HELIOS2007 Team Description

Hidehisa Akiyama

Information Technology Research Institute,
National Institute of Advanced Industrial Science and Technology
Ibaraki, Japan
hidehisa.akiyama@aist.go.jp

Abstract. HELIOS2007 is a 2D soccer simulation team which has been participating in the RoboCup competition since 2000. This paper describes the overview and the research focus of HELIOS2007. Our main goal is to develop a more realistic simulated soccer team. In particular, we are interested in applying a human's training operations without programming knowledge into the soccer agent's decision making. For this purpose, some GUI tools have already been developed. And, we propose a novel positioning mechanism, which utilizes the Delaunay Triangulation and can be adjusted by human's intuitive operations using our GUI tool. Our positioning mechanism has many advantages than previous methods.

1 Introduction

HELIOS2007 is a 2D soccer simulation team which has been participating in the RoboCup competition since 2000. Our former team name is TokyoTechSFC, that is ranked 3rd in RoboCup2005 and ranked 4th in RoboCup2006. Our code is written by C++ and implemented from scratch without the source code of any other simulated soccer teams. And also, we are developing several tool programs that helps us to develop a team and to make a experiment environment.

Our main goal is to develop a more realistic simulated soccer team. In particular, we are interested in applying a human's training operations without programming knowledge into the soccer agent's decision making. In the last few years, we mainly concentrated on the training of agent positioning behavior using a human's instruction. In order to realize this, we applied a novel positioning mechanism which utilizes Delaunay Triangulation.

The setup of this paper is as follows. In section 2, we will introduce our open source project. In section 3, we will describe our proposal positioning mechanism. And in section 4, we will end with a conclusion.

2 Development Project

We are developing three program packages as an open source project:

– librcsc, a base library for the simulated soccer agent and related tools.

– agent2d, a basic agent program that can work as a simulated soccer team.
– soccerwindow2, a viewer program for the soccer simulation.

These packages have already been released at SourceForge.jp[1]. We are planning to maintain them continuously. We hope that our programs help a new team to participate the RoboCup event and to start a research about the multi-agent simulation using the RoboCup soccer simulator[1].

2.1 librcsc

librcsc is a base library for us to develop a simulated soccer team and related tool programs. librcsc contains several modules for the RoboCup soccer simulation such as geometry, network connection, synchronization and communication with server, world model, basic actions, log parser, and so on.

librcsc can be used as the framework for the simulated soccer team. HELIOS2007 also uses librcsc.

2.2 agent2d

agent2d is a basic agent program that can work as a simulated soccer team. We implemented a simple behavior for each player. That behavior is more complicated than UvA base code[8]. Each player can intercept, dribble, pass and shoot according to the current situation. The team strategy is still simple, but the team performance is so good. Our proposal positioning mechanism described in the next section is also implemented in agent2d. agent2d can be used as a good start point for the new team.

2.3 soccerwindow2

soccerwindow2 is a viewer program for the RoboCup soccer simulator. soccerwindow2 has many useful features. For example. the following functions help us to develop a team:

– It can work as a monitor client compatible with the official rcssmonitor.
– It can work as a stand-alone log player.
– It can work as a visual debugger. An online debug server and an offline debug message viewer are available.
– It can work as a formation editor. We can edit the desired formation using a GUI.

Figure 1 shows the snapshot of soccerwindow2. In this figure, the online debug server is active and the one player's internal state is shown.

soccerwindow2 is developed using Qt, which is a cross-platform application development library[6]. So, we can use soccerwindow2 in several systems. Now, we support Linux and MacOSX and Windows.

