Interactive Cooperative Learning System Based on Virtual Shared Space: HyCLASS

Katsumi Hosoya[1], Akihisa Kawanobe[1], Susumu Kakuta[1] and Munish Sharma[2]

[1]*Nippon Telegraph and Telephone Corporation*
[2]*University of Toronto, Department of Electrical and Computer Engineering*

Abstract

A collaborative educational system has been designed that enables several students in distant locations to share a virtual three-dimensional space. It can be used to perform virtual experiments and carry out creative tasks. Students can dynamically create a new object and modify its properties. The system maintains a consistent status of the virtual space as seen by each student present in a room. An efficient communication method to maintain this consistency is proposed. Within the virtual space, the three-dimensional materials are based on OMG-CORBA, a distributed object modeling architecture. Furthermore a method to create a new object and modify its properties is also introduced. Lastly, after developing and implementing the prototype system, we evaluate the various issues centering on the proposed methods.

Keywords—CSCL, virtual reality, shared space, CORBA, multicast

1 Introduction

Recent studies on intelligent Computer Assisted Instruction (CAI) have suggested that interactive learning, in which students actively interact with the educational material, and collaborative learning, in which students talk with others in a group, play significant roles in the process of acquiring new knowledge. Additionally, people are becoming more interested in education supported by computers, especially "Computer Supported Cooperative Learning (CSCL)". In CSCL, various technologies are used, such as electronic mail/news, chatting, voice/video conferencing, and "white boards".

With recent improvements in computer technology, three-dimensional computer graphics (3D-CG) is being applied in various fields, including education. By using educational materials presented using 3D-CG, students can more intuitively understand three-dimensional knowledge, and actively study, for example, moving around the space and observing a object from various viewpoints [1]. This 3D-CG technology enables the building of a virtual world that cannot be built in the real world, and to describe complicated structures.

Recent works have described educational materials by using the virtual reality modeling language (VRML) [2]. Using VRML, modeling data can be downloaded from a server in a remote location through networks, enabling several users to access the same virtual world. However, a user cannot recognize other simultaneous users and cannot cooperatively manipulate a virtual object with those users.

We have been developing an environment where users can perceive and respond to the actions of other users and manipulate the objects cooperatively [3]. In this paper, we propose a collaborative learning environment that runs on computer networks and discuss the requirements for managing a multi-user virtual space. We then explain how we place the educational objects in distributed locations, and synchronize communication between them. Finally, we evaluate our proposed methods based on their performance in a prototype system.

2 Collaborative learning environment using a virtual space

2.1 Features

Our learning environment has the following features:

(1) virtual shared space (room)

An ordinary school contains several units of space, each classified as a classroom. Students inside a given room work on the same educational material. A student can exist in one room at a time, but can freely move from one room to another.

(2) view of three-dimensional educational materials and walk-through

Students in the same room see the same three-dimensional objects. If a property of an object changes, then all students see it changing at the same

time. Each student can walk inside the room and observe the objects from various viewpoints.

(3) avatars

To enable a student to recognize other students collaboratively working in the same room, the "avatars" are shown on the student's PC screen, showing where the other students are in the room and in which directions they are looking.

(4) real-time conversation

To enable students to communicate (discuss, ask/answer questions, etc.), students can use text chatting or voice/video conferencing. This results in effective collaborative learning.

(5) cooperative operation of virtual educational materials

Students can actively interact with the virtual materials in a room. For example, they can move and rotate an object, transform its shape, change its color, and attach a new part to it. They can also activate one of its pre-defined behaviors, for example a ball can be forced to fall downwards, as it would fall under the influence of gravity in real life. Each student observes the changes caused by many types of attached behaviors at the same time.

