# HELIOS Base: An Open Source Package for the RoboCup Soccer 2D Simulation

Hidehisa Akiyama[1] and Tomoharu Nakashima[2]

[1] Faculty of Engineering, Fukuoka University, Japan
akym@fukuoka-u.ac.jp
[2] Department of Computer Science and Intelligent Systems,
Osaka Prefecture University, Japan
tomoharu.nakashima@kis.osakafu-u.ac.jp

**Abstract.** To promote the research of multiagent systems, several base codes have been released for the RoboCup soccer 2D simulation community. As described herein, we present HELIOS base, currently the most popular base codes for 2D soccer simulation. HELIOS base involves a common library, a sample team, a visual debugger, and a formation editor, which help us to develop a simulated soccer team.

## 1 Introduction

In this paper, we present a base code, named HELIOS base, for the RoboCup soccer 2D simulation. The RoboCup Soccer Simulation 2D League is a long-running competition among the RoboCup leagues. It is based on the RoboCup Soccer 2D Simulator [8,11], which enables two teams of 11 autonomous player agents and an autonomous coach agent to play a game of soccer with highly realistic rules and real-time game play. Because of its stability, the 2D soccer simulator is extremely useful for research and education related to multiagent systems, artificial intelligence, and machine learning.

The soccer simulation league has devoted more attention to team work techniques than to robot control techniques. The 2D soccer simulator adopts a discrete timer model and an abstract and simple kinematic model, although its virtual soccer field has a continuous space. Therefore, we can avoid the burdens of developing and maintaining mechanical devices and also developing complex robot control tasks such as bipedal walking. These characteristics enable us to concentrate on research efforts related to multiagent systems. However, developing an agent program from scratch is as difficult a challenge as ever because other complex modules, such as a stable network communication, synchronization, world modeling, and so on, are necessary to produce an agent program that fully functions in the soccer simulator. We must resolve these technical problems before progressing with research of multiagent systems. The base code presented in this paper provides a framework that enables us to concentrate on teamwork techniques.

The remainder of this paper is organized as follows. Section 2 introduces the base code released by other teams. Section 3 introduces an outline of our

S. Behnke et al. (Eds.): RoboCup 2013, LNAI 8371, pp. 528–535, 2014.

base code. Section 4 describes our base code components. Section 5 describes the impact of our base code on the 2D soccer simulation community. Section 6 concludes this report.

## 2 Related Works

In the 2D soccer simulation community, several teams have released (parts of) their respective source codes to promote the research of multiagent systems using the soccer simulator. Especially, the champion teams often release their source codes after competitions.

CMUnited [10,6] is an important code release in the early 2000s. This team was the RoboCup champion of 1998 and 1999. Their released code was widely used as a base by teams around the world. Its concept, such as Locker Room Agreement, Layered Learning, and so on, still have an impact on the development of simulated soccer agents. The base code released by TsinghuAeolus [14], the champion team of RoboCup 2001 and 2001, provides excellent skills such as ball kicking and dribbling. However, because the code was designed for a Windows environment at first, it did not capture the global popularity. UvA Trilearn [7,12], the champion team of RoboCup 2003, has released an extremely successful base code. Their code provides a sophisticated design and extremely rich documentation. Consequently, a large number of new teams have managed to participate in the competition. One team still uses this base code in 2013. Recent champion teams such as Brainstormers [9,5] and WrightEagle [4,13], also released their base code. Although their codes have sophisticated implementation and provide satisfactorily high performance, none is widely used yet because it is difficult for new users to use them.

## 3 Outline of HELIOS Base

HELIOS base is a base code and related development tools released by HELIOS, which is the champion team of RoboCup 2010 and 2012 [1].

### 3.1 History

HELIOS is a simulated soccer team for the RoboCup soccer 2D simulation league. The team, a joint team of Fukuoka University and Osaka Prefecture University since 2010, has been participating in RoboCup competitions since 2000. The team has won 2 championships and 2 runner-up places to date, and has remained among the top 3 in the world championships since 2007.

The first release of HELIOS base was in 2006. The code is still being maintained continually according to the change of competition rules. HELIOS base is said to be the most popular base code in 2012. All source codes are available at our project site http://sourceforge.jp/projects/rctools/.

### 3.2 Features

HELIOS base provides a sample source code for developing a team and a data set that can run as a simple but competitive team, for new teams to participate to the competition of the 2D soccer simulation easily. HELIOS base consists of several software components designed to reduce the maintenance cost. an overview of each component is described in Section 4.

All software components included in HELIOS base are written using Standard C++ and implemented from scratch without the source codes of other simulated soccer teams. The code depends on the POSIX API, the boost C++ libraries[1], and Qt[2], but never includes environment specific dependencies. Therefore, HELIOS base has high portability to various operating systems. Now, HELIOS base supports Linux, Mac OSX, and Windows (Cygwin).

