A Deeper Look at 3D Soccer Simulations

Amin Habibi Shahri, Ali Almasi Monfared, and Mohammad Elahi

Electrical & Computer Engineering Faculty
Shahid Beheshti University, Tehran, Iran
{habibiamin,almasimonfared,elahimohamad}@gmail.com

Abstract. Developing an intelligent agent requires more than an Integrated Development Environment (IDE). In multi agent environments or systems equipped with artificial intelligence it is often difficult to obtain the function or method which led to a particular behavior that is noticeable from outside. In addition to previous dilemma, the publicity that the RoboCup events get from the media provides an ideal opportunity to show the state of art of these systems during RoboCup World Cup.

This paper describes the concept and the implementation of *Team Assistant 2006* as the next generation of TA2002. The idea is to provide a tool that is able to assist developers to detect problems of their agents both in single and cooperation mode and also organizers to have better games.TA2006 won the second place in RoboCup 3D development competition 2006.

Keywords: RoboCup, Soccer Simulation, Presentation, Analyzing.

1 Introduction

Developing an intelligent agent requires more than an Integrated Development Environment (IDE). Many visualization and analysis requirements may arise at any time in the project; and developing a tool to satisfy each requirement is cumbersome and time-consuming.

In the past 10 years the simulation league was two dimensional, all players and even the ball moved on the ground. During this time numerous sophisticated tools were created for analyzing the simulated games such as Logalyzer or Team Assistant[4].

The Logalyzer provides information about detected actions like passes and several visualizations for the collected data about the game. The Team Assistant is able to display information provided via agent logfiles along with statistics about detected actions. The Team Assistant is also mentioned in the 2002 league summary[7] as the winner of the presentation tournament.

In 2003, the 3D simulation was introduced including basic tools to view and replay the simulated game. The tools used in 2D can not be used in 3D simulations because of the lack of one dimension and a different format of the logfiles.

The current monitor is capable of showing the current simulated game (at current time) and of replaying monitor logfiles. The replaying mode can be used

U. Visser et al. (Eds.): RoboCup 2007, LNAI 5001, pp. 294–301, 2008.

to watch previously simulated games again. There is also a "single step mode" which provides slow motion replay. When trying to develop a behavior for an agent or verifying behaviors acquired by machine learning methods it is hard to determine which methods or functions led to the actions observed in the game. This information is crucial when trying to debug or improve the agents in their behavior and collaboration. Especially in the case of collaboration it is tedious and time-consuming to check what each agents intention is. This is caused by the fact that every logfile has to be searched for the right record of the actual time displayed in the monitor by hand. Additionally, the agent logfiles are not numbered according to the uniform numbers of the agents which complicates finding the desired logfile.

The aim of this work as the next generation of Team Assistant[2] is to provide a rich extensible tool, providing decent log playing capabilities alongside useful agent debug and analysis information about agent's behavior and collaboration with other agents. The importance of the evaluation of agent teamwork has been addressed in many papers for 2D simulation[5][8]. It also can be used as an offline trainer to run training sessions with virtually unlimited scenario definition options.

2 Related Work

In the 2006 soccer simulation development competition just a few tools were introduced, such as "Virtual Werder Analyzer" or "UTUtd Monitor".

The Virtual Werder Analyzer offers information about ball possession, successful and unsuccessful passes and about good and bad actions[1]. In UTUtd monitor, there are some common functions with this work such as the possibility of agents to draw into the displayed scene or the displaying ability of text messages according to the current scene[6]. In 2D presentation competitions, tools like Team Assistant also provides the detection of (double-) passes, goal shots, dribbling and etc. Statistics about these detected events are shown while playing game. In Caspian monitor, there is a commentator that can comment the game according to the current detected situation. These comments were the main aspect of that work. But none of the above tools were extensible enough for new requirements.

3 Requirements

In general we can divide 3D Soccer Simulation requirements into two main parts: "Developers' requirements" and "Organizers' requirements".

3.1 Developers' Required Features

In consultation with other agent developers, providing an easy and clear way to gather information about what the agent intends and how the world looks

like, according to the agent, is essentially needed. The following features are judged beneficial and essential to debug handwritten behaviors and to verify the decisions of learned behaviors.