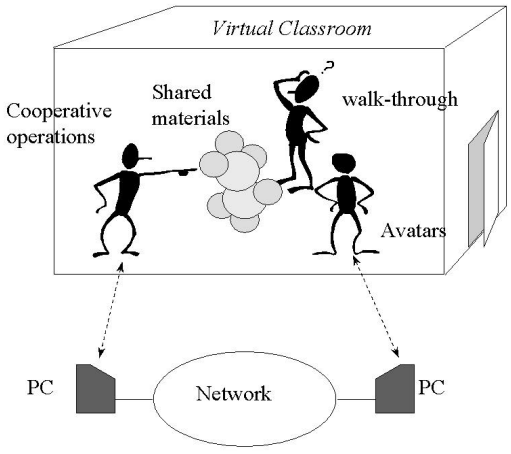

FIGURE 1. Concept of HyCLASS

2.2 Related work

Presently some other studies allow multiple users to share a virtual space through networks, which include DIVE [4], World Chat [5] and Community place [6]. Most of them are designed in a server-client model, consisting of a server that monitors the status of the shared space and manages its consistency, and several clients which are used by the individual students. DIVE is an exception, which uses a unique protocol developed at the Swedish Institute of Computer Science (SICS) and it features a distributed architecture without a server.

Most of these systems include features (1), (2), and (3) described in the previous section. For feature (4), they integrate various things like chatting, and voice/video conferencing, depending upon the bandwidth of their networks. Very few systems provide feature (5), entirely because the main objectives of these systems are such fields as games, chatting, and electronic shopping. For these fields, it is not always necessary to strictly maintain the consistency of the shared space.

For education, some of the reported systems can be applied to learning foreign languages. However, in situations where close interaction with objects is needed, such as physical experiments and creative work, the mechanisms provided are not sufficient. In these situations, it is very important for the system to strictly manage the consistency of the dynamic status of the shared virtual space. For changes in object status to be observed by all students consistently, the following issues must be addressed:

- how to keep the current status of the shared space and to manage its dynamic changes?
- what is an appropriate object model of educational materials?
- what is the best interface for creating/destroying objects and modifying their properties?
- how can communication be established between distributed objects ?

In many existing distributed systems, many algorithms have been developed, including message based system[7] and distributed memory [8]. However requirements that a shared virtual space impose on data consistency differ from those in traditional distributed systems. In traditional algorithms data consistency is always maintained even though the algorithms are designed to relax strict serial consistency. On the other hand, applications using a shared virtual space are more concerned with a virtual notation of consistency, where perceptual consistency is more important than logical consistency [9]. Usually there are only a small number of cases where strict consistency is required.

3 Management of data in a virtual shared space

3.1 Distributed management of educational objects

In such educational systems as described in the previous section, it is important for students to be able to create new objects and change their properties.

Therefore, the system must store the static status of the educational materials and manage their dynamic status. To speed up the graphical drawing process, each client should have a copy of the dynamic graphic data. For a new client to acquire the current status of the shared space, there are two alternatives: get it from a server, or get it from an existing client. In the latter case, if all clients exit the room, the current status disappears. To keep the current status in this situation, we added one special client into the system, which is called a "pseudo-client", as shown in Fig. 2. It may be implemented on the same computer as the server or on a separate one. As shown in Fig. 2, the educational material data is created and stored in the material database. When the pseudo-client starts, it loads the material from the database and makes its replica locally. When Client A and B join the room, the replica data on the pseudo-client is copied on to each client. The change of status is informed to all the clients and the pseudo-client, and then all the replicas are updated.

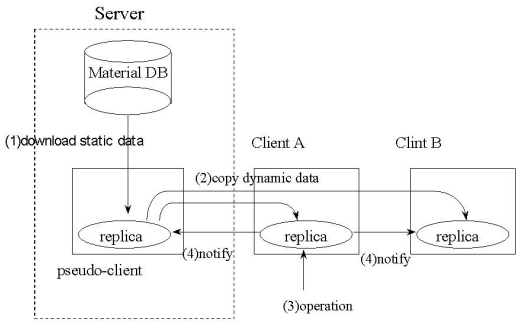

FIGURE 2. Management of material data

3.2 Structure of an educational object based on CORBA

We define a distributed object based on the common object request broker architecture (CORBA) model [10]. The basis of CORBA is the object request broker (ORB) and the interface definition language (IDL). Each interface specification is defined by IDL, and the distributed objects communicate with each other through the ORB. Thus, each object can be accessed through ORB without knowing its geographical locations by using the simple interface defined by IDL. Figure 3 shows the structure of each educational object in the virtual space. All resources in each educational object are managed by a single CORBA object "equipment". It manages scene objects units, behaviors and other resources. A scene object unit has graphic nodes and a behavior may have sound objects.

