WrightEagle2009 2D Soccer Simulation Team Description Paper

Ke Shi, Aijun Bai, Yunfang Tai, Xiaoping Chen

Multi-Agent Systems Lab.,
Department of Computer Science and Technology,
University of Science and Technology of China,
Hefei, Anhui Province, China
{shike15, baj, tyf}@mail.ustc.edu.cn, xpchen@ustc.edu.cn
http://wrighteagle.org/2D/

Abstract. In WrightEagle2009, we continue to research based on the previous WrightEagle 2D soccer simulation team. WrightEagle has won the runner-up of RoboCup 2008 and Robocup 2007, the Champion of RoboCup 2006, and the runner-up of RoboCup 2005. In this paper, we mainly present the team structure of our new team WE2009, and the new techniques since the last competitions.

1 Introduction

The WrightEagle RoboCup Team was established in 1998, starting with only the 2D soccer simulation team. In the following years, the 4-legged team, the 3D soccer simulation team and the MSRS team have joined in WrightEagle. We have participated in annual competition of RoboCup from 1999. Last year, we have got three runner-ups(2D, 3D and 4-legged) and one third place(MSRS) in RoboCup 2008, Suzhou, China.

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-theoretic planning and other challenging projects in artificial intelligence.[1] This year, we have developed a new team structure based on the theory of partially observable stochastic game. Besides, we have developed some new techniques for the low level and high level in our new team WE2009, based on our basic researches on decision theoretic planning.[2][3] In this paper, we present a brief description of some of progress mentioned above.

2 Team Structure

We have developed a new team structure in WE2009. This structure is based on the theory of partially observable stochastic game. POSG can be used to model the cooperation and counterwork problem in large scale multi-agent system such as RoboCup 2D soccer simulation. [4] We take the whole decision process as decentralized partially observable markov decision process, which is given as a tuple $\langle \tau, K, S, A, P, \{R_i\}, \{B_k\} \rangle$, where

- τ is a finite set of agents' index, for example, $\tau = \{1, \ldots, 22\}$ is the set of all players in the field

– **K** is a finite set of behavior generators' index
– **S** is a finite set of factorial state
– **A** is a finite set of all atomic action of the soccer server
– **P** is the state transition function, given a state and an action, the output is the possible successor states and their probabilities
– R_i is the reward function of the i_{th} agent
– B_k is the k_{th} behavior generator, given the state and the appointed agent, the output is the set of effective behavior and each behavior's successor states

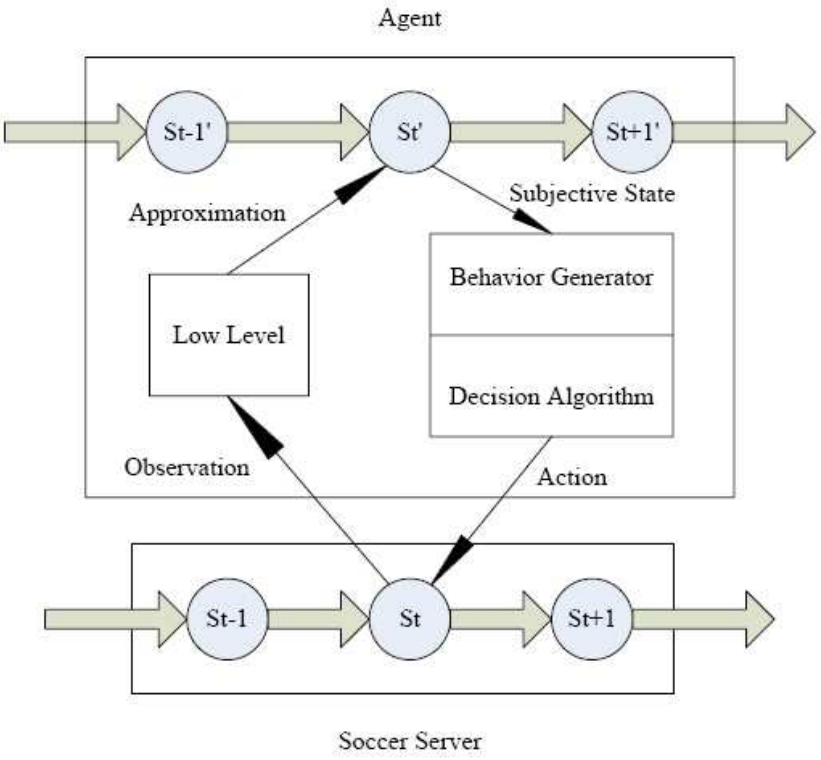

Fig. 1. Team Structure of WE2009

Based on this model above, the team structure is shown as Figure 1. Relative to the objective markov process, the agent holds up a subjective state transition process from the observation. After the high level module makes decision at each cycle, the agent takes action and goes to the next cycle. As pointed in our TDP last year, the anytime structure is used in the decision module.

