Autonomous Information Indication System

Atsushi Shinjoh[1 2] and Shigeki Yoshida[1]

[1] IAMAS
3-95, Ryoke-cho, Ogaki-city,
Gifu 503-0014. Japan.
[2] Faculty of Engineering, Gifu Univ.
1-1, Yanagido, Gifu-city,
Gifu 501-1193. Japan.

{kaminari,shige}@iamas.ac.jp

Abstract. We developed an Autonomous Information Indication System for the RoboCup simulation league. This delivers and displays a three-dimensional view of the game to an audience using low-speed networks such as the Internet. Moreover, the audience has the ability to select a favorite shot from four different ones that are positioned on the field. Recenltly, our system performed succesfully at the RoboCup Japan Open 99. This paper outlines the feasibility and effectiveness of our system based on our evaluation of various experiments.

1 Introduction

The research into an infrastructure for the RoboCup simulation league has developed rapidly. Its initial motivation was to create a system that can indicate the situation of the simulation league more attractively. This research can be divided into two kinds. The first is research into a commentator system that appreciates a situation of a soccer game and makes comments similar to a on the spot broadcast in real-time[4][5]. The MIKE, developed by ETL team and demonstrated at the RoboCup-98[6] is a typical system of this type. The other is research into a three-dimensional viewer system that describes three-dimensional situations of a soccer game in real-time. Our proposal was to develop the system, actually developing two kinds of three-dimensional viewer systems, and demonstrated these systems at the RoboCup-97 and the RoboCup-98[1][2]. Several three-dimensional viewers such as RoboMon and Virtual RoboCup were also developed after our initial demonstrations[9][7].

The creation of these systems, the first stage of infrastructure domain, was demonstrated at the RoboCup-98. Each system was able to provide a vivid picture of the game situation. The infrastructure domain must be challenge to go to the next stage. The goals for this next stage are the followings:

- Using several techniques, the system can indicate the information (scene data) in several environments in real-time.
- The system can cope with the various choices of an audience in real-time.

M. Veloso, E. Pagello, and H. Kitano (Eds.): RoboCup-99, LNAI 1856, pp. 469–480, 2000.

- The system can indicate the most suitable scene data based on a prediction of the indicated situations and an evaluation of scene data.

These factors turn the infrastructure system into an information indication system for a digital broadcasting environment. An audience can watch a particular situation of a game anyplace that is connected to computer network. Furthermore, an audience does not have to control the indication system while using it. Based on these architectures, this system will be the ideal information environment for an audience.

We already started to develop an autonomous information indication system (AIIS) as a support system for personal information environments. The AIIS transmits and displays the information to each person using those methods that it has predicted as the most effective. Based on this, we developed an AIIS for the RoboCup simulation league that possessed the above features. This development has also been guided by our evaluation and analysis of our two three-dimensional viewer system. As first, we mounted two mechanisms. One was a scene data delivery mechanism that supports data delivery using low-speed networks such as the Internet. This mechanism was able to cope autonomously with rapid changes in the data delivery rate. The other was a view selection mechanism. An audience can select a favorite shot from four different cameras, including zoom. After selection, the system can indicate the scene data to several audiences. We experimented with this system at the RoboCup Japan Open 99 that held in May.

In this paper, we describe the basic concept of AIIS, and discuss its potential.

2 Autonomous Information Indication System (AIIS)

As noted above, we developed the AIIS for the RoboCup simulation league. Figure 1 shows its system architecture. The Following is a description of each part of the system.

Communication mechanism: This mechanism contains two mechanisms, a scene data deliverer and a network's traffic detector. This unit measures the immediate amount of network traffic, and continually informs the "scene selection mechanism" of its condition.

There are several methods for measuring the amount of network traffic. After evaluating the results of previous experiments, we selected one of these at the RoboCup Japan Open 99. A detailed description of each method is described in the next chapter.

Scene selection mechanism: This selects scene data for delivery based on an evaluation. First, the mechanism receives a scene request from each client system, checks the "evaluation value" added and sent by a "game condition mechanism", and receives the current scene delivery rate which has been determined by a "communication mechanism". Second, this mechanism selects the scene data based on an analysis of it.