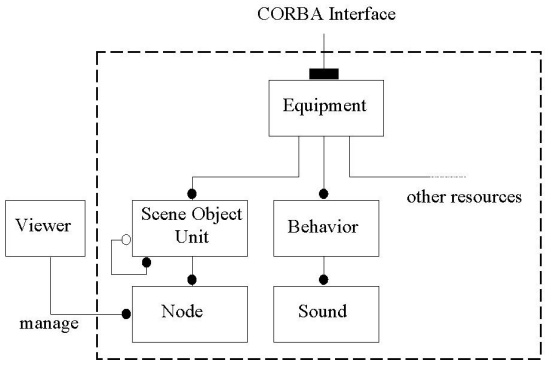

FIGURE 3. Structure of each educational object

4 Synchronization between objects

4.1 Classification of cooperative operations and communication methods

For several clients to share the current status of the virtual space, they must exchange messages with each other and update their local status whenever the room status changes. Messages must be exchanged very frequently to ensure that the status in each client is correctly synchronized. However this requires high-speed networks with short delays, and this is constrained due to a higher cost. For practical use, the messages should be categorized and the most suitable communication method should be identified for each category. We categorized operations for three-dimensional objects in the shared virtual space as follows and identified a suitable communication method for each of the category as shown in Fig. 4.

Type 1: operations that do not affect other clients (e.g. reading the status of an object). For these operations, a requesting client directly sends the request to the destination objects.

Type 2: operations that require strict synchronization among all clients (e.g., turning on a light). For these operations, a requesting client first sends the request to the room server, which verifies its acceptability and its status in the room, and then forwards the request to the destination. The room server also broadcasts a callback message to all clients advising them to update their proxy objects.

Type 3:operations that require a quick response (e.g., moving an object or one's own viewpoint). For these operations, a requesting client sends a request to all clients by IP multicasting, not using the ORB. Requests of this type are usually followed by Type 2 requests. For

example, while an object is being moved, its position data is broadcasted by IP multicast. When movement stops, every client adjusts the status of its proxy object by a Type 2 request.

In our system, these three synchronization mechanisms are automatically selected in a proxy object in each client to hide the selection process and to make the application interface simple.

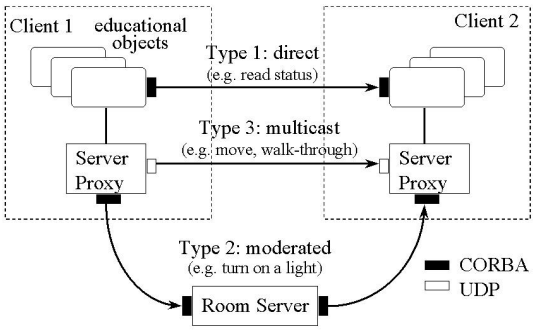

FIGURE 4. Synchronization mechanism

4.2 Levels of consistency

In Type 2 messages, the room server verifies the acceptability of messages sent from each client to maintain the consistency of the status in the room. To keep strict consistency, the simplest way would be to provide a "lock mechanism", enabling only one user to operate on an object at a time. However this would make it difficult for users to get the feeling of actually operating an object collaboratively with others, defeating our objective of providing a collaborative learning environment.

For several students to manipulate the same object, several messages affecting that object must be sent simultaneously. For such messages changing the object's status, the message must be kept in the correct order. For example, suppose several users want to change the color of a traffic signal having three colors (red, blue and yellow). If the order of the messages arriving at each client varies, the final color of the signal may not be the same among all clients.

On the other hand, messages affecting separate objects or independent properties may not need strict management of the order. For example, even if a message causing the change of color and a message causing the change of shape arrive in different orders, then the final state will be the same as other clients.

In addition, for some messages, whether they are valid or not depends on the current status of the object. The server must therefore have a mechanism that rejects a request message if it is no longer valid. For example, for a light switch, when the light is "off", the operation "turn on" is valid but the operation "turn off" is invalid. To check such restrictions, a local checking function at each client is not sufficient, instead a global mechanism managing the status of entire room is needed.