[1] http://sourceforge.jp/projects/rctools/

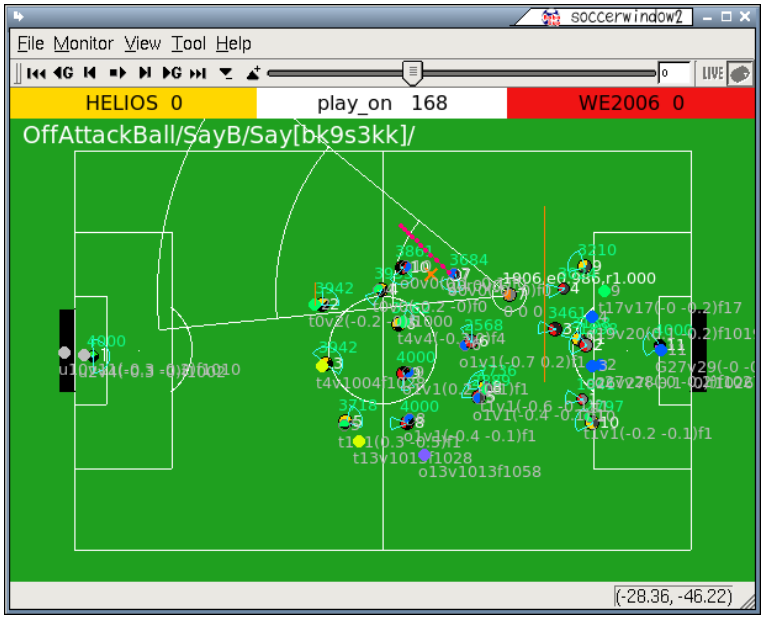

Fig. 1. Snapshot of soccerwindow2

3 Positioning Mechanism using Delaunay Triangulation

In the RoboCup Soccer Simulation domain, Situation Based Strategic Position(SBSP)[7] is well known as the player agents' positioning mechanism, that the ball position is used as the attention target point. SBSP uses the simple function that output the player's move position. This function uses the the ball coordinate value as a main input value and outputs the player agent's move position. If we assume that all players always pay attention to the ball, the cooperative behavior can be done indirectly.

The problem of SBSP is that the output value is depend on the used function. In the basic SBSP algorithm, the characteristic of agent's positioning behavior also becomes simple because the simple linear function is used. If we want the more complicated player agent's positioning behavior, we need to prepare the several parameter set for each situation. But, it is difficult for us to manage such many parameters and the relation of the situations correctly.

In order to solve these problems, we propose a novel positioning mechanism that utilizes Delaunay Triangulation[3] and the linear interpolation algorithm. In this model, the region is divided into several triangles based on the given training data, and each training data affects only the divided region where it belongs to. So, the proposal model is locally adjustable.

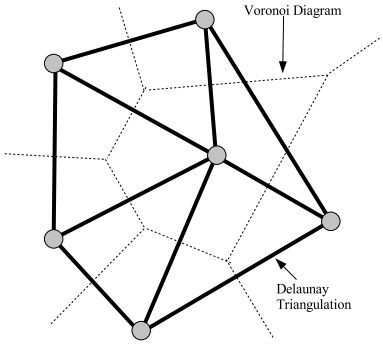

Fig. 2. Example of Delaunay Triangulation

3.1 Delaunay Triangulation

Delaunay Triangulation is one of the method to triangulate tha plane region based on the given point set. Delaunay Triangulation for a set P of points in the plane is a triangulation $DT(P)$ such that no point in P is inside the circumcircle of any triangle in $DT(P)$. Delaunay Triangulation maximize the minimum angle of all the angles of the triangles in the triangulation. So, we can get the most stable triangles from Delaunay Triangulation. Figure 2 shows the example of Delaunay Triangulation. In this figure, Voronoi Diagram is also shown. There is a duality between Voronoi Diagram and Delaunay Triangulation.

If the size of given points is more than 3, we can get a unique triangulation. The several algorithms to calculate Delaunay Triangulation are known and the time complexity of the fastest one is $O(n \log n)$. So, if the number of points is hundreds of levels, we can calculate Delaunay Triangulation in the real time. In librcsc, we implemented one of the fastest algorithm, the incremental algorithm[3].

In our method, the ball positions in training data are used as the vertices of triangles. Each vertex has the output value as the agent's move position for that vertex(=ball) position. When the ball is contained by one triangle, player agent's move position is calculated by interpolation algorithm described in next section.