The users of HELIOS base can use it freely if they follow its license. The common library of HELIOS base is licensed under GNU Lesser General Public License[3]. The code of the sample team and development tools are licensed under GNU General Public License[4].

## 4 Components

HELIOS base provides the following components:

- librcsc
- agent2d
- soccerwindow2
- fedit2

This section presents a description of the overview of each component.

### 4.1 librcsc: The Common Library

librcsc (LIBrary for the RoboCup Soccer simulation Client) is a basic and common library for developing a 2D soccer simulation software. librcsc contains several library files such as geometry, network interface, communication and synchronization with simulator, world model, basic actions, log parser, debug message management, formation model, and so on.

librcsc encapsulates almost all things related to the communication between the simulator and agent programs. The users of HELIOS base need not consider the synchronization problem related to the network programming and the timing of decision making, which are usually out of focus from the viewpoint of multiagent research.

[1] http://boost.org/
[2] http://qt-project.org/
[3] http://www.gnu.org/copyleft/lesser.html
[4] http://www.gnu.org/licenses/gpl.html

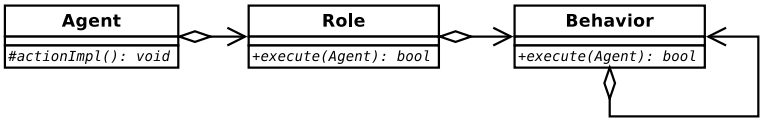


**Fig. 1.** The UML diagram showing the decision making architecture of agent2d

### 4.2 agent2d: A Sample Team

agent2d is a sample team that utilizes librcsc and also contains a data set of the simulated soccer team that can run as a simple but competitive team. The implemented behavior is more complicated than that of UvA base code [7]. Each player can intercept, dribble, pass and shoot by judging from the field situations. Although the team strategies remain simple, the team performance is better than any other sample teams. We assume that this sample team is used as a template when starting team development.

The decision making process of agent2d comprises three layers: agent class, role class and behavior class. Any decision making originates from the agent class. The agent class decides the current strategy and the player's role in the team. The role class is responsible for determining strategic behavior. The role class first gets the current situation and then executes tactical behavior according to the current strategy. Finally, the behavior class performs the actual action. Figure 1 shows the relation among these classes.

Developers of a team never need to implement their own agent class, but might need to implement their own role classes or behavior classes. The default implementation provides several role instances, but in agent2d there is no difference among them. If developers would like to step into more detailed development such as improving or adding new role classes or behavior classes, then they require some knowledge and experience of C++ because agent2d is written entirely in C++.

In agent2d, our formation framework [3] and online multiagent planning framework [2] have already been implemented. These frameworks enable us to change the characteristics of team behavior by modifying the team formation and the evaluation function. Consequently, we can concentrate on improving the team strategy without considering a complicated decision tree. For example, we can focus on the optimization of team formation only by changing parameters in configuration files. The multiagent planning framework was introduced into agent2d in 2010. This framework brings up several research issues related to the online search approach in a continuous state and action space.

### 4.3 soccerwindow2: A Visual Debugger

soccerwindow2, a viewer program for the 2D soccer simulation, has many useful features. For example, the following functions help us to develop a team:

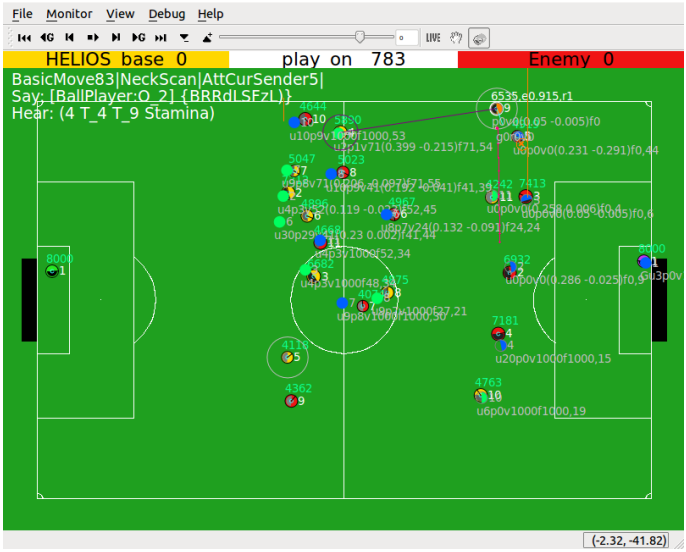


**Fig. 2.** soccerwindow2 is useful not only as a compatible monitor client but also as a visual debugger. This image shows the actual simulation state and one agent's internal state on the virtual soccer field.