4.3 Consistency management using sequence numbers

The room server should control the sequence of messages so that messages causing change in the status arrive at all clients in the same order. In our system, the server serializes arriving messages and then sends them on to the clients. However, the message order may change during transmission through the networks. A sequence number attached to each message by the server would enable each client to determine the correct message order.

We designed the message management mechanism in our system to use sequence numbers, as shown in Fig. 5. The server and each client has a table that stores the latest sequence number, S_n, corresponding to each object. In a stationary situation, the server and all clients have the same sequence number corresponding to a certain object. When an event occurs at one client, it sends a message with the sequence number S_{n+1} to the server, and the server forwards it to the other clients. The server and the other clients update the sequence number to S_{n+1} when they process this message. The server accepts messages with later sequence numbers than those stored in its table. When two clients send messages corresponding to the same object at the almost the same time, the earlier message is accepted and the later one is rejected because its sequence number is not the latest. In this way, the status of the room and of its objects are always consistent, even when several clients send requests randomly.

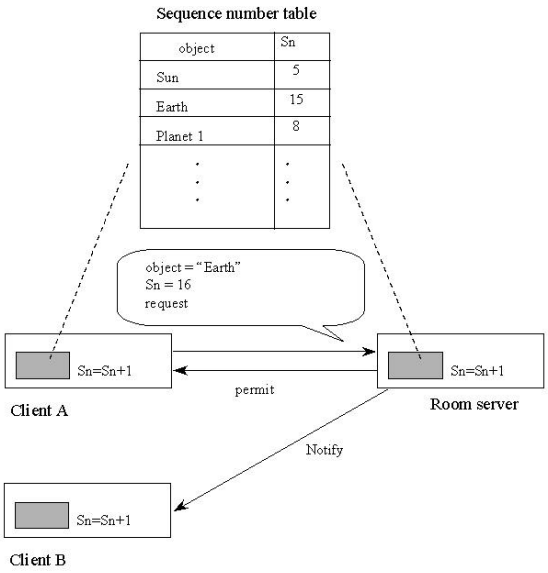

FIGURE 5. Message sequence number

5 Prototype system

5.1 Features

We have developed an experimental system providing a collaborative learning environment based on the above considerations. This "HyCLASS" system was designed so that several students can work together in a three-dimensional environment, to learn in active and collaborative manner. Its main features are allowing students to "walk around" in the learning space, to change their viewpoints, to see avatars representing other students, and to talk with the students in real. To achieve this interactive and collaborative leaning, it enables students

- to activate behaviors attached to objects,
- to create new objects and modify their properties,
- to attach new behaviors to objects,
- to define a group object that includes several child objects.

5.2 System architecture

As shown in Figure 6, the system consists of a server that manages the educational material data in the room, several clients operated by students, and networks connecting the server and the clients. The server and clients run on a Windows NT- or Windows 95-based PC. The rendering software used for three-dimensional graphics is Open Inventor (Template Graphics Systems Inc.). An Open GL accelerator board is used to achieve sufficient rendering performance. For the CORBA software, we use Orbix (IONA Inc.). The system consists of several blocks:

(1) server side

Material database: stores educational materials and graphical data. It can be accessed using HTTP protocol. Each client downloads the required material data from it.

Room server: receives a message from a client, verifies its acceptability, and forwards it to other clients when the corresponding operation occurs on the client.

Pseudo client: maintains the static status of the room. A client joining the room gets the latest status of the room from it. The rest of the functions are almost the same as those of a standard client.

(2) client side

Communication control block: controls sending and receiving of messages between the room server and clients, and downloading them from the material database.

User interface control block: controls the interface to change viewpoints and to manipulate objects.

Graphic control block: displays three-dimensional space.

Audio control block: controls sound effects and real-time conversation between students.