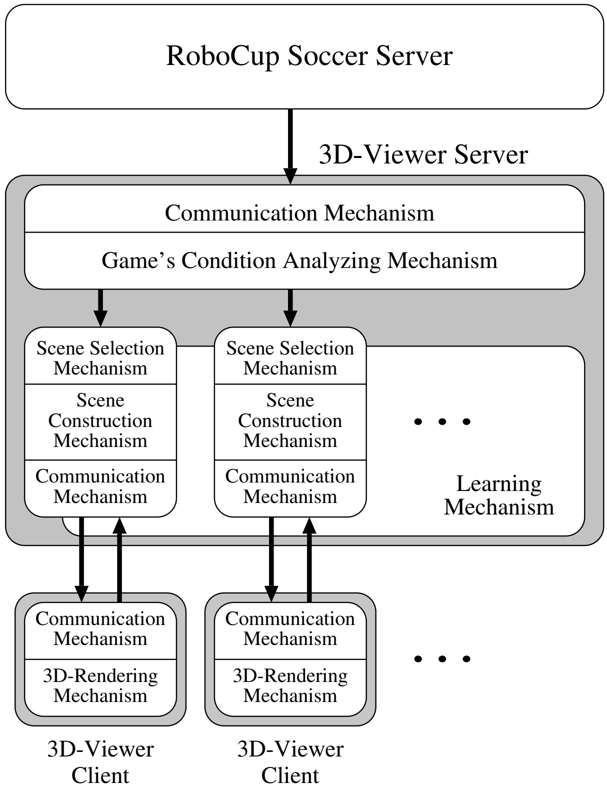

Fig. 1. The AIIS for the RoboCup.

Game condition analyzing mechanism: This mechanism evaluates the conditions of the game based on the situation of the camera and each player's positions, the formation, and the score, and adds an evaluation value to each scene. The evaluation value shows each scene's value.

3D-rendering mechanism: This renders a three-dimensional scene on the screen. This is part of a 3D-viewer client. A 3D-viewer client is composed of this and the communication mechanism. Our AIIS for the RoboCup simulation league has two 3D-viewer clients. Each client uses a different 3D-rendering mechanism. Though its detailed descriptions of a three-dimensional world are very effective in showing the situation, it needs a great deal of rendering power. As a result, this mechanism limits the number of running environments.

To cope with numerous running environments, the system adopts the two 3D-rendering mechanisms, and offers two 3D-viewer clients that use each of the mechanisms. One is a VEGA version (Figure 2). VEGA is one of the major applications of virtual reality systems for real-time processes, and supports an

Fig. 2. 3D-viewer client: VEGA version.

OpenGL library under Irix and WindowsNT environments. This version needs a VEGA runtime application. The representations of this system are quite detailed, and each player's movements are very smooth. However, this system needs a high-end graphic accelerator board with a high-performance CPU.

The other 3D-rendering mechanism is DirectX (Figure 3). This version runs under Windows95, 98, NT, and needs a DirectX library only[3]. Its description is very simple, and it can run smoothly using a mid range personal computer. This client system has six buttons for the audience to use when selecting camera positions.

3 Experimentation

3.1 Prototype 1

We developed a prototype of the AIIS for the RoboCup simulation league that has a delivery mechanism that selects the most suitable scene data that is sensitive to the network traffic. This system has the following two functions:

1. The system detects the network traffic autonomously, and controls the amount of scene data transmission corresponding to the network traffic.
2. The system selects the most suitable scene for the extraction of the "delivery scene data" in real-time.

In terms of the first function, the communication condition is judged automatically, and the scene data is selected and delivered at the appropriate rate to

[3] This library can be downloaded from Microsoft Web site

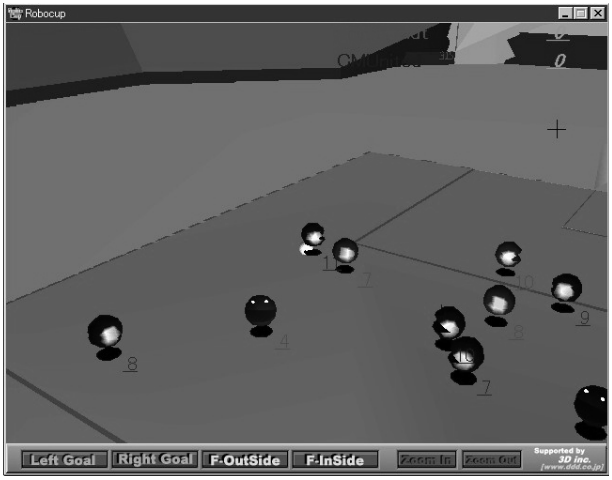

Fig. 3. 3D-viewer client: DirectX version.

the three-dimensional viewer client. This function corresponds to the communication mechanism and the scene selection mechanism in Figure 1.

