TIT HELIOS Team Description

Hidehisa Akiyama and Tobias Lindh

Department of Computational Intelligence and Systems Science
Tokyo Institute of Technology
Nagatsuta-cho, Midori-ku, Yokohama-shi, Kanagawa 226-8502, Japan
{akiyama, tobias}@ntt.dis.titech.ac.jp
WWW: http://www.ntt.dis.titech.ac.jp/home/akiyama/robocup/

Abstract. The team TIT HELIOS is a simulation soccer team for RoboCup Soccer Simulation League. This paper explains about this team's enhanced low level code and new high level approaches.

1 Introduction

The team TIT HELIOS inherits the code from the team ATeam that is our previous team, the participant of RoboCup 2000 and 2001. And our code takes in the some libraries from the other teams' source code, CMUnited99[2], YowAI[3] and so on. Especially, we thank for YowAI's position estimation module.

In last two years, we tried to develop the learning system for multi-agent cooperation using the Genetic Programming(GP)[1]. The result was very interesting, but it was not the sufficient level to use in the game.

From this year, we started new challenges. We divided the team development process into two levels, " Low-Level approach. Noiseless World Modeling and Optimization of Basic Skills" and "High-Level approach. Learning or Coaching of Cooperative Behavior".

2 Low-Level approach

2.1 World Modeling

The basic idea of our player's world model architecture is based on the CMUnited and YowAI. Furthermore, We extended these models and introduced the management of the relative position information. This technique enabled the player to recognize the object noiselessly when the object's distance from the player is near. When the player is controlling the ball, this has much effect.

2.2 Basic Skills

Our player's basic skills are totally hand coded, not acquired by learning. But we used some techniques, and we explain about them in this section.

Interception The ball interception skill is very important because the faster you reach the ball, the more advantage the team has. This skill was improved due to the improvement of the world model. We also introduced the backward interception for the quick ball interception.

Kicking We introduced some online search algorithms to improve the kicking skills. This idea is based on FCPortugal's kick optimization technique[4], but our method is more simplified. When the player can kick the ball, the player searches the appropriate kick queue. To be exact, the player searches the subtarget point within his kickable radius. For each kick type (hard kicking, rotation kicking, etc.), we used different search algorithms.

These search techniques enabled the player to control the ball very quickly, powerfully and correctly in any direction. And because these are online, the player can kick the ball in order to avoid the near opponent, and these have the robustness for the changes of server configuration and heterogeneous parameters. Furthermore, these search algorithms require very little computation time (less than 0.5msec in Pentium III 600MHz), therefore the sufficient computation resource is available for other processes.

Dribbling It became easier to implement the dribble skill by combining the multiple kick modules described above. The stop kicking and rotation kicking is very important in dribble action. And we also use the relative position estimation to predict the future position of the ball and the dribbling player himself. Our dribble skill is totally hand coded, combining them and using simple rules so that the ball is kept to the opposite side from the closest opponent player. Therefore, our current dribble skill enables the player to keep the ball from only one opponent player. Since the dribble skill is very important in the game, it is necessary to develop the higher level one.

3 High-Level approach

3.1 Outline

The main idea behind the team's strategy lies in the question whether the players can be taught to make quick and clever decisions when passing other players. How can the players be taught to think like humans? The tools used are a visual debugger and a training program, from which one can access and manipulate situation data, and a learning program which uses this data to recognize, analyze and generalize situations in order to help the players making suitable decisions.

In short, old games are run through a visual debugger, in which a specific player can be told who to pass in any given situation. This added to the information from the respective player's log will result in a new collection of data, from which a learning program, will use a special algorithm to generate a set of rules that will aid the players in their decisions. Hence the players don't actually make the decisions themselves, but it is left to a separate module, which main purpose is to recognize and generalize a given situation.

3.2 Visual Debugger and Training Program

The visual debugger, which uses data from log files to replay an old game, can be stopped at any time. This makes it easier for a human to analyze the situation. To reduce the development time, we use the SoccerScope which was developed by the team YowAI[3] as a template of our visual debugger. For our purpose, we added many features to SoccerScope. Fig.1 shows the screen-shot of the changed SoccerScope.