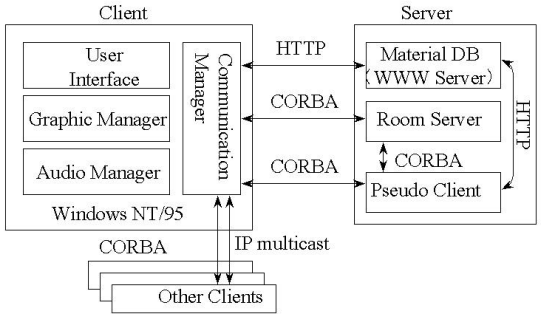

FIGURE 6. System architecture

5.3 Sample materials

To evaluate the system, we developed materials for providing an astronomy lesson, as shown in Fig. 7. In this "virtual room" there is the sun, the earth, and the moon. A student can change their positions and create a new planet. Several students can enter this room at the same time, walk in it, step on top of the earth which is rotating and revolving, and finally observe the sky from various points.

Figure 8 shows an example of studying the structure of the human skeleton. A student can study a single bone by grasping it and bringing it closer. In this way the student gain an intuitive understanding of the three-dimensional structure of the skeleton. A teacher may give a group of students the task of finding the correct position of each bone. Scince each bone has a behavior bringing it to a pre-defined home position, the students can get help by activating the behavior.

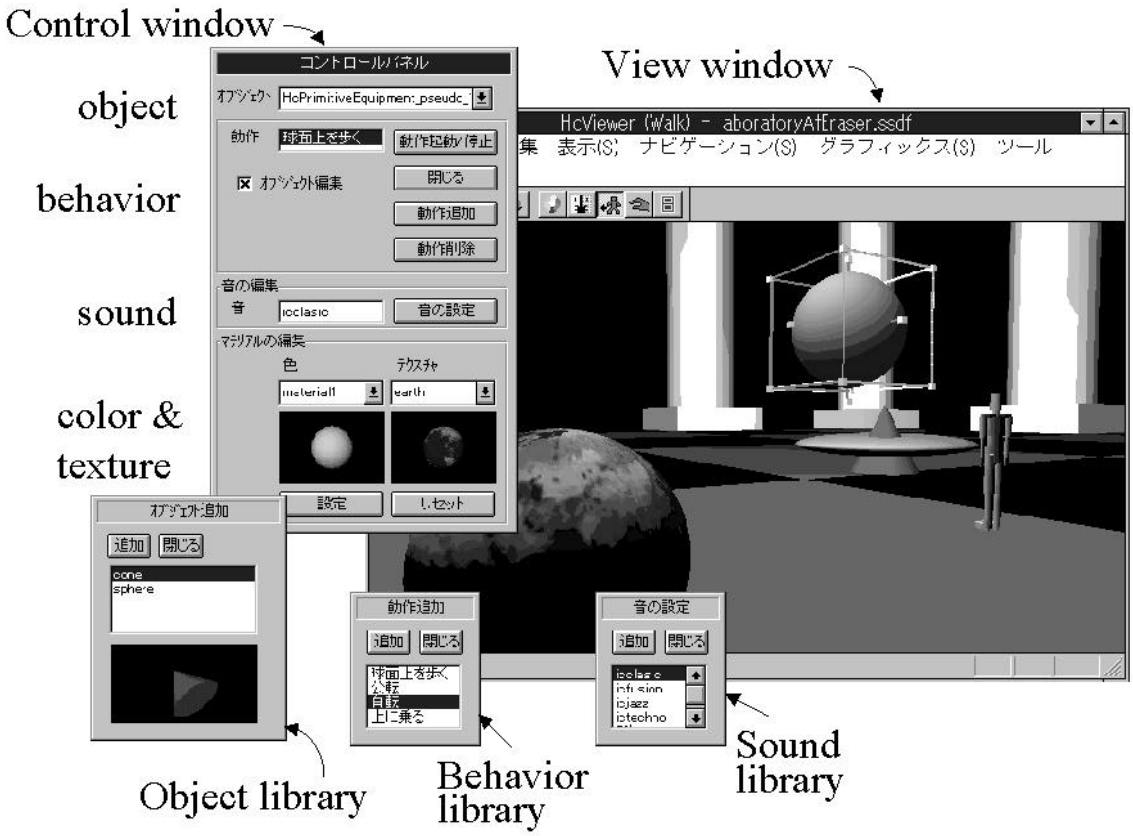

FIGURE 7. Example 1 (astronomy)