As for the second function, the scene data from soccer server is analyzed. If additional scene data is required to display a game situation, this data is also selected. This function corresponds to the game condition analyzing mechanism and scene selection mechanism in Figure 1.

A basic mechanism of a traffic control At this time, we have adopted three kinds of methods that may be more effective for network control from many methods that we would have selected.

These methods are based on the following fundamental mechanism. (Figure 4)

1. When scene data is received from the soccer server, the three-dimensional viewer server adds a sequential number to each scene to distinguish each unit of scene data, and delivers this data to each client.
2. When each client receives the scene data, it makes an acknowledgment (ACK) which contains the scene data's sequential number, and returns it to the server. In one method, the client calculates the scene delivery rate, and includes it in the ACK.
3. The server or the client decides the scene delivery rate based on the return rate of the ACK or the receiving rate of the scene data.
4. The server decides whether a scene should be delivered or not, in accordance with the determination of delivery rate. Then, only after the delivery of the current scene has been decided on, does the server send scene data to the client.

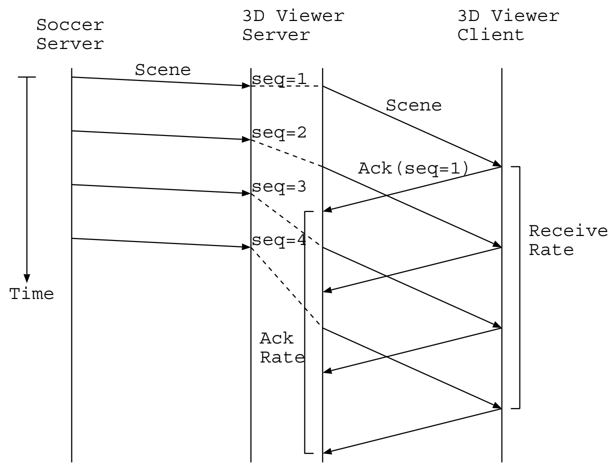

Fig. 4. The basic mechanism of traffic control.

Thus, there are two kinds of methods for deciding the scene delivery rate. The server uses one, and the client uses the other. In this prototype system, "type 1" and "type 2" adopt a method when the server decides a scene delivery rate, and "type 3" adopts a method when the client decides a scene delivery rate.

Type 1 In this method, the three-dimensional viewer server sends one scene data to the client(Please check Figure 4.). Then, the server sends a new scene data after an ACK of the scene is returned from the client.

This method is limited in that the server can't deliver the next scene before the server receives the ACK from the client. Because this situation leads to an interruption of delivery, a server sends scene data when the interval time is timed-out.

Type 2 In this method, the server calculates the delivery rate based on the rate of the ACK that is sent back from the client. The average ACK return rate is measured per "unit time". This measurement method is calculated in the following way:

- The server sends the all scene data during the first "unit time".
- When unit time has expired, the server calculates the average ACK return rate. Based on this, the server then calculates the delivery rate of the scene data for the next "unit time".
- The server delivers the scene data at a calculated delivery rate before this "unit time" has expired.

  The same procedure is then repeated.

Type 3 In this method, the client calculates the received scene data as the received scene dates rate, and then sends this data rate to the server. Based on this rate, the server determines the delivery rate.

This method adopts the two kinds of measurement methods also utilized by "type 2"

The suitable scene selection mechanism The scene data to deliver is only selected with a definite interval depends on the delivery rate, when the traffic control mechanism is used.

To select and deliver the most suitable scene data, we implemented the following mechanism into the three-dimensional viewer server.

- The server checks the scene data that was received from the soccer server. It then determines what the next scene data should be delivered when game conditions change a free kick, a goal, etc.
- The server delivers both scene data selected by the suitable scene selection mechanism and scene data selected by the traffic control mechanism.

The server judges the change in the game conditions using the "play mode flag" of the scene contained in the scene data.

3.2 The result of experimentation with the "Prototype 1"

The implementation and the method of measurement The prototype system was implemented by using a three-dimensional viewer server for UNIX, and the two-dimensional monitor that complies with the three-dimensional viewer server for UNIX and MS-Windows.

The followings are test environments.