While analyzing a special situation of the game it is obvious that forward and backward replay in different speeds is useful. With this possibility the situation can be analyzed again without starting the game from the beginning until the desired situation is reached. To gain knowledge of the agents' intentions in a situation an output of agents' logfile according to current time is needed.

In addition to the logfile displaying it is valuable to enable the agents to draw information directly into the displayed scene, like a line from the agent to the position it intends to move to. Also filtering the logfile output may be helpful to display only those information needed by the developer. This way only those information provided by the current developed behavior could be displayed. This idea can be achieved by using layered logs[3].

New camera positions, like birdview which resides directly over the agent of interest, may be beneficial. When using the single step mode of the monitor displaying the ball's and player's movements for some time forward can be an improvement, the developer could see the next movements without proceeding forward in scene display. Displaying the offside line is necessary. In some situations it could be difficult to distinguish on which side of the offside line an agent is, so marking agents that are in an offside position if the ball would be played to them may be a good feature. Detecting events and the number of their occurrence may provide essential information about what parts of the agent have to be worked on (i.e. if passes often fail there is some space for improvements). Those events are: pass success/fails, ball dribbling, lost balls, goal shot (success, out, intercepted) and kick out. By detecting event sequences more information can be extracted such as lost balls after dribbling.

3.2 Organizers' Required Features

The running of simulation league in comparison to other RoboCup fields, is so quiet and therefore it doesn't have enough visitors. So with showing a better illustration of games and final goal of RoboCup, this league could be more interesting for visitors. The publicity that the RoboCup events get from the media provides an ideal opportunity to show the state of art of these systems during RoboCup World Cup.

4 Implementation

When talking about a logplayer or a monitor this essentially means the same program in two different operation modes, logplayer means that the program replays logfiles of a previously simulated game, monitor that a game that is currently simulated is displayed.

4.1 Features

Some features of TA2006 that can be useful in developing agents are:

Forward and backward replay in different speeds even in live games is achieved by storing data into a new data structure. Playback is much faster now, it has to be slowed down artificially leading to the opportunity of different playback speeds.

The tool is also helpful for probing the agents' internals. The Layered Disclosure concept[3], first introduced in CMUnited99 simulated agents, and it proved to be very helpful in development of intelligent agents. The goal of the Layered Disclosure concept is to make various agent characteristics.

Observable, without overwhelming the human observer with data. Realizing this goal needs the agent to store a log of all relevant information from its internal state, world model, and reasoning process. Then a tool must be used to synchronize the log with a recording of the observable world and it must provide an interface to allow the developer/observer to probe a given agent's internal reasoning at any time and any level of detail. Our tool provides all the necessary functionality. The times according to the various agent outputs is also parsed at startup. When displaying, the fitting record of the agent logfile is found using the time currently displayed by the monitor. Some none-textual logging facility comes handy for the human developer/observer; the tool understands a simple notation for describing 3D or 2D geometric shapes. For example an agent may write in its log file:

```
<0><49> <Behavior> (Command) Sphere ball
( 0.023f, 0.09f, 0.111f, 0.130f, 0.5f, 0.5f, 0.5f, 0.3f );
```

And when the log-player is showing the cycle 49, the developer can select the agent, and the log- player draws a sphere with radius 0.130 at point (0.023, 0.09, 0.111). (fig. 1).

Implemented commands are: Drawing circles, spheres, cubes, lines, playing sounds, Showing a text on the screen and many more. In case of filtering the logfile output the developer just have to change the log level. New camera view modes were implemented, most of them are adjustable in a way (camera height,

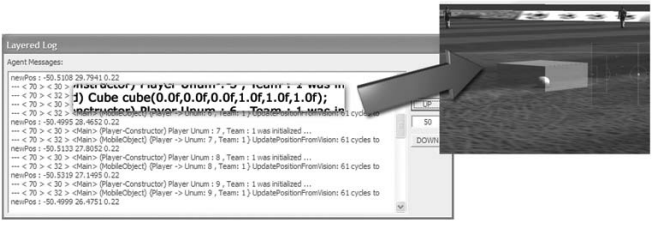

Fig. 1. Draw shapes based on output of an agent, for example agent can command the monitor to render its internal world model on the screen