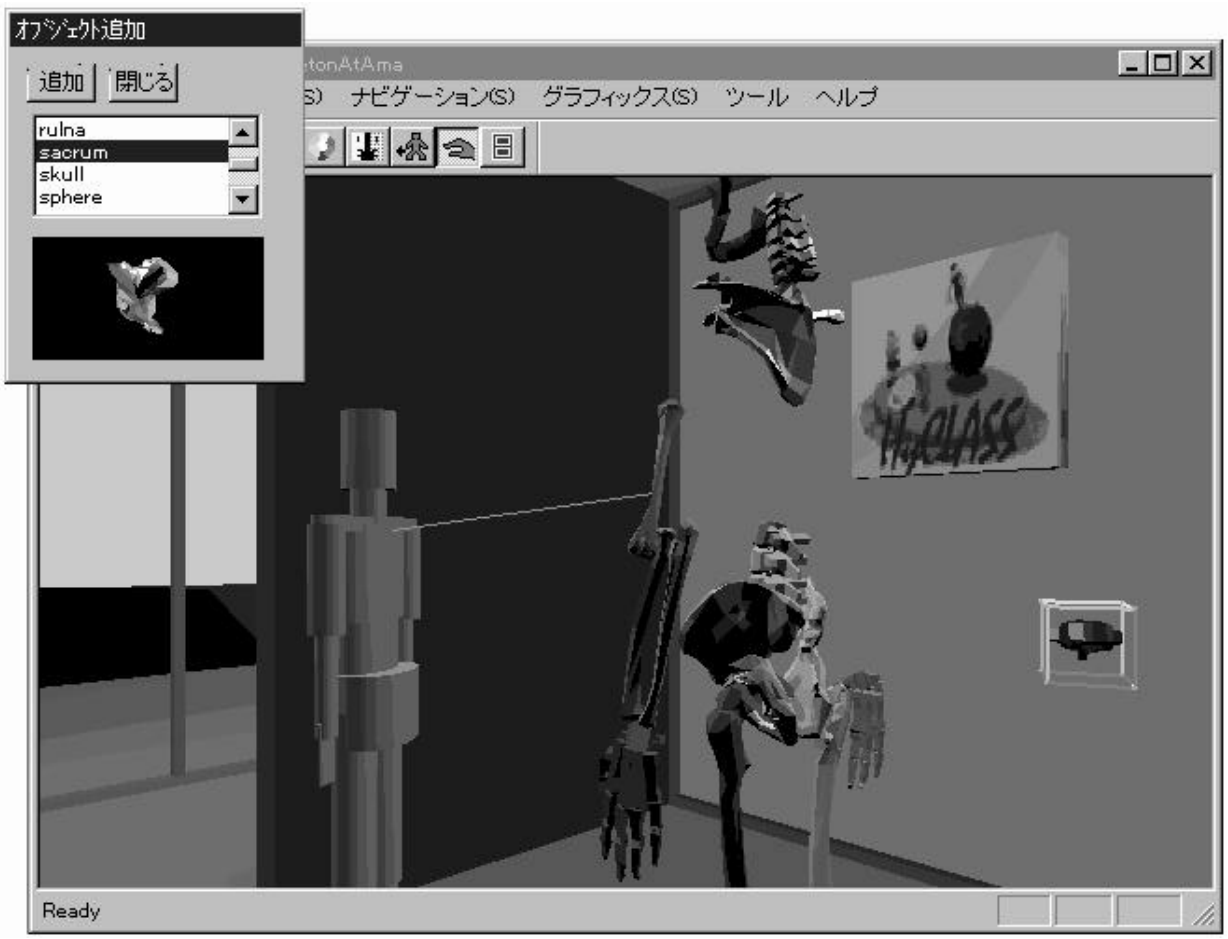

FIGURE 8. Example 2 (skeletal system)

6 Discussion

6.1 Creation of materials and definition of a new object

In HyCLASS, because each material is constructed based on the object-oriented model, a new material can be created using components of existing materials. We could therefore develop the materials mentioned above very quickly (about three days for each one). In our prototype, the function allowing an end user to define a new object was a little confusing to some users because only fixed list of a few objects was provides. We are enhancing the system so that a user can modify the object list; this will enable users to carry out more creative tasks.