- Substituting a logplayer for UNIX for a soccer server.
- A three-dimensional viewer server for UNIX.
- One two-dimensional monitor that complies with a three-dimensional viewer server for MS-Windows.
- A PPP connection circuit via a modem connected at 28.8 Kbps.
- A log data of the final game of the RoboCup-98 simulator league.

The measurement of the non-implemented viewer For comparison sake, we studied the case where above mentioned traffic control mechanism and a suitable scene selection mechanism were not mounted. The results of our measurements are as follows. (Table 1)

- One ACK was returned on an average of 3 - 4 scenes (0.3 - 0.4 seconds).
- After the client received 20 scenes, further scene data failed to reach to the client, and ACK wasn't returned.

Table 1. The behavior of the non-implemented viewer. (rate=scenes/sec.)

sec.	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0
rate	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10
sent	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
ACK										1				2				3		

Table 2. The behavior of "type 1". (mode: k=free kick, p=plan on)

sec.	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.1	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0
rate	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
mode	k	p	p	p	p	p	p	p	p	p	p	p	p	p	p	p	p	p	p	p
sent	1	2							9				13				17			
ACK									1				2				9			

Type 1 When the method of "type 1" is implemented into a server client system, then the resulting measurements occurred. (Table 2)

- The system approaches stability, and works.
- The RTT (Round Trip Time) of one scene data is about 0.8 - 0.9 seconds (about 8 - 9 scenes).

When game condition was unchanged due to a longer delivery time of the scene data and ACK rate, compared with real possible rate (calculated by hand, so only potentially), the calculated average scene delivery rate is about 1/2.

However, when the game condition was changed, the scene data is still delivered to the client by the suitable scene selection mechanism even when ACK wasn't return to the server. Because of this behavior, the scene delivery rate increases and the game is displayed more smoothly.

Type 2 When the method of "type 2" is implemented, the following measurements result (Table 3):

1. When the unit time is 2 seconds (20 scenes), the ACK comes to a halt after about 200 seconds.
2. When the unit time is 1 second (10 scenes), the system becomes virtually stable and works.

The situation where "unit time is 2 seconds" was very similar to the situation: "non-implementation". Based on a calculation of the delivery rate of the scene data,

Table 3. The behavior of "type 2". ("unit time" = 1 sec.)

sec.	66.1	66.2	66.3	66.4	66.5	66.6	66.7	66.8	66.9	67.0	67.1	67.2	67.3	67.4	67.5	67.6	...
rate	2	2	2	2	2	2	2	2	2	3	3	3	3	3	3	3	...
mode	k	k	k	k	k	k	k	k	k	p	p	p	p	p	p	p	...
sent			663					668		670			673			676	...
ACK	600			603				606				609					...

the server changed the delivery rate after 2 seconds. However this subtle difference is accumulated at the clients step by step, until the client is unable to transmit the ACK to the server.

In this measurement, when the "unit time is 1 second", the situation becomes stable, because this unit time is quite effective for thie environment. However, when the measurement environment changed, and the speed of the connection circuit slowed down, the system had to choose a shorter unit time to deal with this environment.

Therefore, to work the system at every circuit speed, it is necessary to mount a mechanism that can find the proper delivery rate based on the measurement of the ACK information. This must be done before the system cannot handle the situation.

Type 3 In most cases, it showed the same result as "type 2".

However, when the server did not receive an ACK, the server delivered the scene data on the basis of the received scene data rate calculated by the client. Therefore, the server would not cope with changes in situation.

3.3 Prototype 2

The implementation Based on a result of the study of the Prototype 1, we developed a prototype of a system where a three-dimensional viewer client can send a request about the form of a display.

On the client side, the following items are available for selecting the position and movement of the camera. Once a selection is mode, a request is then sent to the server.

- The movements of each camera:
  - A camera positioned behind the goal of "team A".
  - A camera positioned behind the goal of "team B".
  - A camera that is moved along the side lines.
  - A camera that is moved freely around the ground.
- The motion of the camera:
  - Zooming in.
  - Zooming out.

A server has two modes to select and deliver the scene data. One mode selects a scene individually in response to each request from the clients, creating and sending the data of each scene to the clients. The other mode selects one scene as a result of a majority decision, however also creating and sending the data of the scene to the clients.

The mechanisms In this prototype system, the following methods are adopted for each mechanism in Figure 1.

A mechanism for analyzing a game condition: Any change of the game's condition is decided by the play mode flag of the scene.

A learning mechanism: Requests for a particular position or a movement of a camera concerned in response to the position of the ball or a change in the game are recorded for each client.

