HELIOS2009 Team Description

Hidehisa Akiyama, Hiroki Shimora, and Itsuki Noda

Information Technology Research Institute, AIST, Japan
{hidehisa.akiyama,h.shimora,i.noda}@aist.go.jp

Abstract. HELIOS2009 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 HELIOS2009. The current our research focus is to utilize human's knowledge to acquire a team strategy and a tactics. We have proposed an agent positioning mechanism based on a triangulation method. This model enables us to understand the influence of our operations. We are trying to extend our model to solve known issues and to apply our model to more complicated problems.

1 Introduction

HELIOS2009 is a simulated soccer team for the RoboCup soccer 2D simulator(RCSS)[1, 2], which has been participating in the RoboCup competition since 2000. Our code is written by C++ and implemented from scratch without a source code of any other simulated soccer teams. We are developing several tool programs that helps us to develop a team and to make a experiment environment. Almost all of them have already been freely available.

Our main goal is to develop a more realistic simulated soccer team. In particular, we are interested in applying human's knowledge into the agent's decision making without any programming knowledge. In recent years, we mainly focused on acquiring agent positioning strategy using human's instructions. In order to realize this, we applied a novel positioning mechanism which utilizes Delaunay Triangulation. In this year, we extended our model using Constrained Delaunay Trianglation.

The setup of this paper is as follows. In section 2, we will introduce the overview of our released software. In section 3, we will describe our positioning mechanism. In section 4, we will describe the extended model. And, in section 5, we will end with a conclusion and future directions.

2 Released Software

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

– librcsc: a base library to develop a client program for RCSS. librcsc can be used as a framework for a simulated soccer team. librcsc is licensed under LGPL.

- soccerwindow2: a viewer program for RCSS. soccerwindow2 can work as a monitor client, a log player and a visual debugger. And, soccerwindow2 package contains one more component, fedit, that is an editing tool to compose a team formation using human's intuitive operations. Because soccerwindow2 is developed using Qt[3] which is a cross-platform application development library, soccerwindow2 can run on several platforms. soccerwindows is licenced under GPL.
- agent2d: a sample agent programs that use librcsc. agent2d can work as a simple simulated soccer team. The behavior of agent2d is still simple but is more complicated than UvA base code[4]. The team strategy is also simple, but the performance is rather good. agent2d can be used as a good start point for new teams. agent2d is licensed under GPL.

Because all programs are written in Standard C++, support POSIX and use GNU autoconf, automake and libtool as build tools, these packages have high portability.

These packages have already been freely available[1]. We hope that our released software helps a new team to participate the RoboCup event and to start a research of multiagent systems using RCSS.

3 Triangulation based Positioning Mechanism

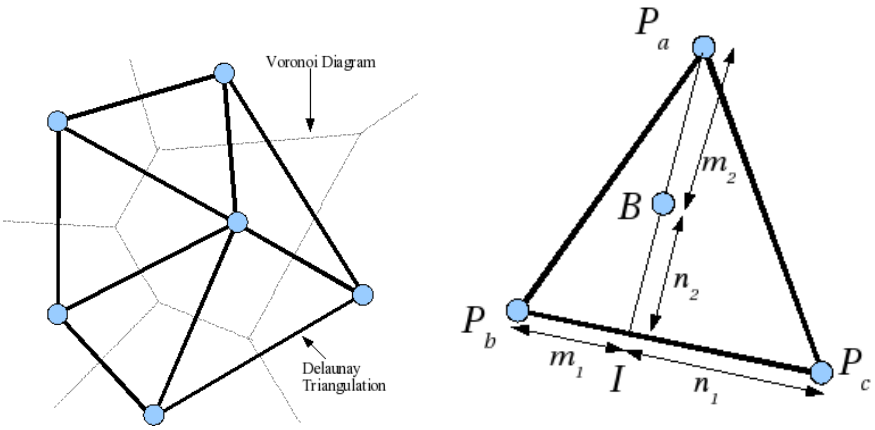
We have proposed a function representation model to define a team formation, which utilizes Delaunay Triangulation and a linear interpolation algorithm[5, 6]. The main idea of our model is similar to Situation Based Strategic Positioning[7]. An input value of the function is a focal point on the soccer field, usualy the ball position, and an output value of the function is agents' strategic positions according to the input values. In our model, an input plane region (a soccer field) is divided into several triangles according to given training data. Such triangulated regions enable us to understand the influcence of each training data easily.

