Packetization Interval of Haptic Media in Networked Virtual Environments

Masaki Fujimoto*
Department of Computer Science and Engineering
Graduate School of Engineering
Nagoya Institute of Technology
Nagoya 466-8555, Japan

Yutaka Ishibashi
Department of Computer Science and Engineering
Graduate School of Engineering
Nagoya Institute of Technology
Nagoya 466-8555, Japan
ishibasi@nitech.ac.jp

ABSTRACT

This paper deals with the packetization interval of haptic media in networked virtual environments. We here handle work in which a user moves an object by manipulating a haptic interface device. In previous studies, we packetize haptic media on capturing them. In this paper, we propose a scheme which lengthens the packetization interval and evaluate the efficiency of the work by experiment. Experimental results show that the packetization interval of several milliseconds produces higher efficiency of the work. We also examine the influences of the packetization interval and network latency on the efficiency of the work in the proposed scheme.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Communications Applications; H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems—*Artificial, augumented, and virtual realities*

General Terms

Algorithms, Performance, Human Factors, Experimentation

Keywords

Networked virtual environments, Haptic media, Packetization interval, Quality of Service (QoS)

1. INTRODUCTION

In networked virtual environments, a number of researchers have studied the usage of haptic media as well as voice and video [1]. By using haptic interface devices, we can touch objects in a 3-D virtual space. Therefore, we can largely improve the efficiency of collaborative work such as remote surgery simulation and design [2] and immerse ourselves in playing networked games [3]. However, haptic media have severer constraints on the network delay than voice and video. For example, the maximum allowable delay is about 30 to 60 ms for haptic media [4], and that of voice and video is around 400 ms [5]. Thus, in previous studies, we packetize haptic media on capturing them [6], [7].

In [6] and [7], where the PHANToM [8] is used as an haptic interface device, haptic media are input at 1 kHz [9]; a *media unit* (MU), which is defined as an information unit for media synchronization (e.g., a picture for video), is transmitted as a packet from each client to a single server every millisecond. However, this produces large overhead and heavy network load.

On the other hand, as for voice, voice information is usually input at 8 kHz. The packetization interval of voice is set to around 20 [10] to 50 ms [11]. This is because the maximum allowable delay of voice is larger than that of haptic media as described earlier; also, it is difficult to process (e.g., transfer and synchronize) voice information at 8 kHz. By transmitting the voice information input during the interval as a packet (that is, an MU in this case), we reduce the overhead.

To decrease the transmission rate of haptic media, in [12] and [13], Hikichi *et al.* and Kanbara *et al.*, respectively, employ a technique of dead-reckoning [14]. Dead-reckoning can keep the output rate of haptic media at 1 kHz by prediction and convergence. However, owing to network delay jitter and packet loss, dead-reckoning may produce large prediction errors.

This paper proposes a scheme which reduces the transmission rate of haptic media by lengthening the packetization interval. The haptic media information generated during the interval is transmitted as a single packet. We also reduce the size of the packet by information compression. The basic idea of the proposed scheme is simple, but its effects are not clear. We investigate the effects of the packetization interval on the efficiency of work by experiment. In the experiment, we further examine the influence of network latency.

The rest of this paper is organized as follows. Section 2 describes a system model for haptic media, and Section 3

*Presently, he is with Casio Computer Co., Ltd., Tokyo 192-8556, Japan.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
NetGames'05, October 10–11, 2005, Hawthorne, New York, USA.
Copyright 2005 ACM 1-59593-157-0/05/0010 ...$5.00.

explains the proposed scheme. Section 4 shows the experimental system, and experimental results are presented in Section 5. Section 6 concludes the paper.

2. SYSTEM MODEL

For simplicity, we suppose that a user moves an object in a 3-D virtual space by manipulating a haptic interface device. We employ the PHANToM DESKTOP [8] as a haptic interface device.

As shown in Fig. 1, a client inputs/outputs an MU, which contains the positional information of the PHANToM cursor (i.e., the positions or contact points) or that of the object, every millisecond by repeating the servo loop [9] at a rate of 1 kHz. Each MU also includes its timestamp, which is the generation time of the MU (denoted by the sequence number of the servo loop [7]). The client transmits a packet which includes multiple MUs to a server.

When the server receives a packet, it gets MUs out of the packet. It calculates the reaction force against the object and updates the position of the object for each MU. It also sends a packet (including multiple MUs) to the client.