3.2 Linear Interpolation Algorithm

We use the simple linear interpolation algorithm to calculate the agent's move position. This algorithm is same as Gouraud shading algorithm[2]. Gouraud shading algorithm is a method used in computer graphics domain to simulate the differing effects of light and color across the surface of an object.

Figure 3 shows the process of Gouraud shading algorithm. The output values from vertices P_a, P_b and P_c are $O(P_a)$, $O(P_b)$ and $O(P_c)$ respectively. Now, we want to calculate $O(B)$, the output value of the point B contained by the triangle $P_aP_bP_c$. The algorithm is as follows:

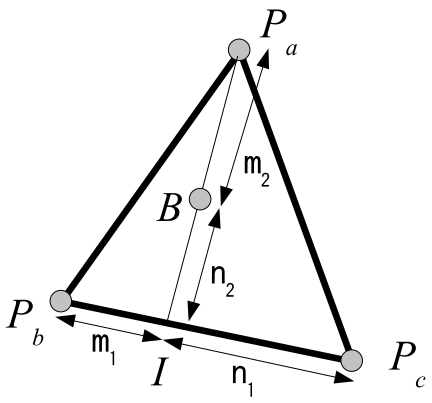

Fig. 3. Liner interpolation by Gouraud shading algorithm

1. Calculates I, the intersection point of the segment P_bP_c and the line P_aB.
2. The output value at I, $O(I)$, is calculated as:

$$O(I) = O(P_b) + (O(P_c) - O(P_b))\frac{m_1}{m_1 + n_1}$$

   where $|\overrightarrow{P_bI}| = m_1$ and $|\overrightarrow{P_cI}| = n_1$.
3. $O(B)$ is calculated as:

$$O(B) = O(P_a) + (O(I) - O(P_a))\frac{m_2}{m_2 + n_2}$$

   where $|\overrightarrow{P_aB}| = m_2$ and $|\overrightarrow{BI}| = n_2$.

3.3 FormationEditor

To compose desired training data set, it is necessary to use human's observation and their intuitive input. In order to enable us to do such operations, we developed a GUI tool, FormationEditor, as an compornent of soccerwindow2. Figure 4 shows the screenshot of FormationEditor. FormationEditor can visualize the training process and enables us to edit the training data easily.

3.4 Advantages

The proposal positioning mechanism has following advantages:

- High approximation accuracy. Our method realizes higher accuracy than other known function approximator. Especially, the relation of input and output in training data is completely guaranteed.
- Fast running. If the number of given training data is hundreds of levels, it is possible to use it in real time.

