INFORMATION
SCIENCE
TECHNICAL
REPORT

NAIST-IS-TR97010
ISSN 0919-9527

Scalable Networked Virtual Reality System with Live Video

Seiwoong OH, Daisuke KADO,
Tomohiro TAIRA, Kazutoshi FUJIKAWA,
Shinji SHIMOJO, Masatoshi ARIKAWA,
Hideo MIYAHARA

June 1997

NAIST

〒 630-01
奈良県生駒市高山町 8916-5
奈良先端科学技術大学院大学
情報科学研究科

Graduate School of Information Science
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara 630-01, Japan

Scalable Networked Virtual Reality System with Live Video

Seiwoong OH†, Daisuke KADO‡, Tomohiro TAIRA†, Kazutoshi FUJIKAWA‡,
Shinji SHIMOJO††, Masatoshi ARIKAWA*, Hideo MIYAHARA†

†*Department of Informatics and Mathematical Science*
Graduate School of Engineering Science, Osaka University
Toyonaka, Osaka 560, Japan

‡*Graduate School of Information Science*
Nara Institute of Science and Technology
Ikoma, Nara 630-01, Japan

††*Computation Center, Osaka University*
Suita, Osaka 567, Japan

* *Faculty of Information Sciences, Hiroshima City University*
Hiroshima, Hiroshima 731-31, Japan

Abstract

Currently, most of networked virtual reality systems do not consider the scalability of the system in terms of the network. They assume high speed network among rather small number of distributed users. However, as high performance graphic workstations and PCs become inexpensive, a large-scale networked virtual reality and use of a wide area network with the limited performance are expected. In this paper, we propose a scalable networked virtual reality system with a new dynamic quality control mechanism. We introduce the notion of the *importance of presence(IoP)*, to represent the degree of the importance of objects. The proposed system can tolerate changes of both network and end system resources by reducing the quality of less important objects dynamically based on their IoP, and keep total quality of a virtual space as much as possible from point of view of user's perception. Unlike conventional virtual reality systems, our enhanced virtual reality system retrieves objects in a virtual space incrementally from a server as they are required and also integrates not only real-time videos stored in the server but also live videos into a virtual space seamlessly. We have been implementing a prototype system on SGI indigo2 workstation. We achieved good result of the waiting time on the prototype system with the proposed quality control mechanism.

Keywords

VRML, Distributed interactive simulation(DIS), Level of detail(LoD), Importance of presence(IoP)

1 Introduction

Soon after the invention of virtual reality where one can navigate computer generated 3-dimensional virtual spaces, people on computer networks can share virtual spaces easily by exchanging their descriptions of virtual spaces, which are described in Virtual Reality Modeling Language(VRML)[1, 4]. We call this kind of application as a networked

virtual reality (NVR) system. In such an environment, when one just clicks a mouse button in a virtual space, a file which contains the description of another virtual space is transferred from an server to his client and he can go into another virtual space. VRML provides a basic mechanism for sharing and exchanging virtual spaces among distributed users in the network.

Although VRML provides a mechanism to exchange virtual spaces among users, it hardly provides a way to exchange live information among distributed users. We may consider two kinds of interactive information. One is represented by a motion or a change of a computer generated 3-dimensional object(3D object) such as an avatar in the virtual shared space or a combat air plane in the distributed interactive simulation(DIS)[5]. By exchanging this kind of interactive information, distributed users can share awareness of other persons' behavior as well as a virtual space. Another interactive information is given by a live video image from a remote site. A live video such as a face image of an attendee in the virtual meeting room or an actual outside scenery seen in the virtual window[2] increases user-friendliness and enhances interactivity of the NVR system.

Currently, most of NVR systems do not consider the scalability of the system in terms of the network. They assume high speed network among rather small number of distributed users. However, as high performance graphic workstations and PCs become inexpensive, a large-scale NVR and use of a wide area network with the limited performance are expected. Therefore, introduction of the scalability to the NVR system in terms of the network performance becomes attractive and important.

Because without the scalability of the NVR system, the system and network resources may starve during the execution of a NVR application, the quality of a NVR application such as rendering speed and interactivity to users is damaged. Even if the network and the end system could preserve a certain amount of resources, the required resources change dynamically during navigation of a large virtual world. In case of live video, a server sends a live video to the client as a stream of video frames through the network and the client renders them as images in the virtual space. In this case, the quality of the video is also greatly affected by the currently available network and end system resources.

In addition to such a scalability problem, another problem in the NVR is the scalability of the object transfer. Although most DIS systems take care of exchanging object's behavior, they don't take care of exchanging object description itself. They assume all the descriptions of objects are pre-loaded.