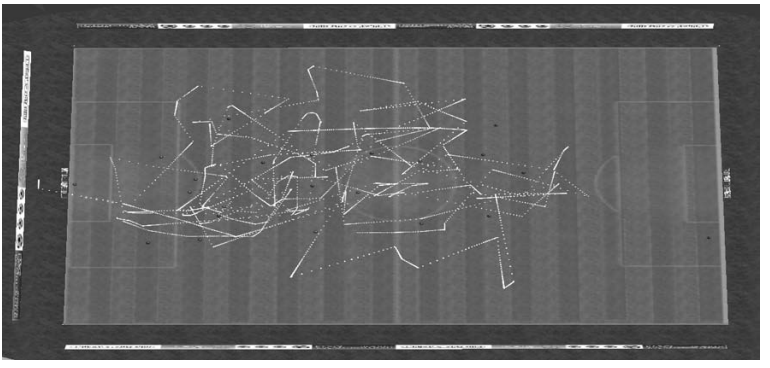

Fig. 2. Ball trajectory

Fig. 3. Robots vs. Humans, Different models can be used to represent the agents in the visualization

distance or point to look at). Some camera types like tele, top, tower views are implemented. Also cameras can reside directly over the agent of interest.

Displaying the ball's and player's movements for some time forward is solved as lines in the color of the team or white for the ball. The length of the line (how far into the future movements are shown) can be adjusted. The agents track (movements throughout the whole game) can also be displayed, this track can be colored according to the agents movement speed (fig. 2).

In case of presentation, Team Assistant 2006 provides rich game-viewing controls, plus some eye- candy features for on-site demonstration of a game, including:

- Commentator
- Automatic replays
- Customizable 3D agent model (fig. 3)

- Complete control over camera position and direction
- Several preset cameras (Tele, Top, Tower, ...)
- The ability to display different camera views side-by-side
- Visualizing Catch and Kick commands of agents

When the tool is in trainer mode, it allows defining training sessions or scenarios. A scenario definition contains a starting state and/or an end condition. Those can be specified using Angel Scripts. The trainer mode also needs a server counterpart.

4.2 Additional Features

TA2006's main power lies in its ability to be extended using *AngelScript* plug-ins. To get a glimpse of what can be done within a plug-in, it's good to mention that in current release the Commentator itself is a plug-in, all sound effects are provided by a plug-in, some training/test sessions are wrote using plug-ins, the game statistics are both calculated and rendered on screen by a plug-in. In general, plug-ins can Obtain:

- Locations of all objects
- State of the match (play mode, time,...)
- Player actions (requires new server to monitor protocol)
- Some processed values (ball and agents' speed)

And Can Perform:

- Move agents and ball
- Change play mode of the match
- Control the log player (change playback speed, jump to a specific cycle,...)
- Draw shapes in the field
- Draw markers on the field
- Write/Draw on the screen
- Control the camera
- Play audio file

4.3 Other Features Derived from the Implementation

The new data structure of the program allows the user to use single step mode, for- and backward even when watching the game "live". The server is continuing the simulation in background, even if the monitor is in single step mode. Functions that are independent from agent logfiles, such as movement display, can be shown in this mode.

5 Results

The logfile output is useful to understand the agents behavior. This program has replaced the lite monitor that comes with the simulation server. The agent

may communicate its decisions to the developer. (i.e. which role it plays, which behavior of the role it has chosen and what action it selected). In occurrence of an error in the ball movement prediction the agent was changed to display its data about the ball (its position and movement vector according to the agents world model). This way the error became visible. Without displaying that data, localizing the error would have taken much longer. When creating a behavior that covers an opponent, it is mostly wanted that only one player covers an opponent at a time. To verify the opponent selection the agent can simply draw a line from himself to the opponent it intends to cover or include the chosen position into its drawings. When displaying the draw commands of all agents it becomes visible if something is wrong with the selection or the position the agent has chosen to move.