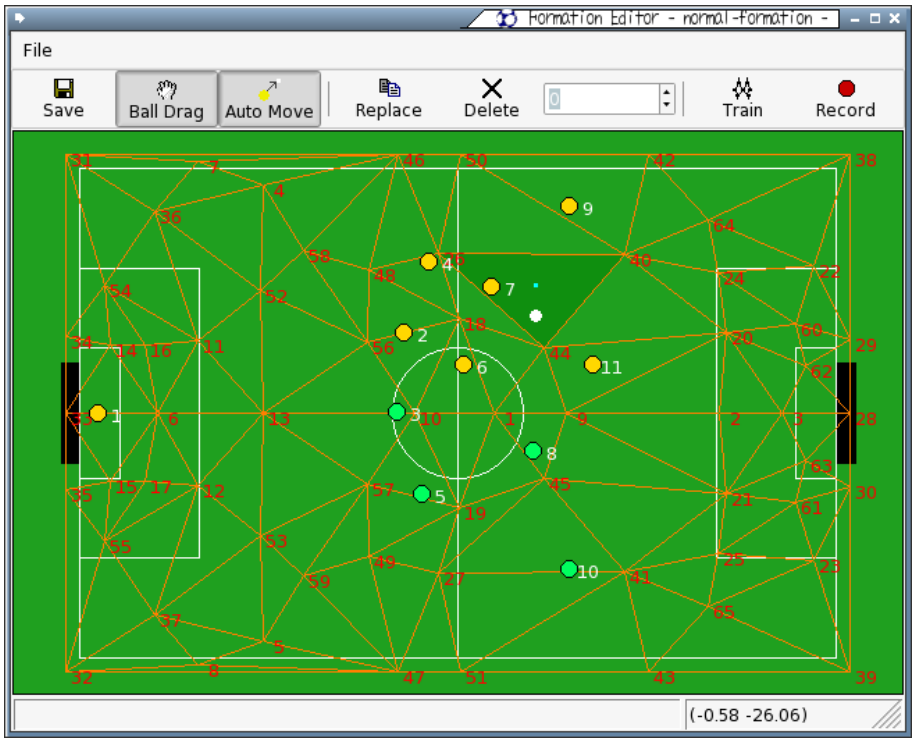

Fig. 4. The main window of FormationEditor. The example training data set is shown in the window.

- Locally adjustable. Even if a new training data is added or existing data is modified, the triangle region where that data is not contained is never affected.
- Easily understandable. We can easily understand the acquired characteristic and estimate the effect of each training data.
- High flexibility. If we want more smooth approximation or sharp gradient, we just need to add more training data. If we are satisfied with the current result, it is not necessary to improve the density of the training data.
- High scalability. Even if the considered plane region is extended or shrinked, it is easy to deal with the new plane region without any changes to the old data.
- Complete reproducibility. If the training data is same, we can acquire the completely same result.

Especially, the complete reproducibility is an important advantage. Because there is no limitation of the input order of the training data, any training data can be added at any time. This means that human can intervene at any time.

4 Conclusion and Future Directions

This paper described our simulated soccer team, HELIOS2007, and the related programs. And, our research focus and current status were also described.

We can say that our proposal positioning mechanism has many advantages obviously. However, the proposal method has the following disadvantages:

- The proposal method requires many memories. Other known function approximator such as the three layer perceptron, Radial Basis Function network[5], Normalized Gaussian network[4] and so on, absorb the training data into the network parameters. So, the training data can be compressed significantly. On the other hand, our proposal method have to store the all training data and always use them.
- The proposal method requires the high cost to keep the consistency of the training data. If the number of divided region is increased, the smooth positioning behavior can be acquired. But, the number of training data is also increased. If one of them is modified, the near training data may be also required to modify.

We plan to develop the method to adjust the training data automatically in order to reduce the management cost of the training data set. At least, we need the method to help us to find the inconsistent training data.

And also, we should consider about the multiple dimensional input and output. Now, our proposal method can handle only two dimensional input and output. For example, if we can handle other player agents' positions as an input and the other decision making parameters as an output, it will be very useful. However, if the dimension is increased, the visualization becomes difficult and it is difficult for us to understand the characteristic. So, we should consider about the method to compress the dimension or to overlap the information.

References

1. The RoboCup Soccer Simulator. http://sserver.sourceforge.net/.
2. Henri Gouraud. Continuous shading of curved surfaces. In Rosalee Wolfe, editor, *Seminal Graphics: Pioneering efforts that shaped the field*. ACM Press, 1998.
3. Marc van Kreveld Mark de Berg, Otfried Schwarzkopf and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer Verlag, second edition, 2000.
4. John Moody and Christian Darken. Fast learning in networks of locally-tuned processing units. Number 1, pages 289–303, 1989.
5. Tomaso Poggio and Federico Girosi. Networks for approximation and learning. Number 78, pages 1481–1497, 1990.
6. Qt: http://www.trolltech.com/products/qt/.
7. Luis Paulo Reis, Nuno Lau, and Eugenio Olivéira. Situation Based Strategic Positioning for Coordinating a Simulated RoboSoccer Team. *Balancing Reactivity and Social Deliberation in MAS*, pages 175–197, 2001.
8. UvA Trilearn 2003 Base Source: http://www.science.uva.nl/~jellekok/robocup/2003/.