In case of VRML, although one can change a virtual space by just clicking, he must have a long wait until an entire file is retrieved and a new virtual space is constructed completely on his local client. As a result, file transfers interrupt a navigation of a large virtual world which consists of several VRML files whenever one changes a virtual space. Therefore, the quality of navigation is greatly decreased by the time for the transfer of a VRML file and the construction of the corresponding new virtual space. Although transferring an entire file before navigation can avoid the influence of limited performance of best effort network like the Internet, it makes a large virtual space intractable and loses interactivity to the user in navigation. Therefore, there needs a mechanism by which the system can adapt to changes in both network and system performances. One way to tackle this problem is to control presentation quality based on the available end system and network resources.

In this paper, we propose a scalable NVR system with a new dynamic quality control mechanism. Our approach is firstly, introducing the notion of the *importance of presence (IoP)*, secondly, choosing the object which has the least IoP, and finally, deteriorating the

quality of the least important object to adapt to the available end system and network resources. IoP is determined by the distance from the user, the angle from his eye vector and the size displayed on the computer screen.

Once we choose the object of which quality is reduced by the system, that object is scaled. For a 3D object, there is a notion of scalable object called *level of detail(LoD)* in VRML version 2.0. A user can define different level of the quality for each object in a virtual space by using LoD of which each level consumes different degree of end system and network resources. Our virtual reality system choose appropriate LoD of each object based on its IoP. For a video object, we can change its quality by choosing temporal resolution and spatial resolution. In order to resolve the scalability of object transfer, streaming mechanism of a VRML description, which sends the objects constructing a virtual space incrementally from the server is introduced.

The proposed system can tolerate changes of both network and end system resources by reducing the quality of less important objects dynamically, and keep total quality of a virtual space as much as possible from point of view of user's perception. Unlike conventional virtual reality systems, our enhanced virtual reality system retrieves objects in a virtual space incrementally from a server as they are required and also integrates not only real-time videos stored in the server but also live videos into a virtual space seamlessly. We have been implementing a prototype system on SGI indigo2 workstation using Performer[7].

Other related works focusing on virtual reality systems[1, 2, 6] proposed QoS control models based on LoD which consider only the distance between objects and user's position. Moreover, their models don't take account of smooth navigation among several virtual space of which a large virtual world consists. Even though RING system[3] proposed a prediction mechanism for navigating a virtual space, it can handle only stand-alone virtual reality systems. In DIS, many researches are going on for the scalability of the NVR system in terms of live 3D objects. However, there is few research about the scalability of the NVR system in terms of the live video images.

This paper is organized as follows. Section 2 discusses an overview of a NVR system. In section 3, we mention the integration methodology of live videos into virtual reality, and then in section 4, show the object transfer mechanism. The implementation is presented in section 5. Finally, section 6 concludes this paper.

2 Overview of a Networked Virtual Reality System

A networked virtual reality system consists of severs which store descriptions of virtual spaces and clients which retrieve and render the virtual spaces based on the descriptions(see Figure 1). A description represents a collection of 3D objects with their geometrical information in the virtual space. Such a description is called a *scene graph*[1]. In our proposed VRML system, real-time and live videos are introduced into a virtual space. Therefore, a motion video is handled as an object and also included in a scene graph.

Our purpose is to integrate real-time and live videos into virtual spaces seamlessly and to provide fine interactivity for a user who navigates a large virtual world.

3 Integration of Live Video into Networked Virtual Reality

In NVR systems using VRML, each 3D element is treated as an object called *3D object*. In our virtual reality system, motion videos including live videos are also handled as objects called *video objects*.

It is desired to avoid the drastic decrease of the total quality of the presentation by deteriorating the quality of each object ran-

