Wright Eagle 2-D Simulation Team

Chen Xiaoping, Fan Changjie, Wu Feng, Wang Jiliang, Cai Jingnan

AI Center, Department of Computer Science,
University of Science and Technology of China, Hefei, China
xpchen@ustc.edu.cn, {cjfan, wufeng, jlw, south}@mail.ustc.edu.cn
WWW Home Page: http://ai.ustc.edu.cn

Abstract. In Wright Eagle 07, we continue to research based on the previous Wright Eagle simulation 2D team. Wright Eagle has won the champion in Robocup06-Bremen, and the Robocup China Open-Suzhou, China, as well as the first runner-up in Robocup05-Osaka. In this paper, we present the innovations of our team since the last simulation league competitions, and relate them with previous work developed by simulated RoboCup teams and our previous work.

1 Introduction

We present a brief description of the implementation principle of the Wright Eagle 2-D Simulation team. Our long-term goal is to build a robot soccer team where the decision making part is applicable to the general multi-agent system. As a result, we not only focus the high-level models for decision making, but also implementation details in the low level. In high-level decision making models, we improve the POMDP[3, 4] model. In low-level implementation details, we improved our dribble in many different features as well as the model of the stamina. Improvement of Low-level design of our new team is based on the implementation of last year, thanks to the information of the CMUnited-98[1] and FCPortugal2000[2].

1.1 A Novel Turn-dribble Search Method

The basic dribble-decision process is shown in the Figure 1.

By Turn-Dribble searching, we want to find the best turn-angle and dribble target in one decision process. In previous method, we first fix the angle which the player turns to, and then scan each angle value in the direction from far to near to the dribbler to find the best target. Now the problem is that there may be numerous angle values which have to be evaluated, thus it will take much time online. The novel method is an other searching strategy. We fix the target areas first, and then find the best target point in every target area, as illustrated in Fig. 2.

In every target area, we find the best target point by the follow methods (Fig. 3).

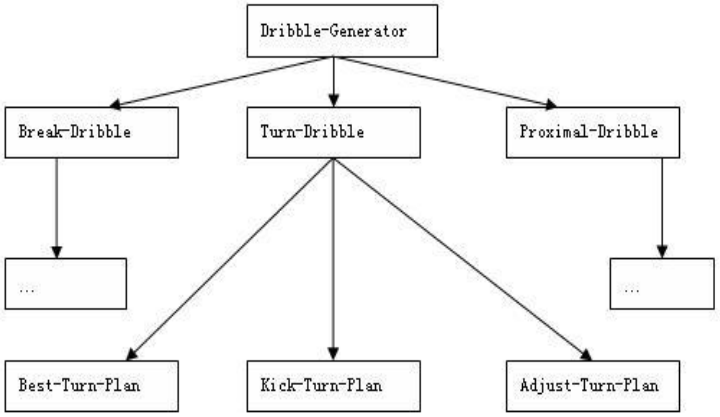

Fig. 1. Basic dribble-decision process

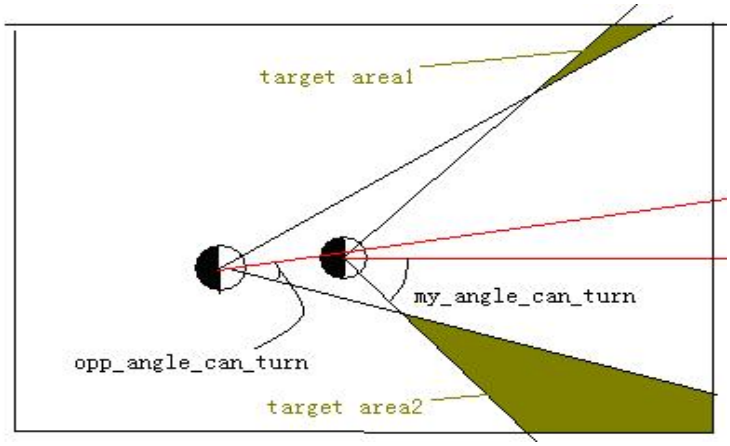

Fig. 2. First target area base on turn-angle

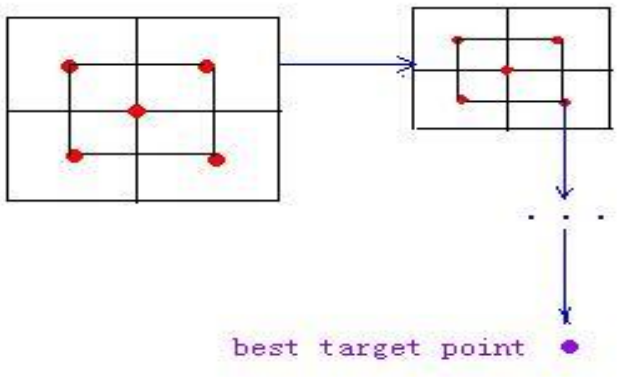

Fig. 3. Searching process of best target point

The agent evaluates the five points and find the best one among them. It continues the process in the next step until it finds the point good enough in this area. In this version of our team, these two methods on dribble-decision searching are both used.

1.2 Statistical offline analysis is used

The values of the rates for a good ball-pass and a good ball-dribble will be used in the Decision-making of our Robocup programs. In previous method, these values are specified based on human's experience, so naturally it may not coincide with the realtime situation. In the competitions, these values always vary online according to the style and aggressiveness of the opponent teams. In this version of our team, we make use of offline analysis. By offline analyzing large numbers of log files with the scene-recognize technology, we can get an interval of the rate values. Thus when online, the exact values will be specified dynamically from the intervals determined offline, depending on the situation of the competition. For example, in anti-tackle decision-making, we have specifed a value of the possibility if the opponent will slide tackle in one cycle, but whether the anti-tackle decision-making should be executed depends on the opponents' sliding tackle threshold value in our method. and the opponents' slide tackle threshold value in our team is a offline statistical value, because we have no time to get it online.