3.1 Delaunay Triangulation

Delaunay Triangulation[8] is one of the method to triangulate a plane region based on a 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 maximizes 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 1(a) shows the example of Delaunay Triangulation. This figure also shows Voronoi Diagram. 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. There are several algorithms to calculate Delaunay Triangulation and the

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



(a) Delaunay Triangulation.

(b) Interpolation using Gouraud shading algorithm.

Fig. 1. Delaunay Triangulation and Linear Interpolation.

time complexity of the fastest one is $O(n \log n)$. Therefore, 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[8].

3.2 Linear Interpolation

In our model, one training data is corresponding to one vertex in the triangulation. So, each vertex has a ball position as an input value and has agents' positions according to the vertex (ball) position as output values. When an unknown input value is given, output values are calculated by Gouraud shading algorithm[9]. 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 1(b) 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:

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 $|\overrightarrow{P_cI}| = n_1$.

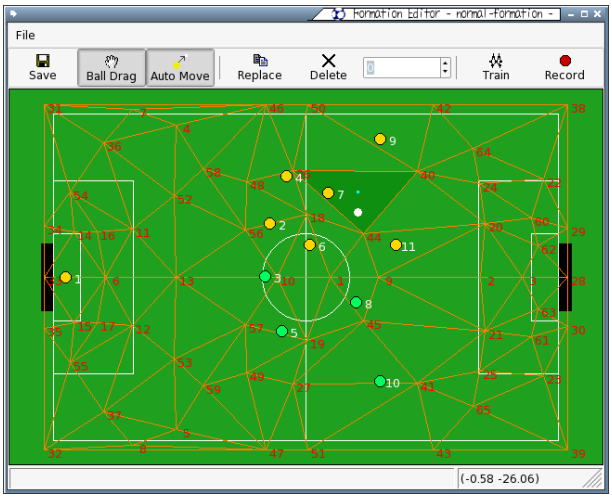


Fig. 2. The main window of fedit.

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 $|\overrightarrow{BI}| = n_2$.

Using the above interpolation algorithm, if a certain triangle in the triangulation contains an unknown input value(ball position), output values(agents' positions) can be calculated according to the vertices of the triangle.

3.3 Formation Editing Tool

We have been developing a team formation editing tool, fedit, as a part of soccerwindow2. fedit enables us to intuitively compose desired agents' positions according to the ball position. These data can be used as the training data for our positioning mechanism. Figure 2 shows the screenshot of fedit. fedit can visualize the training process and enables us to edit the training data easily.

3.4 Problems

Our model is simple, fast and easily understanble. Moreover, it guarantees the continuity of output values. However, our model has two problems. 1) It cannot fix the existing topological relationships between training data. We may need a redundant training data to keep the consistency. 2) It has several difficulties when dealing with large data set or noisy data.

To solve the first problem, we propose the extended model using Constrained Delaunay Triangulation. And, we are trying to apply some unsupervised learning methods to solve the second problem.

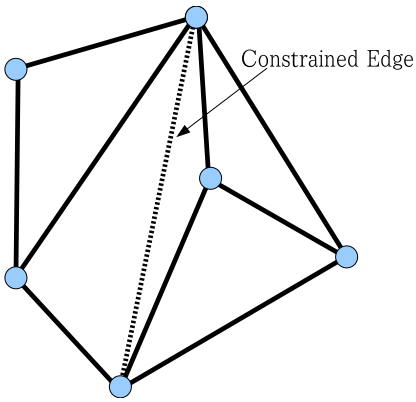


Fig. 3. Constrained Delaunay Triangulation.