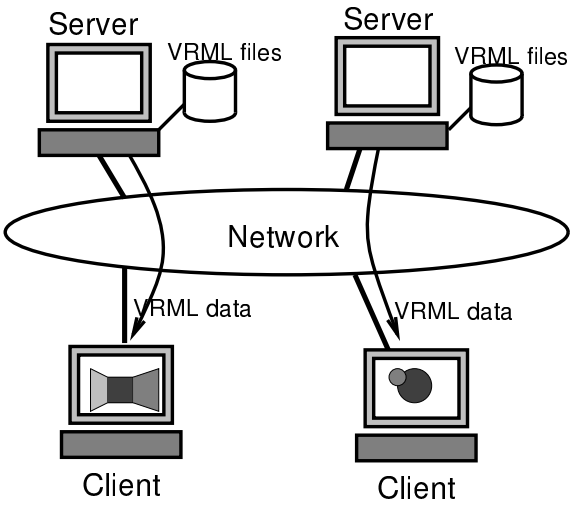

Figure 1: Overview of a networked virtual reality system

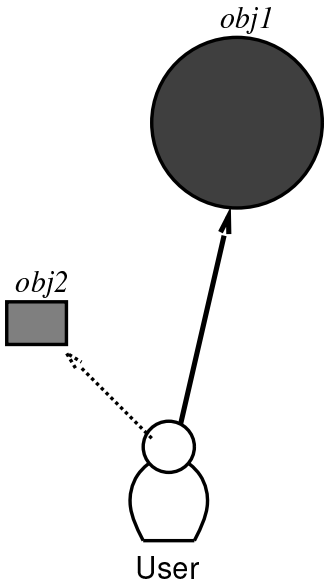

Figure 2: User's perception

domly even though network and end system resources are going to starve. Therefore, at first we introduce the notion of *importance of presence(IoP)* of objects in a virtual space, which is calculated based on the user's perception, while the conventional virtual reality systems take account of only the distance from the user. Secondly, our NVR system chooses the object which has the least IoP, and finally, deteriorates the quality of the least important object to adapt to the available end system and network resources.

3.1 Importance of Presence

Reducing the quality of objects which have lower IoP value can lighten the load of network and end systems. Objects which have a lower IoP value are likely to have a lower quality in the presentation of a virtual space, while ones which have a higher IoP value can keep a higher LoD when the network or the end system is over-loaded.

Now, we describe how to calculate the IoP value for both 3D objects and video objects. If an object is far from the user's position or away from the direction of his eye vector, he may have little awareness of it. As shown in Figure 2, it seems that a user is likely to perceive an object(obj_1) which is further from the user but is larger in size more than one(obj_2) which is closer to the user but is smaller. Therefore, in our NVR system, the IoP value of each object is calculated by the distance from the user, the angle from the user's eye vector, and the size of the object.

Figure 3 shows the positional relation between a user and an object. Let l, θ, and S be the distance between the user and the object, the angle of the object from the user's eye vector, and the actual size of the object displayed on the computer screen, respectively. Then IoP value of the object can be calculated as a function of l, θ, and S:

$$I = f(l, S, \theta)$$

where I is the IoP value before normalization. Assume that there are N objects in a virtual space, and let IoP_i be the normalized IoP value of ith object ($i = 1 \ldots N$).

$$I_i = f(l_i, S_i, \theta_i),$$

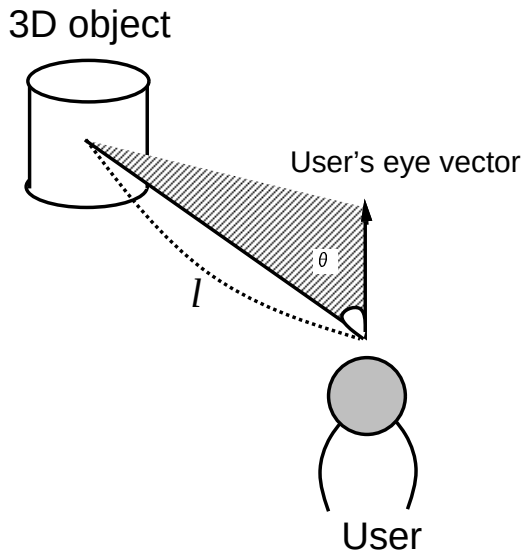

Figure 3: Positional relation between a user and an object

$$IoP_i = \frac{I_i}{\sum_{i=1}^{N} I_i} \quad (1)$$

Here, l_i, S_i, and θ_i are l, S, and θ of the ith object respectively.

Now, in order to determine I_i, we consider the following assumptions:

- Objects behind the user can be ignored in terms of quality.
- Objects which is considerably further from the user can be ignored in terms of quality.
- A user is likely to perceive an object which is far from the user but is large more than one which is close to the user but is small.
- A user is likely to perceive an object quite close to him even if its size is small.

For the first assumption, if $-\pi \leq \theta < -\frac{\pi}{2}$ or $\frac{\pi}{2} < \theta \leq \pi$ in Figure 3, the object is behind the user and its IoP can be considered as 0. For the second assumption, if $l > L_{max}$ where L_{max} is the maximum distance visible to users, the object is further from the user and its IoP can be considered as 0. In order to take account of the remainder of the assumptions, the space around a user is divided into two regions by using threshold $L(< L_{max})$ (see Figure 4). If an object is within L, the distance between the user and the object is considered as an important factor to determine its IoP. Otherwise, the size of the object is dominant.