3 Newly introduced techniques

Based on the new team structure above, we have developed some new techniques in our new team. We use the MDP model or POMDP model to describe some issues in 2D soccer simulation. Besides, we use some methods to make our team play better in the latest soccer server. We have tested the new techniques in the latest soccer server, and they bring more advantages to our new team.

3.1 Dash Decision Model

As a new multi-direction dash model which is more natural is introduced to soccer server 13.0 and later version, we use a MDP based method to solve this problem. In our MDP model, state stands for agent's position and velocity, action is taking as dash and turn, immediate reward is -1 in common states and 0 in goal states. Under the consideration of simplifying the problem, we do not take account of the uncertainty of observation. Because the state space is continuous, a usual table based value function can not be used here, we use an artificial neural network instead. After using this method, our agents dash more naturally and efficiently.

3.2 Kick Decision Model

As in real soccer games, because of the initial ball speed and the ball position relative to the player, the player can not make the ball get a high speed at the expected direction after only one cycle kick. Therefore, there should be a multi cycle kick model which we call kick decision model. This model is the base skill of the agent, and its goal is to try to take few cycle to make the ball get the appointed speed at the expect direction. This skill is always used in the shoot behavior and pass behavior, so it is very important for a 2D soccer simulation team. we use the reinforcement learning method in our previous team, and we take this problem as a MDP problem in our new team.[5] We take the multi cycle kick process as two markov processes. The 1st step takes only one cycle. This step is a search process and all reachable states are calculated in this step. In the 2nd step, we use the value iteration algorithm to calculate the policy. The update uses the Bellman equation:

$$V^{n+1}(s) = \max_{a \in A}\{R(s) + \gamma \sum_{s' \in S} T(s, a, s')V^{n}(s')\}$$

In the iteration process, we set the max iteration step to 3, because 3 cycle is enough for a player to kick the ball to the max ball speed in the current soccer server. Relative to the reinforcement learning method, the MDP method is much better in time efficient and kick successful probability. The combination of search method and value iteration method can also provide a new solution for the similar problems.

3.3 Visual Decision Model

In our old visual decision system, each object is represented as a circle known as a maximal movable area. Now we find that this is not enough, as maximal movable area will not always be a circle if we consider the update routine of sight information. For example, if the agent took a look at part of the maximal movable area of a teammate which has not been seen for several cycles, but after the update routine of sight information, the agent found that the teammate has not been seen again. It means that the seen part should be removed from the agent's maximal movable area, thus the maximal movable area will not be a circle anymore.
In common situations, the maximal movable area can be as complicated as possible, and the old way of visual decision will not perform well. So we introduce the POMDP model here, in which the belief state is obviously taken as objects' position distribution. Also under the consideration of simplifying the problem, we just use some position samples (grid and it's appearance possibility) to represent the position distribution which original idea is taken from Monte Carlo Localization.

3.4 How to Save Stamina

According to the new release of soccer server, players have stamina capacity. It means that player could run out stamina and stop if he does not save stamina among the match. So we think it is necessary to find a new way to help players saving stamina. Since one player does not know another player's stamina, saving stamina is hard to proceed. To solve this problem, we think that coach is a good player to compute each player's stamina. Because the coach can get information in field without noise, also the coach can use freeform to tell each player information in special play mode.
Therefore in our new team, the coach computes the cost of stamina of each player according to their movements at each cycle, then update their stamina and stamina capacity. In special play mode, coach use freeform to tell each player who has the minimum stamina capacity, then players can use some way to saving its stamina and help other players. For example, when in kick_in mode, player who will kick the ball can wait for the one who has the minimum stamina to recover it. Also he can pass the ball to the one who has more stamina capacity instead of others.

4 Conclusion

As is shown above, we spent a lot of time in improving the low level implementation details and high level decision making models. We evaluate our improvements by a lot of experiments to observe if it is efficient and useful and the result is good. All of features above enhance the strength our team. We will try to make more progress along the line described in this paper.

References

1. Xiaoping Chen, et al, Challenges in Research on Autonomous Robots, Communications of CCF, Vol. 3, No. 12, Dec 2007.

2. Changjie Fan and Xiaoping Chen. Bounded Incremental Real-Time Dynamic Programming. IEEE Proceedings of FBIT 2007, Jeju Island, Korea, 2007.
3. Feng Wu and Xiaoping Chen. Solving Large-Scale and Sparse-Reward DEC-POMDPs with Correlation-MDPs. Proceedings of RoboCup Symposium 2007. Atlanta, America, July 2007.
4. L. P. Kaelbling, M. L. Littman and A. R. Cassandra. Planing and Acting in Partially Observable Stochastic Domains. In Artificial Intelligence, pages 101-134, 1998.
5. Lihong Li and Michael L. Littman. Lazy Approximation for Solving Continuous Finite-horizon MDPs. In AAAI, pages 1175-1180, 2005.