2 Improve the method of using the stamina

How to use the stamina rationally is a problem in the simulation-2d team. Making a good model of the opponent's stamina will greatly improve our agent

because the stamina of the opponent is critical to our agents' decision. In this version of our team, we model the problem of opponent's stamina as Fig. 4.

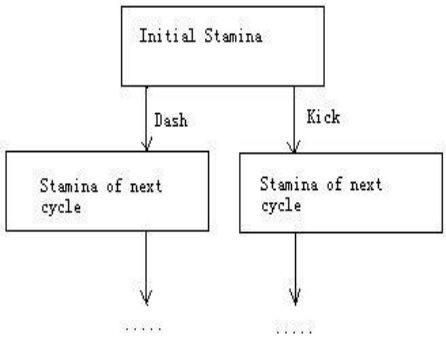

Fig. 4. Original stamina model

The problem is that we can not know the stamina of the opponent which is used to dash or kick accurately enough.

What we can know more correctly is the opponent's position, velocity and their variations in each cycle. We assume that the stamina change be only related with the position change and velocity change, rather than how one agent makes the changes. For example, we don't care whether an agent uses 40 power to dash or 100 power to dash. Therefore, we make some experiments to achieve this. In one experiment, the values such as distance one agent run, the initial velocity and the final velocity are fixed, and we use different methods to finish this running with these conditions fixed. At last, we find that the stamina this agent consumed is almost the same.

Then we can generator a model like this:

After we make this assumption, we use the simulation method to generate the formula indicating the relationship between stamina change, position change and velocity change, we describe the function as below:

$$\Delta(stamina) = f(\Delta(position), \Delta(velocity)) \tag{1}$$

With even more experiments, we minimize the error by introducing another factor to fix the error. As the initial stamina of each player is known, we can use the function above to obtain the stamina of every player at each cycle.

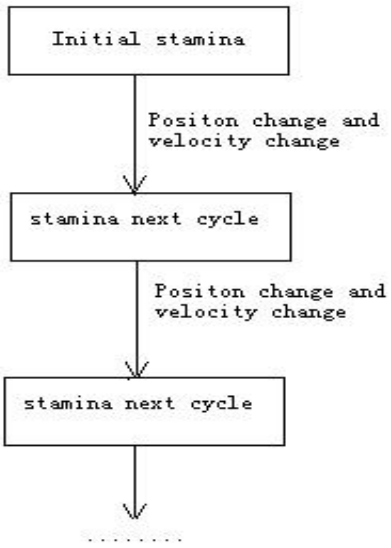

Fig. 5. Advantage stamina model

3 Decision Evaluation Mechanism

We regard the whole decision process as a POMDP model, in which the probabilities of observation are transformed to the probabilities of state-transition of the next stage. Thus the model is simplified to an MDP model[3, 4].

States S: In each state the ball is controlled of some player. When the ball is free, we directly predict the possible states of the ball under control, based on the interception model which is presently quite mature.
Action A: The atom actions prescribed by the server are not used. Instead a behavior module is introduced with domain knowledge. Examples of action that we defined are Passing, Dribble, Shoot Position and etc. Each action has its own formal model, to compute the possible state and the probability of state transition.
Reward R: The basic idea is to discriminate the several main scenarios, appointing different goals to each. The highest reward is achieved when the goal is reached; otherwise a small reward will be given according to state point sensitivity. A negative reward is given when failure.

A basic decision process is as follows. We analyze the situation, make sure the current state and confirm the goal state. The action generator can generate many actions as candidates. Each candidate has its reachable states and the transformation probability. The obtained values of the states is computed by an approximate method iteratively.

Especially, we make some improvements on this model when the ball is near the opponent's penalty area. Regarding the controller's change as a state change, we should consider more states, because as long as the shoot state is generated, it will possess a high evaluation. As a result, we can regard such state change as a ball pass from one to another near the penalty area and then shoot.

We can demonstrate it as Fig. 6.

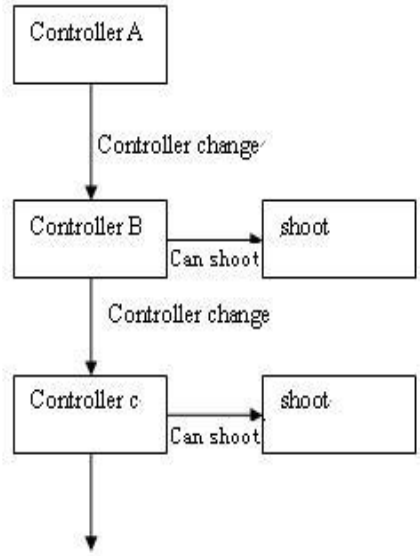

Fig. 6. Optimized model near the penalty

4 Conclusion

As is shown, we spent a lot of work in improving the low-level implementation details and high-level decision making models. In low-level implement, we add new features to dribble, while in high-level decision making model, we improve our models near the penalty. Besides, a new stamina model is proposed. We evaluate our improvements by a lot of experiments to see if it is efficient and useful and the result is good. All of features above enhance our team.

References

1. P. Stone, P. Riley, M. Veloso: CMUnited-98 champion simulator team. AI Magazine, 2000

2. Luis Paulo Reis, Nuno Lau: FC Portugal Team Description: RoboCup 2000 Simulation League Champion.
3. E J Sondik. The optimal control of partial observable Markov decision processes. Ph. D thesisStanford University1971.
4. H T Cheng. Algorithms for partially observable Markov decision processes[D]. Ph. D. thesisUniversity of British Columbia1988.