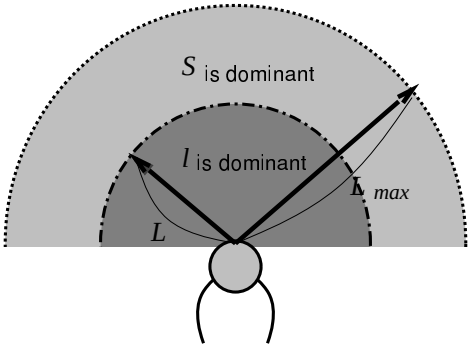

Figure 4: Two regions

Therefore, I_i is defined as follows:

$$I_i = \begin{cases} 0 & (\text{if } -\pi \leq \theta_i < -\frac{\pi}{2} \text{ or } \frac{\pi}{2} < \theta_i \leq \pi) \\ 0 & (\text{if } l_i > L_{max}) \\ \omega_1(L_{max} - l_i)\cos^2\theta_i + \omega_2 S_i & (\text{if } L < l_i \leq L_{max}) \\ \omega_3(L_{max} - l_i)\cos^2\theta_i + \omega_4 S_i & (\text{if } l_i \leq L) \end{cases} \quad (2)$$

Here, $L_{max} - l_i$ represents the degree of the importance of the distance between the user and the object, and using $cos^2\theta_i$ can represents that the object is more important if it is close to the user's eye vector. And ω_1, ω_2, ω_3, and ω_4 are the weight coefficient. If an object is further than L, S_i must be the dominant factor, therefore $\omega_1 \ll \omega_2$. If an object

is within L, $(L_{max} - l_i)\cos^2\theta_i$ must be the dominant factor, therefore $\omega_3 \gg \omega_4$. Moreover, it should be that $\omega_1 \ll \omega_3$ and $\omega_2 \gg \omega_4$.

The normalized IoP value(IoP_i) of each object in a virtual space can be derived from two expressions (1) and (2), and it must be:

$$0 \leq IoP_i \leq 1,$$

$$\sum_{i=1}^{N} IoP_i = 1$$

3.2 Scalable Object

The object whose quality can be controlled according to the currently available resource is called *scalable object*. Current VRML provides a notion of scalable object called *level of detail(LoD)* for 3D objects. In case of the integration of live videos into virtual spaces, video objects must have the scalability. LoD for video objects is defined based on the temporal resolution or the spatial resolution. Also, applying layered coding to a live video, a video is considered to have a notion of LoD by changing the number of receiving layers.

Providing the scalability to the individual object makes a NVR system more tolerant for resource starvation, because the system can control the quality of the presentation of a virtual space subtly.

3.3 Dynamic Object Scaling Algorithm

When the network or the end system can not provide guaranteed service, and their resources are going to starve, our NVR system deteriorates the LoD of each object based on its IoP, in order to adapt currently available resources.

Figure 5 shows the algorithm of the dynamic object scaling. At first, our virtual reality system tries to display all objects visible to a user at the highest degree of LoD. When the resource starvation occurs, the LoD of the object which has the lowest IoP is deteriorated until the lower limit of LoD. The lower limit of LoD is selected as follows:
Assume that there define M_i degrees of LoD for ith object in a VRML file which represents a virtual space, and that the first degree of LoD is the lowest and the M_ith one is highest. The jth degree of the ith object($j = 1 \ldots M_i$) that satisfies the following condition is considered as lower limit of LoD:

$$\frac{j-1}{M_i} \leq IoP_i < \frac{j}{M_i}$$

If resource starvation is solved during this operation, the deterioration of LoD for objects will be stopped. Otherwise, the LoD of the object which has the second lowest IoP is deteriorated until the lower limit of LoD. This operation is carried out from the lowest IoP's object to the highest IoP's one until resource starvation is solved.

3.4 Detection of Starvation

In order to detect the resource starvation, our virtual reality system periodically monitors the buffer of the client which receives the VRML description of objects in a virtual space and the refresh rate of rendering the virtual space at the client. If the refresh rate decreases, it is considered that the client resource is going to starve. And also, if the underflow occurs at the buffer of the client, it is considered that the network resource or the server resource is going to starve. Then, LoD deterioration is performed.

After the LoD deterioration if the resource starvation is solved and the stable state of the virtual reality system continues for a certain period, the system tries to upgrade the LoD of the object from the highest IoP's object to the lowest IoP's one.