A scene selection mechanism: The method of prototype 1 is used for a decision about whether current scene data should be sent or not.

The decision to select a camera position and movement is made on a request from each client, and trends perceived by the learning mechanism.

The decision about which mode will be used to select and to deliver scene data is determined from out side of the system by an operator.

A mechanism for scene construction: Basically, individual scene data is created for each client. However, when a mode to select one scene for all clients is chosen, only one scene is created.

A communication mechanism: The function that adds a request from a client to the ACK data and sends it to a server is included in the mechanism implemented in the prototype 1.

3D-rendering mechanism: This mechanism rendered each scene using OpenGL or DirectX graphics library.

3.4 The result of experimentation with the "Prototype 2"

We performed an experiment of the delivery capabilities of prototype 2 using the Internet. This was a public experiment, open to everyone, was held at May 2 - 3, 1999 as a part of the RoboCup Japan Open 99. This experiment was not a replay of past simulation leagues, but an actual broadcasting of games in real-time. This experiment occurred in the following environment:

- Delivery data: The final and semi-final game of the RoboCup Japan Open 99 simulator league.
- The number of 3D-viewer clients: 10 - 20.
- The number of client machine: 10 - 20.

- Connection speed: 1.5Mbps (From conference center to the Internet)
- Operating system of clients: Not determined (MS-Windows95, 98, or NT)

Because there was a long holiday in Japan during this period, the maximum number of client accessing at same time was 20. However, in this experiment, each number of the audience could connect to a communication mechanism in low speed network environment such as 28.8Kbps. Moreover, some of there members were able to use the scene selection mechanism such as switching and zooming of cameras.

The AIIS was succeesful in managing each connection, and delivered scene data according to each client's request. Based on an analysis of this experiment, we decided to expand the "communication mechanism" and "scene selection mechanism", as well as the architecture that supports very low speed network connection such as 9.6Kbps. As you may know, this is the connection speed of mobile telephones in Japan.

4 Conclusion

In this paper, we described the basic concept of the AIIS for the RoboCup simulation league. We also showed its effectiveness and feasibility through an evaluation of the results of several experiments. Through our evaluations, we learned what areas need to be improved. We also decided to develop an AIIS for the RoboCup-rescue. The purpose of this system is to create an ideal wearable information environment for disaster rescue. Later, we will describe our developing concept of "Wearability Design", a concept of an ideal individual information environments[3].

Acknowledgement

We would like to thank Internet Initiative Japan Inc., Intergraph Corp. Japan, and 3D Inc. for their generous support. We would also like to thank many individuals who helped us. This research is one of the projects of International Academy of Media Arts and Sciences (IAMAS), which is supported by the Gifu prefecture.

References

1. A. Shinjoh: "RoboCup-3D: The construction of Intelligent Navigation System", RoboCup-97: The First Robot World Cup Soccer Games and Conferences. 179–190, Springer (1998).
2. A. Shinjoh, S. Yoshida: "The intelligent three-dimensional viewer system for RoboCup", In *proc. of the second RoboCup Workshop*, 37–46 (1998).
3. A. Shinjoh, S. Yoshida: "A development of Autonomous Information Indication System for RoboCup Simulation League", In *proc. of The 1999 IEEE Systems, Man, and Cybernetics Conference (SMC'99)*, 1999 (This will be held at October 1999).

4. H. Matsubara, I. Frank, K. Tanaka, I. Noda, H. Nakashima, K. Hasida: "Automatic Soccer Commentary and RoboCup", In *proc. of the second RoboCup Workshop*, 7–22 (1998).
5. K. Binsted: "Character Design for Soccer Commentary", In *proc. of the second RoboCup Workshop*, 23–36 (1998).
6. K. Tanaka, I. Noda, I. Frank, H. Nakashima, K. Hasida, H. Matsubara: "MIKE: An automatic commentary system for Soccer", In *proc. of Int. Conf. on Multi-agent Systems 98*, 285–292 (1998).
7. RoboMon: http://medialab.di.unipi.it/Project/Robocup/index-en.html
8. S. Yoshida, A. Shinjoh: "The prototype system of the suitable network traffic mechanism for the three-dimensional viewer for RoboCup", In *The proc. of PRICAI workshop of RoboCup*, 134–144 (1998).
9. Virtual RoboCup: http://www.TechFak.Uni-Bielefeld.DE/techfak/ags/wbski/3Drobocup/3Drobocup.html