Using a training program one can give advice, by clicking the mouse, to a specific player in focus on what to do. Fig.2 shows the sample training menu. Ergo, the human becomes the coach. This has advantages, since normally it is difficult for a computer program to analyze a situation. The human brain can easily generalize and abstract information. The advice given by the human is sent to a data base along with information provided by the player in focus. This information consists of attributes such as the other player's team belonging, position, speed, direction and neck direction. After having repeated this a large number of times, the learning program can now process the data.

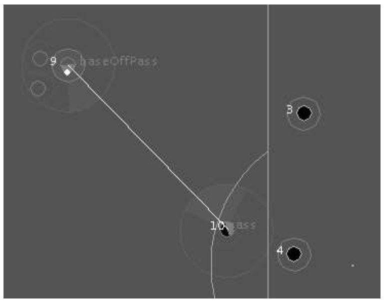

Fig. 1. screen-shot of changed SoccerScope

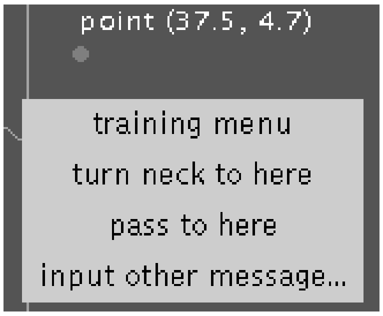

Fig. 2. sample of training menu

3.3 Learning Program

This program uses decision tree learning, which is a method for approximating discrete-valued target functions, in which the learned function is represented by a decision tree. At first, we use the ID3[5]. The main point of this algorithm is to select which attribute, such as the players' position or speed, to test at each node in the tree. The tree is built by first calculating what attributes are most important.

To find out the quantitative measure of the value of the attribute, ID3 is making use of a statistical property, called *information gain*, that measures how well a given attribute separates the training examples according to their target classification. In other words, it is the *information gain* function that calculates how quickly an attribute will result in a positive output. The advantage of this tree is that important attributes will be tested first and this will result in quicker positive outputs.

3.4 Advantages and Disadvantages

The advantages of our system are that the players will be provided a set of rules applied to the information they have about the situation on the field. These rules are generalized and therefore it would require less analyzing before coming to a decision. Also the way in which the decision tree is built results in even quicker decisions. It is also our hope that the players will actually start “thinking” like humans. Although this might be a difficult task, since an enormous amount of data would be required to even begin generalize situations and since it is hard to generalize situations because of the complexity of such a multi-agent system, it is an appealing idea that perhaps human intuition could somehow be baked into such learning algorithms.

These decision trees can be built based on different parts of the field and on different situations, making it easier to generalize more specific situations. This approach can prove to be an advantage in pre-set situations such as corner-kicks and free-kicks.

4 Conclusion and Future Work

Our Low-Level approach is almost all finished, but still required to improve some complex skills. We will try to apply some learning or optimization method. We think that it is more effective to apply the evolutional method to the basic skills than the cooperative behavior in the Soccer Server System.

Our High-Level approach has just started. Our team’s strategy will be based on learning from human coaching so that the players will start thinking like humans, making decisions based on decision trees. Our approach will hopefully result in a more clever and intuitive game play.

References

1. Hidehisa Akiyama, Tomoharu Nagao.: An Optimization Method of Soccer Players’ Behavior using Genetic Programming. JSAI SIG-Challenge-0111, pp,35-40. 2001
2. Peter Stone, et al. CMUnited99’s source code.: http://www-2.cs.cmu.edu/ pstone/RoboCup/CMUnited99-sim.html
3. Takashi Suzuki, Koji Nakayama, et al. YowAI’s source code and Soccerscope.: http://ne.cs.uec.ac.jp/ koji/
4. Luis Paulo Reis and Nuno Lau. FC Portugal team Description: RoboCup 2000 Simulation League Champion. In Stone P., Balch T., Kaetzschmar G., editors, RoboCup 2000:Robot Soccer World Cup IV. pp.29-41, Springer Verlag, Berlin, 2001
5. Quinlan, J. R. Induction of decision trees. Machine Learning, 1(1), pp81-106, 1986