4 Object Transfer Mechanism

A user in a virtual space waits until the entire VRML file is transferred completely from the

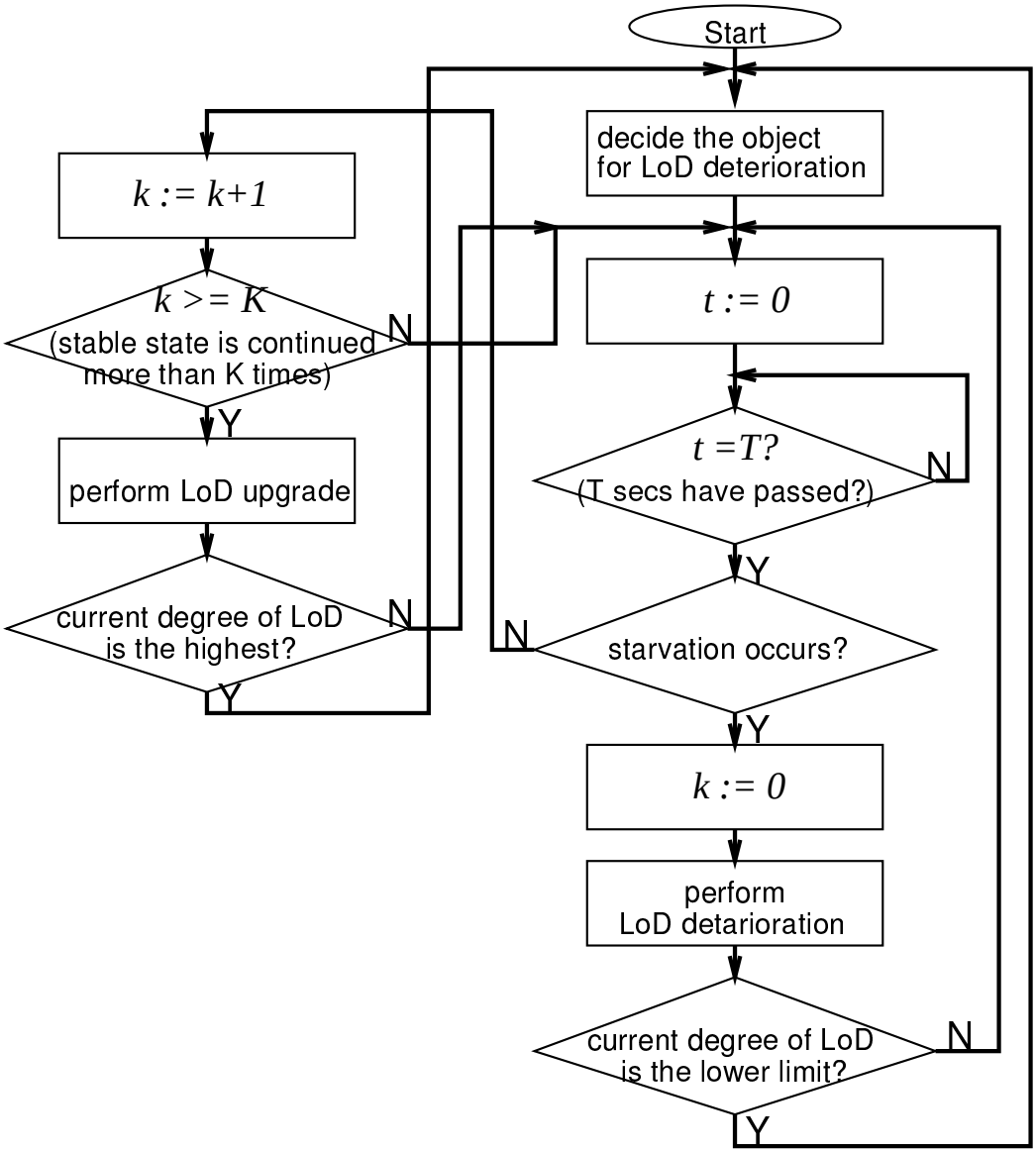

Figure 5: Dynamic QoS adaptation algorithm

server to his client when he starts a NVR application. Also, every time a user moves from a virtual space to another, the navigation may stop and he may wait until the transfer of another entire VRML file is finished and the virtual space is constructed. The larger or the more complex the virtual space is, the longer a user waits. To avoid such a problem, we adopt the object transfer mechanism called *object streaming* which transfers a subset of objects in a virtual space incrementally along with user's motion. The basic policy of the object streaming is that when the IoP value of an object becomes greater than zero, the description of the object will be sent.