- It can work as a monitor client that is compatible with the official monitor client. The timeshift function is also available. We can always replay any recorded scene during the game online.
- It can function as a stand-alone log player. The game log files recorded by the 2D simulator can be replayed.
- It can function as a visual debugger: not only as an online debug server but also as an offline debug message viewer. The agents' internal state can be examined during the game online if agents send their internal state to the integrated visual debug server. Moreover, if agents record their internal state with the specified format as their own log files, then soccerwindow2 can load and replay them.

Figure 2 portrays a snapshot of soccerwindow2 in which the online debug server mode is active. As the figure shows, soccerwindow2 can visualize not only the actual simulation state but also the agents' internal state on the virtual soccer field. This feature helps us to observe a gap separating the actual simulation state from the agents' internal state. This debugging information can be sent from agent programs to the visual debug server integrated to soccerwindow2 via UDP/IP communication during the game. The debug server function facilitates our development process to a considerable degree.

Figure 3 presents a snapshot of the debug message window, which can show agents' more detailed internal states by unlimited length text message with arbitrary format. We assume that the text messages are loaded from files recorded

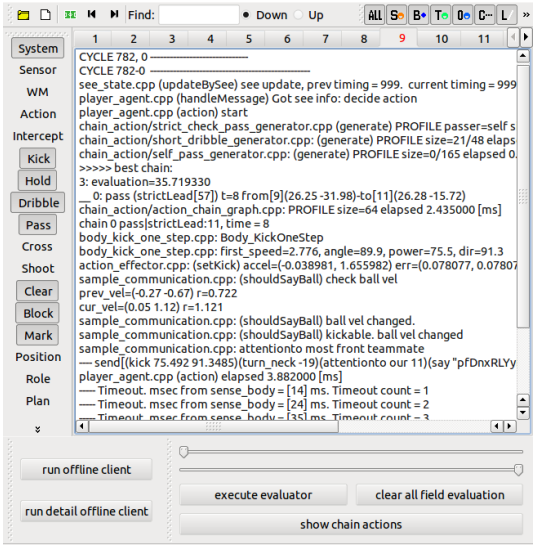


**Fig. 3.** Snapshot of the debug message window that can show the agents' internal state by text with greater detail. We can check all agents' respective states by changing the tab window. The left pain shows log level buttons that can toggle the information type shown in the main message panel.

by agents. Therefore, this feature cannot be used during the game. However, this feature helps us to recognize more details related to agents' decision making process.

soccerwindow2 is developed using Qt, which is a cross-platform application development library. Therefore, we can use soccerwindow2 in several systems.

### 4.4 fedit2: A Formation Editor

fedit2 is a GUI application to edit the formation data for agent2d. agent2d supports the formation framework [3] provided by librcsc. This framework enables the definition of the team formation through external configuration files. Here, agent2d can change the team formation easily by loading different configuration files. However, this framework defines the team formation using Delaunay triangulation. Therefore, it is difficult for us to modify the team formation by editing the text data. In addition, fedit2 helps us to modify the team formation.

Figure 4 portrays a snapshot of fedit2, which can not only display the ball and 11 player agents on the soccer field, but can also enable them to be edited intuitively by the mouse and keyboard. The recorded ball positions are used as vertices of Delaunay triangulation. The agents' positions are calculated using a linear interpolation algorithm, resembling the Gouraud shading algorithm if the input ball position is unknown and if it is contained by one triangle. For more details of this framework, another report of the literature is helpful [3].

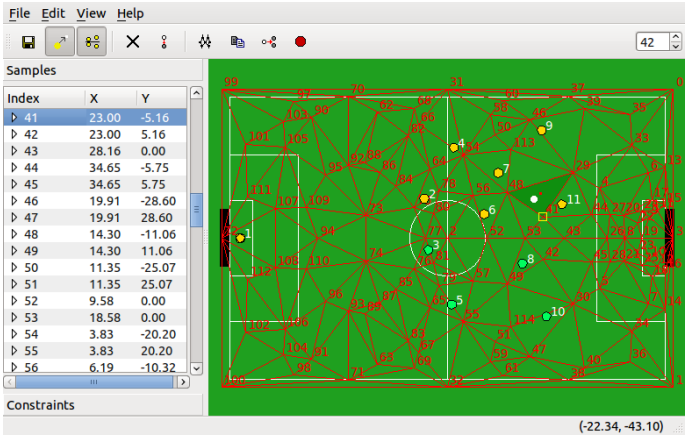


**Fig. 4.** Snapshot of fedit2. The left panel shows a list of recorded data sets. Each row contains the ball position and ideal agent positions according to the ball position. The right pane visualizes Delaunay triangulation constructed from the recorded data. Each vertex represents the ball position of input data. We can modify them easily using a mouse and keyboard on the soccer field.