Fig. 4. Drawing offside line and automatic replay

The ability of reverse playback of the game gives the opportunity of analyzing the same scene again without watching the game from the beginning. Also slight transitions in action selection may be analyzed without much interference. This feature in addition to the display of agentlogs (in both ways, text and drawings) gives the opportunity to understand the agents decisions according to its own world model, and not only the real simulated world displayed in the monitor. The ability of delayed playback and direct analyzing of a currently simulated game provides a real speedup in development in comparison to other tools. Simulating a 3D game may take up to 10 or 20 min which is a long time the developer has to wait in order to for example analyze the previously mentioned ball approach in slow motion. With the delayed playback that situation may be analyzed while the game is still simulated in background. After the analyzing was finished the game may be watched time shifted in normal speed or the replay may be fastened to reach the most recent scene. The track display in the 3D scene along with the plotting opportunities were giving the developer essential information about the agents or ball movements throughout the game. This way the developer can

find agents which are not moving much or fast, which may point to a suboptimal formation or behavior. By viewing the ball movement plot (fig. 2) of the game between Aria (left) and ZJubase (right) it is obvious that the ball was mostly located on the left side of the field and though that ZJubase dominated the game. ZJubase won this match 1:0.

6 Conclusion

The logplayer can be a useful tool when trying to resolve strange behavior of an agent or verifying the proper work of the agent. The SBCe team resolved some problems within their code using the logplayer. These resolved problems are namely the examples given in section 5. The resolving of these problems would have taken longer without the possibility of drawing or text output. Due to the parsing at the beginning the startup of the logplayer takes longer in comparison to the lite monitor/ logplayer, but the playback is much faster and the monitor does not have to be restarted to watch the game again.

Finally this tool is intended to be a general-purpose, highly customizable package. We think it has the potential to be used as the primary analyzer, visualizator, and logviewer for anyone interested in developing agents for the RoboCup 3D Soccer Simulator. TA2006 won the second place in RoboCup 3D development competition 2006.

References

1. Planthaber, S., Visser, U.: Logfile player and analyzer for RoboCup 3D simulation. In: Lakemeyer, G., Sklar, E., Sorrenti, D.G., Takahashi, T. (eds.) RoboCup 2006: Robot Soccer World Cup X. LNCS (LNAI), vol. 4434. Springer, Heidelberg (2007)
2. Nazemi, E., Kazemi, V., Habibi Shahri, A., Hosseingholizadehm, A., Nooraei B., B.: SBCE SmartSpheres 3D Development Team Description. In: Proc. of RoboCup 2006 (2006)
3. Riley, P., Stone, P., Veloso, M.: Layered Disclosure: Revealing Agents' Internals. In: Castelfranchi, C., Lespérance, Y. (eds.) ATAL 2000. LNCS (LNAI), vol. 1986. Springer, Heidelberg (2001)
4. Nazemi, E., Zareian, A., Samimi, R., Shiva, F.A.: Team assistant team description paper. In: Proc. of Robocup 2002 (2002)
5. Raines, T., Tambe, M., Marsella, S.: Automated assistants to aid humans in understanding team behaviors. In: Veloso, M.M., Pagello, E., Kitano, H. (eds.) RoboCup 1999. LNCS (LNAI), vol. 1856, pp. 85–104. Springer, Heidelberg (2000)
6. Aghaeepour, N., Bastani, M., Miri, F.D., Masoudnia, S., Radpour, S., Fotuhi, S.A.: UTUtd2006-3D Team Description Paper. In: Proc. of RoboCup 2006 (2006)
7. Obst, O.: Simulation league - league summary. In: Kaminka, G.A., Lima, P.U., Rojas, R. (eds.) RoboCup 2002. LNCS (LNAI), vol. 2752, pp. 443–452. Springer, Heidelberg (2003)
8. Takahashi, T.: Logmonitor: From player's action analysis to collaboration analysis and advice on formation. In: Veloso, M.M., Pagello, E., Kitano, H. (eds.) RoboCup 1999. LNCS (LNAI), vol. 1856, pp. 103–113. Springer, Heidelberg (2000)