However, since the calculation of IoP of each object and the decision that which object should be sent in the object streaming may cause some delay at the client, a user may not be able to navigate a virtual space smoothly. For the smooth navigation through a virtual space, RING system proposed a prediction mechanism for reducing the delay of desk-to-memory copy[3]. We apply a similar mechanism to the client-server environment. The prediction is expected to work efficiently in our virtual reality system, because a motion of a user in a virtual space has continuity.

The server in our virtual reality system predicts the next position of a user in t seconds by using the information about the current user motion transferred from the client. For the prediction, the client periodically sends the following information to the server:

- user's current position
- user's eye vector
- type of user's motion such as straight and rotation
- speed of user's motion

5 Implementation

In this section, we describe design and implementation methodology of a NVR system with a dynamic quality control mechanism. Our NVR system is implemented on SGI indigo2 workstation using Performer[7] as a rendering subsystem.

The overview of system equipped with the proposed quality control mechanism is as shown in Figure 6.

In Figure 6, the server is composed of a VRML analyzer, a QoS adjuster, a QoS controller, an object streaming subsystem and a system monitor. The VRML analyzer extracts information such as location, rotation and size of each object from a VRML file and then saves it as an another file. After the QoS adjuster calculates the IoP of each object, the QoS controller adjusts the LoD of each object according to its IoP. The system monitor is watching the CPU load of both the server and the client, and the network load.

The client is composed of a rendering system, a scene graph builder, a user's viewpoint sender, a QoS controller, and object streaming subsystem and a system monitor as shown in the lower part of Figure 6. The rendering system displays objects in the scene graph on the computer screen. It updates the viewpoint of the user due to his moving, and renders the scene graph and finally displays rendered results on the screen. The prototype of our NVR system displays objects in its scene graph at every 1/30 seconds. The scene graph builder inserts/deletes an object into/from the scene graph. A motion JPEG and a live video from a camera at the server are used as a video object in the prototype system and the video object is treated as a texture during displaying cycle. Since only RGB type objects can be put on the scene graph, each frame of a video object is decompressed to a RGB type object in the scene graph builder. In the prototype system, we use the Cosmo compress of SGI which is a hardware accelerator for compressing/decompressing of the video for fast decompressing.

The scene graph builder controls the spatial quality of the presentation. In that case, it changes the LoD of the objects in the scene

