on empirical results taken from actual competitions, the estimated state is sufficient for the player to make good decisions, particularly for the state of the player itself and other close objects.

Table 1 shows the average error between the player's self-maintained own expected state information and the server's actual state information during one random selected competition, where x, y, v_x, v_y, d_b and d_n represent the player's x-position, y-position, x-speed, y-speed, direction of the body and the direction

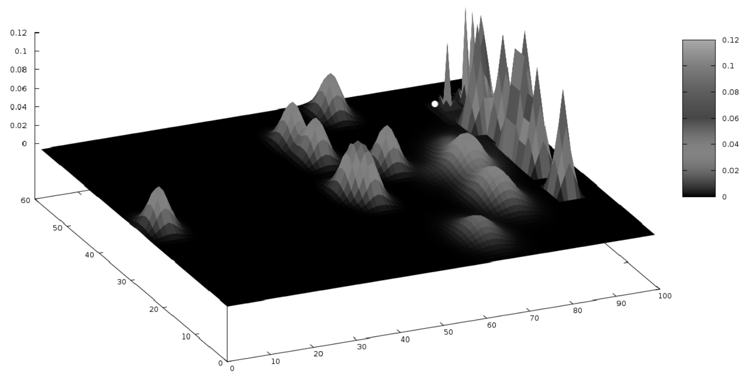


Fig. 2. Belief on other players' position information

of the neck respectively. It is worth mentioning that the error of neck_dir is almost zero, because the turn_neck action in the 2D domain is executed without any noise.

Figure 2 depicts an example of the player's belief on other players' position information (i.e. probability distribution of (x, y) for each player) during that match when the player, denoted by the small white circle, is holding the ball and facing the opponents' goal (the player's team is on the left side) for awhile. The particular (x, y) probability distributions of the player itself and the ball are not showed in Figure 2, because they are relatively too narrow and high to be drawn appropriately in the same figure.

4 Multi-step Positioning and Passing Behaviors

Positioning behavior focuses on the problem that where the players position themselves when they do not have the ball in the attack situation. The performance of the positioning behavior benefits substantially if it can think more than one "step" in the future. For example, if the player 11 successfully speculates that the ball holding by teammate 10 will be passed to teammate 4 in the near future, then it can get ready in advance to cooperate with teammate 4 to make a potential attack by positioning itself to some appropriate places.

A ball running track is defined for this purpose as the sequential ball holders during a successful attack process denoted by a list of state nodes. In the first node of a ball running track, one of our teammates is holding or intercepting the ball. In the last node, the ball will be shot directly to the opponent's goal. Note that all the nodes except the first one represent some virtual states in the future. We assumed that each player in the ball running track has only two available macro-actions: pass and dribble.

A forward search tree is constructed to find the most potential attack process based on the successful probability of a ball running track using some heuristic search techniques. Once the search procedure is finished, the player will find out the most potential teammate who will finally pass the ball to it, and the steps that should be considered in its own positioning behavior. Here a step represents a successful pass between teammates.

A respective multi-step passing behavior is also implemented. It is an option (i.e. macro-action) based forward search algorithm, briefly described as follows.

1. Generate all possible pass behaviors using previous greedy search algorithm;
2. Sample some behaviors and construct them as options;
3. For each option, use a simulator to predict belief state after executing it;
4. Repeat step 1, 2 and 3 to certain horizon H;
5. Select the maximum rewarded option chain.

Currently, the multi-step positioning and passing behaviors can only work in a very limited way due to the unpredictable property of the environment, especially when the states are far in the future. Our future work intends to improve the performance by considering more abstractly.

5 Conclusions

This paper introduced our RoboCup soccer simulation 2D team, WrightEagle, and described our current research efforts and some newly introduced techniques from last competition, including: 1) Opponent Formation Detection System, 2) Monte Carlo Localization, and 3) Multi-step Positioning and Passing Behaviors. It can be seen from the empirical results that our team's final perfermance has been improved based on these efforts.

References

1. Akiyama, H., Noda, I.: Multi-agent positioning mechanism in the dynamic environment. RoboCup 2007: Robot Soccer World Cup XI pp. 377–384 (2008)
2. Bai, A., Wu, F., Chen, X.: Online planning for large mdps with maxq decomposition (extended abstract). In: Proc. of 11th Int. Conf. on Autonomous Agents and Multiagent Systems. Valencia, Spain (June 2012)
3. Chen, X.: Challenges in research on autonomous robots. Communications of China Computer Federation 3(12) (2007)
4. Dellaert, F., Fox, D., Burgard, W., Thrun, S.: Monte carlo localization for mobile robots. In: IEEE International Conference on Robotics and Automation. vol. 2, pp. 1322–1328. IEEE (2001)
5. Fan, C., Chen, X.: Bounded incremental real-time dynamic programming. In: Frontiers in the Convergence of Bioscience and Information Technologies, 2007. FBIT 2007. pp. 637–644. IEEE (2007)
6. Kaelbling, L.P., Littman, M.L., Cassandra, A.R.: Planning and acting in partially observable stochastic domains. Artificial Intelligence 101(1-2), 99–134 (1998)
7. Shi, K., Chen, X.: Action-driven markov decision process and the application in robocup. Journal of Chinese Computer Systems (2009)
8. Wu, F., Chen, X.: Solving large-scale and sparse-reward dec-pomdps with correlation-mdps. RoboCup 2007: Robot Soccer World Cup XI pp. 208–219 (2008)

9. Wu, F., Zilberstein, S., Chen, X.: Multi-agent online planning with communication. In: Proc. of the 19th Int. Conf. on Automated Planning and Scheduling. pp. 321–328 (2009)
10. Wu, F., Zilberstein, S., Chen, X.: Point-based policy generation for decentralized pomdps. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1. pp. 1307–1314. International Foundation for Autonomous Agents and Multiagent Systems (2010)
11. Wu, F., Zilberstein, S., Chen, X.: Online planning for ad hoc autonomous agent teams. In: Twenty-Second International Joint Conference on Artificial Intelligence (2011)
12. Wu, F., Zilberstein, S., Chen, X.: Online planning for multi-agent systems with bounded communication. Artificial Intelligence 175(2), 487–511 (2011)
13. Zhang, Z., Chen, X.: Accelerating point-based pomdp algorithms via greedy strategies. Simulation, Modeling, and Programming for Autonomous Robots pp. 545–556 (2010)