**Table 1.** Transition of the number of teams that use HELIOS base

| | # of participating teams | # of HELIOS based teams | Percentage(%) |
|---|---|---|---|
| RoboCup2007 | 15 | 3 | 20.0 |
| RoboCup2008 | 15 | 4 | 26.7 |
| RoboCup2009 | 19 | 7 | 36.5 |
| RoboCup2010 | 19 | 8 | 42.1 |
| RoboCup2011 | 17 | 8 | 47.1 |
| RoboCup2012 | 18 | 15 | 83.3 |
| RoboCup2013 | 24 | 20 | 83.3 |

## 5 Impact on the Community

Until the release of HELIOS base, the most popular base code in the 2D soccer simulation community was UvA Trilearn. After releasing HELIOS base, the number of teams that use HELIOS base has increased year by year. Table 1 shows the transition of the number of teams that use HELIOS base. Results show that HELIOS base became the most popular base code in 2012.

## 6 Conclusions and Future Works

This paper described HELIOS base, currently the most popular base code for the RoboCup soccer 2D simulation league. HELIOS base involves a common

library, a sample team, and development tools for the 2D soccer simulation to promote the research of multiagent systems.

An important subject for future work is to prepare comprehensive documentation. It is still necessary to improve the design of the decision making architecture.

## References


1. Akiyama, H., Nakashima, T.: HELIOS2012: RoboCup 2012 Soccer Simulation 2D League Champion. In: Chen, X., Stone, P., Sucar, L.E., van der Zant, T. (eds.) RoboCup 2012. LNCS (LNAI), vol. 7500, pp. 13–19. Springer, Heidelberg (2013)
2. Akiyama, H., Nakashima, T., Aramaki, S.: Online cooperative behavior planning using a tree search method in the robocup soccer simulation. In: Proceedings of Fourth IEEE International Conference on Intelligent Networking and Collaborative Systems, INCoS-2012 (2012)
3. Akiyama, H., Noda, I.: Multi-agent positioning mechanism in the dynamic environment. In: Visser, U., Ribeiro, F., Ohashi, T., Dellaert, F. (eds.) RoboCup 2007. LNCS (LNAI), vol. 5001, pp. 377–384. Springer, Heidelberg (2008)
4. Bai, A., Chen, X., MacAlpine, P., Urieli, D., Barrett, S., Stone, P.: Wright Eagle and UT Austin Villa: RoboCup 2011 simulation league champions. In: Röfer, T., Mayer, N.M., Savage, J., Saranlı, U. (eds.) RoboCup 2011. LNCS, vol. 7416, pp. 1–12. Springer, Heidelberg (2012)
5. Brainstormers Public Source Code Release, `http://sourceforge.net/projects/bsrelease/`
6. CMU RoboSoccer RoboCup Simulator Team Homepage, `http://www.cs.utexas.edu/~pstone/RoboCup/CMUnited-sim.html`
7. Kok, J.R., Vlassis, N., Groen, F.: UvA Trilearn 2003 team description. In: Polani, D., Browning, B., Bonarini, A., Yoshida, K. (eds.) Proceedings CD RoboCup 2003. Springer, Padua (2003)
8. Noda, I., Matsubara, H.: Soccer server and researches on multi-agent systems. In: Kitano, H. (ed.) Proceedings of IROS-96 Workshop on RoboCup, pp. 1–7 (November 1996)
9. Riedmiller, M., Gabel, T., Knabe, J., Strasdat, H.: Brainstormers 2d - team description 2005. In: Bredenfeld, A., Jacoff, A., Noda, I., Takahashi, Y. (eds.) Proceedings CD RoboCup 2005. Springer (2005)
10. Stone, P., Riley, P., Veloso, M.: The CMUnited-99 champion simulator team. In: Veloso, M., Pagello, E., Kitano, H. (eds.) RoboCup 1999. LNCS (LNAI), vol. 1856, pp. 35–48. Springer, Heidelberg (2000)
11. The RoboCup Soccer Simulator, `http://sserver.sourceforge.net/`
12. UvA Trilearn (2003), - Soccer Simulation Team, `http://staff.science.uva.nl/~jellekok/robocup/2003/`
13. WrightEagle 2D Soccer Simulation Team, `http://wrighteagle.org/2D/`
14. Yao, J., Chen, J., Cai, Y., Li, S.: Architecture of tsinghuaeolus. In: Birk, A., Coradeschi, S., Tadokoro, S. (eds.) RoboCup 2001. LNCS (LNAI), vol. 2377, pp. 491–494. Springer, Heidelberg (2002)