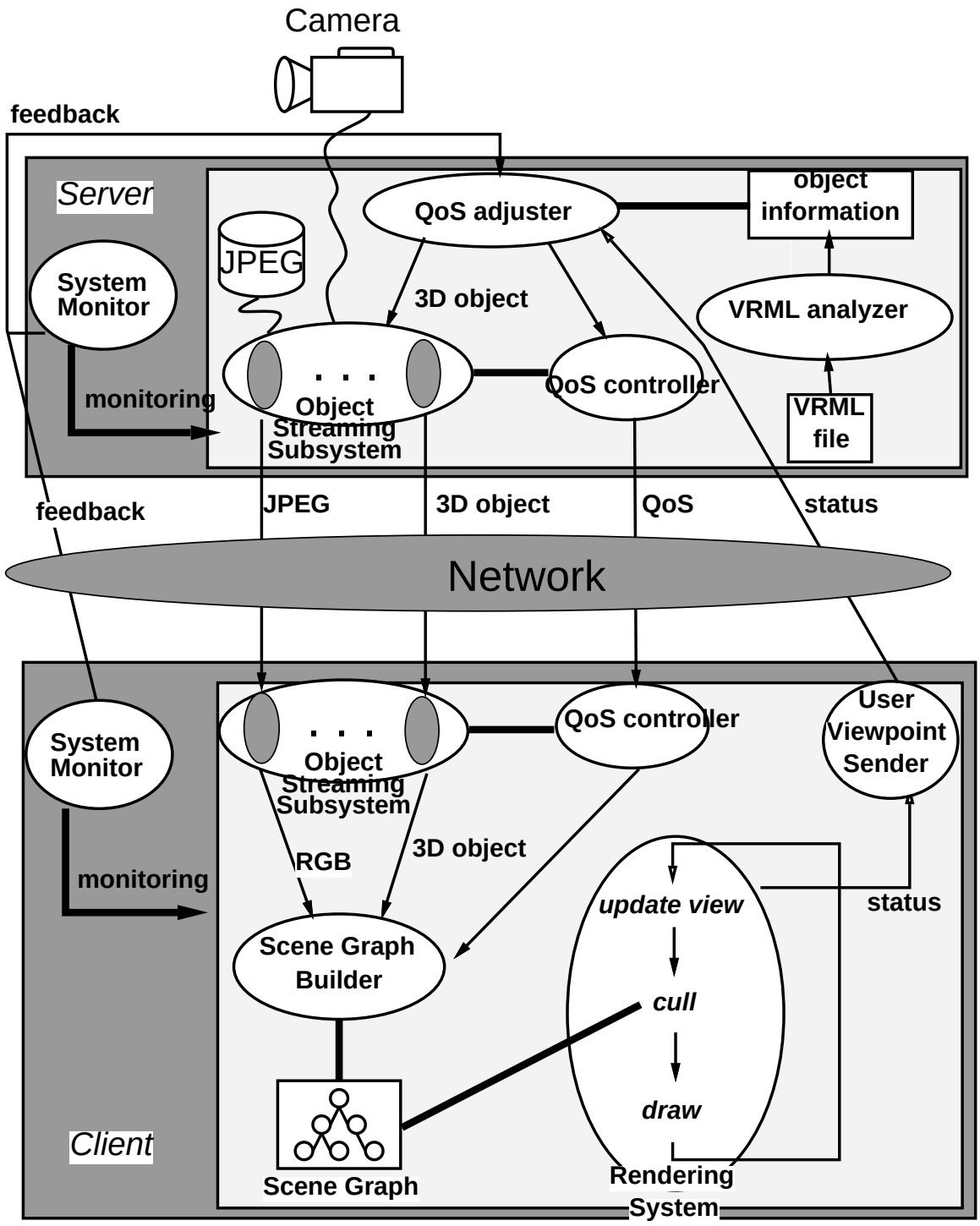

Figure 6: Overview of our networked virtual reality system

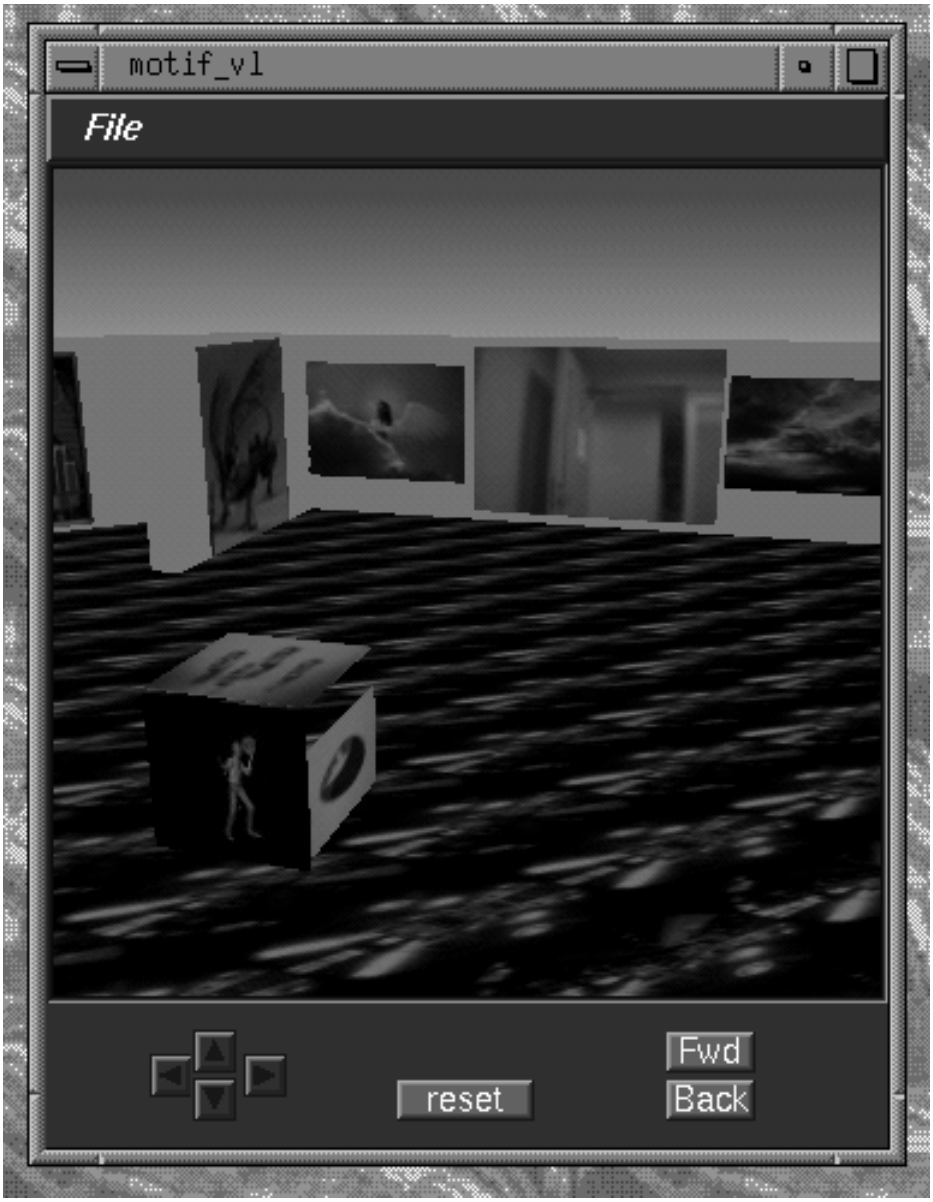

Figure 7: Virtual museum

Table 1: Comparison of mean waiting time

status	mean waiting time(second)
without the mechanism	8.47
with the mechanism	5.32

graph. In case of a video object, the scene graph builder can change temporal resolution by reducing the frame rate of the video object. The user viewpoint sender receives the viewpoint information of the user from the rendering system and then extracts the coordinates of the viewpoint and the moving vector. Finally it sends them to the server at a certain interval.

Now, we describe the experimental results of our NVR system equipped with the proposed dynamic quality control mechanism. To show effective results of our system, we prepared two systems, one with the proposed quality control mechanism and one without any quality control mechanism. In both system, a virtual museum is considered as an example of a virtual space. Figure 7 shows an appearance of the virtual museum where the image in the most front object, where a man is dancing, is a JPEG video and the the image next to the most right is a live video from a camera at a server. The size of the VRML file which represents the virtual museum is 550,432 bytes.

The waiting time until a user can enter the virtual museum is measured on the system both with and without the quality control mechanism. Table 1 shows the mean waiting time of 10 experiments.

Table 1 designates that the difference of the waiting times is rather wide even though the virtual model is relatively simple. We believe that difference becomes greater in larger and more complex virtual space model.

6 Conclusion

In this paper, a *dynamic QoS control mechanism* is presented for the seamless integration of a live video into a virtual space and the fine interactivity in navigation through a large virtual space. In order to provide higher interactivity, our virtual reality system provides the *object streaming* which can treat a large model interactively. Besides, it adopts the prediction of user's movement