6.2 Response time of user operations

It has been reported that a delay in the "action-result" cycle of more than 250msec will deter users from using the system [11]. In our system, transmission overhead is the highest when the message must be verified in the room server. To evaluate the performance in this situation, we measured the delay time between a student requesting the rotation of the earth to stop and then it actually stopping. We focused on the delay between the time the student makes the request to the time when the graphic control block in another client was activated. The server and clients were connected through a single-segment network of 10BaseT. The number of clients at a time was changed from one up to three, and 100 trials were run for each case to get the mean and variation. As shown in Table 1, the delay did not increase when the number of clients was increased to three. A collaborative task can be smoothly carried within the given range of delay, based on the participants' responses.

TABLE 1. Response time

Number of clients	1	2	3
mean (m sec)	20.0	22.7	21.2
std. div. (m sec)	3.6	6.4	3.9

6.3 Advantages of CORBA

In this system, because each educational object is defined based on the CORBA model, any object can be placed on any node and transparently accessed on the network. Moreover CORBA is a substantial standard and we will be able to utilize various CORBA services which are currently being developed, such as naming service, event service, and life cycle service.

7 Conclusion

Our proposed collaborative educational system allows several students in distant locations to share a virtual three-dimensional space. The system maintains a consistent status between objects located on distributed nodes in the network by using an efficient communication method. Each constituent object of the virtual three-dimensional materials is defined as a derived sub-class of a distributed object based on the OMG-CORBA model. A new object can be created and its properties can be modified by a student. Evaluation of our prototype system showed that this system has a promising performance level, while it maintains the consistent status in a virtual room. We are planning to develop a more user-friendly interface and an efficient authoring tool for materials. We will then apply it in more practical educational situations.

References

[1] J. Rickel and W. Johnson, "Intelligent Tutoring in Virtual Reality: A preliminary Report," AIED97, pp.294-301, 1997.

[2] M. Pesce "VRML: Browsing & Building Cyberspace," New Riders Publishing, 1995.

[3] Y. Kato, A. Kawanobe, S. Kakuta, K. Hosoya and Y. Fukuhara, "Advanced collaborative educational environment using virtual shared space," ED-MEDIA'96, pp.348-353, Boston, July, 1996.

[4] O. Hagsand, "Interactive multiuser VEs in the DIVE system," ACM MultiMedia, Vol.3, No.1, 1996.

[5] "Worlds," published on http://www.worlds.net/.

[6] "Community Place," published on http://vs.sony.co.jp/.

[7] L. Lamport, "Clocks and the ordering of events in a Distributed System," Comms. of the ACM 21 (7), pp.558-565, 1978.

[8] K. Li and P. Hudak, "Memory coherence in shared virtual memory systems," ACM transaction on Computer Systems, pp.321-359, 1989.

[9] D. Snowdon and A. West, "AVIARY: Design issues for future large scale virtual environments," Presence, Vol. 3, No. 4, 1994, pp. 288-308, MIT press.

[10] "CORBA/IIOP," published on http://www.omg.org/corba/corbaiiop.htm.

[11] K. Arthur, K. Booth and C. Ware, "Evaluating 3D task performance for fish tank virtual worlds," ACM trans. on distributed systems, 11 (3), pp. 239-265, 1993.

Author's Address

Katsumi Hosoya, Akihisa Kawanobe and Susumu Kakuta: NTT Information and Communication Systems Laboratories, Nippon Telegraph and Telephone Corporation, 3-9-11, Midori-cho, Musashino-shi, Tokyo 180 Japan.
katsumi@isl.ntt.co.jp, nobe@isl.ntt.co.jp, kakuta@isl.ntt.co.jp.

Munish Sharma: Department of Electrical and Computer Engineering, 10 King's College Road, Toronto, Ontario M5S-3G4 Canada.
sharmam@ecf.utoronto.ca.