4 Extende Model using Constrained Delaunay Triangulation

Constrained Delaunay Triangulation (CDT)[10] is a generalized Delaunay Triangulation. The difference between CDT and Delaunay Trianglation is the existence of constrained edges. If there are no constraints, the result of CDT is completely equivalent to the result of Delaunay Triangulation. Figure 3 shows an example of CDT. In this figure, the point set in the plane is same as figure 1(a), but one constrained edge (the broken line in the figure) is given. As shown in the figure, unlike Delaunay Triangulation, CDT has some triangles that their circumcircle contains other triangle's vertices.

Our extended model utilizes CDT instead of Delaunay Triangulation. The extended model enables us to add a training data by not only a point but a line segment. This means we can fix the topological relationship between two training data. Moreover, because CDT can be applied to any polygonal region that does not compose a convex hull, the extended model can be applied to more complicated problems.

However, our extended model cannot accept crossed constrained edges. In CDT, if there are crossed constrained edges, new vertices are created at the crossed point automatically. But, these additional vertices break a continuity of output values and a consistency of topological relationships. So, in our extended model, if there are crossed constrained edges, they are rejected as an incorrect training data.

One more advantage of using CDT is that we can easily apply Neural Gas[11] or Growing Neural Gas[12] to our extended model. These methods are known as an unsupervised learning method and can generate a set of Delaunay Triangulation that represents a characteristic of input point clouds. Although we need more investigation, these unsupervised learning methods are promising to solve the problem of large and noisy data.

5 Conclusion and Future Directions

This paper described our simulated soccer team, HELIOS2009, and related programs. We also described our research focus and current effort. We proposed a function representation model to decide strategical multi-agent/robot positions using triangulation methods and a linear interpolation algorithm. We are trying to extend our positioning mechanism in order to solve the problem of large and noisy data. The another issue is to deal with a tactical behavior. Our model can define a strategical characteristic, but does not focus on a tactical behavior. It is necessary to formalize the representation model that can deal with a tactical behavior.

References

1. Noda, I., Matsubara, H.: Soccer server and researches on multi-agent systems. In Kitano, H., ed.: Proceedings of IROS-96 Workshop on RoboCup. (Nov. 1996) 1–7
2. The RoboCup Soccer Simulator. http://sserver.sourceforge.net/
3. Qt - a cross-platform application and UI development framework. http://www.qtsoftware.com/
4. UvA Trilearn Base Source. http://www.science.uva.nl/~jellekok/robocup/2003/
5. Akiyama, H., Katagami, D., Nitta, K.: Training of agent positioning using humans's instruction. Volume 11. (2007)
6. Akiyama, H., Noda, I.: Multi-agent positioning mechanism in the dynamic environment. In: RoboCup 2007: Robot Soccer World Cup XI. (2008) to appear.
7. Reis, L.P., Lau, N., Olivéira, E.: Situation Based Strategic Positioning for Coordinating a Simulated RoboSoccer Team. Balancing Reactivity and Social Deliberation in MAS (2001) 175–197
8. Mark de Berg, Otfried Schwarzkopf, M.v.K., Overmars, M.: Computational Geometry: Algorithms and Applications. Second edn. Springer Verlag (2000)
9. Gouraud, H.: Continuous shading of curved surfaces. In Wolfe, R., ed.: Seminal Graphics: Pioneering efforts that shaped the field, ACM Press (1998)
10. Chew, L.P.: Constrained delaunay triangulations. In: SCG '87: Proceedings of the third annual symposium on Computational geometry, New York, NY, USA, ACM (1987) 215–222
11. Martinetz, T., Schulten, K.: A "neural-gas" network learns topologies. Artificial Neural Networks **I** (1991) 397–402
12. Fritzke, B.: A growing neural gas network learns topologies. In: Advances in Neural Information Processing Systems 7, MIT Press (1995) 625–632