for smooth navigation. And also, for the seamless integration of a motion video including a live video into a virtual space, We introduce the notion of LoD and scalable object to the video objects as well as the 3D objects. To keep the quality of a virtual space even though network and local resources are going to starve, it introduces the *IoP* (Importance of Presence) of objects and adapts to the starvation by reducing the degree of LoD of the object based on IoP.

Experimental results on the SGI Indigo2 show our NVR system equipped with the proposed quality control mechanism is efficient in respect of smooth navigation and smaller user waiting time.

The following items are considered as our future work:

- extension to the multiuser environment.
- consideration of dynamic objects such as avatars or combat airplanes in the DIS.
- consideration of occluded objects in the calculation of IoP.

References

[1] "VRML 2.0," *http://vrml.sgi.com/moving-worlds/spec/part1/nodesRef.html.*

[2] M. Arikawa, A. Amano, K. Maeda, R. Aibara, S. Shimojo, Y. Nakayama, K. Hiraki, H. Nishimura, and M. Terauchi, "Dynamic LoD for QoS Management in the Next Generation VRML," *Proceedings of third IEEE International Conference on Multimedia Computing and Systems(ICMCS'96)*, pp.24–27, June 1996. Hiroshima, Japan.

[3] T.A. Funkhouser, "Database Management for Interactive Display of Large Architectural Models," *Proceedings of Graphics Interface '96* pp.1-8, May 1996. Toronto, Ontario, Canada.

[4] O. Hagsand, "Interactive Multiuser VEs in the DIVE System," *IEEE MultiMedia*, Vol.3, No.1, pp.30–39, Spring 1996.

[5] M.R. Macedonia and M.J. Zyda, "A Taxonomy for Networked Virtual Environment," *IEEE MultiMedia*, Vol.4, No.1, pp.48–56, January-March 1997.

[6] H. Nakanishi, C. Yoshida, T. Nishimura, and T. Ishida, "Freewalk: Supporting Casual Meeting in a Network," *Proceedings of International Conference on Computer Supported Cooperative Work*, pp.308–314, 1996.

[7] J. Rohlf and J. Helman, "Iris Performer: A High Performance Multiprocessing Toolkit for Real-time 3D Graphics," *Proceedings of ACM SIGGRAPH '94*, pp.381–394, July 1994.