The client extracts MUs from a received packet as in the server. After carrying out media synchronization control over each MU, it updates the position of the object and calculates the reaction force applied to the user. For media synchronization, we use the *virtual-time rendering* (*VTR*) *algorithm* [7][1], which dynamically changes the buffering time of MUs according to the network delay jitter. The VTR algorithm employs several media synchronization control techniques such as the virtual-time contraction and expansion, skipping, and pausing [15]. The rendering rate of the virtual space is 30 Hz.

3. PROPOSED SCHEME

Let us denote the packetization interval by P ms ($P > 0$). In the proposed scheme, the client and server transmit P MUs as a packet every P ms. It should be noted that the proposed scheme with $P = 1$ ms corresponds to the scheme handled in [7], which is referred to as the *conventional scheme* in this paper.

The client constructs a packet by adding a header to P MUs which have been generated during the last packetization interval. The header includes the sequence number (used as a timestamp) of the first MU in the packet[2]. Note that the proposed scheme is different from a scheme which is used for voice in the following point: A packet includes multiple MUs for haptic media, but a voice MU is transmitted as a packet[3].

[1]We also employed the Skipping algorithm [7], which outputs only the latest arrived MU every millisecond (that is, which skips obsolete MUs), instead of the VTR algorithm. As a result of experiment, we found that the Skipping algorithm is inferior to the VTR algorithm. This is because only the last MU in each packet is output in the Skipping algorithm; the other MUs are skipped.

[2]In [7], each MU contains the sequence number as well as the positional information. Since we can calculate the sequence number of the remaining MUs in the packet by using that of the first MU in this paper, the MUs do not include their sequence numbers.

[3]In [11], we set $P = 50$ ms for voice, but each packet is an MU. Therefore, we carry out media synchronization control over voice packets in this case.

We also reduce the size of each packet by information compression. The positional information of the cursor or the object in an MU has a strong correlation with that in the next MU. By using the correlation, we can largely reduce the size. We here adopt differential coding and quantization for simplicity. In the differential coding, we obtain the difference in the position between an MU and the next MU in a packet. The first MU in each packet is not coded, and its size is 24 bytes [7]. Then, we quantize the difference by a quantization factor $Q_{\rm f}$ for the remaining MUs in the packet. We set $Q_{\rm f} = 2^{-7}$ in this paper; in a preliminary experiment, we confirmed that the value of 2^{-7} hardly degrades the haptic media output quality. Thus, we can represent every MU except the first MU by 3 bytes, each byte of which is used to express the position in the x-, y-, or z-axis. When we set the header size to 20 bytes (actually, the header size is 20 bytes in our experiment), the packet size is $20+24+3(P-1)$ bytes. On the other hand, without information compression, the packet size is $20 + 24P$.

4. EXPERIMENTAL SYSTEM

We have carried out an experiment in which a user of the client moves a rigid cube as an object in a 3-D virtual space (height: 120 mm, width: 160 mm, depth: 120 mm) by using the PHANToM. As shown in Fig. 2, where a hemisphere beneath the cube represents the cursor of the PHANToM, the user lifts and moves the cube so that the cube contains a target (a sphere) which revolves along a circular orbit at a constant velocity (it takes 10 seconds to revolve once). The mass of the cube is 0.5 kg, and the acceleration of gravity is 2.0 m/s^2. Each side of the cube is 30 mm, and the radius of the sphere is 10 mm. The sphere and orbit do not collide with the cube or cursor.

Figure 3 shows the configuration of the experimental system. In the figure, we connect the client (CPU: Pentium4 processor at 2.8 GHz, RAM: 512 Mbytes, OS: WindowsXP) to the server (the same specifications as the client) through an Ethernet hub (10BASE-T) and a network emulator (NIST Net [16]) which is used to generate network latency. The client is connected to the network emulator by a 100BASE-T cable. In order to generate a traffic flow of interference, we also connect two workstations (WS1 and WS2) to the Ethernet hub. WS1 sends fixed-size data messages of 1472 bytes each to WS2 at exponentially distributed intervals. As the transport protocol, we use the UDP protocol in this paper.

5. EXPERIMENTAL RESULTS

We first make a performance comparison among the proposed scheme, the proposed scheme without information compression (denoted by the *proposed scheme without IC* in this paper), the conventional scheme, and a scheme which employs adaptive dead-reckoning [13] (referred to as the *dead-reckoning scheme* here). The parameters and thresholds of the dead-reckoning scheme are set to the same values as those in [13]. The other schemes use the same parameter and threshold values as those in [7]. Next, we investigate the influences of the packetization interval and network latency on the performance of the proposed scheme.

As performance measures, we employ the *average distance between the object and the target* and the *average MU rate* [7]. The average distance between the object and the target is defined as the mean distance between the center of

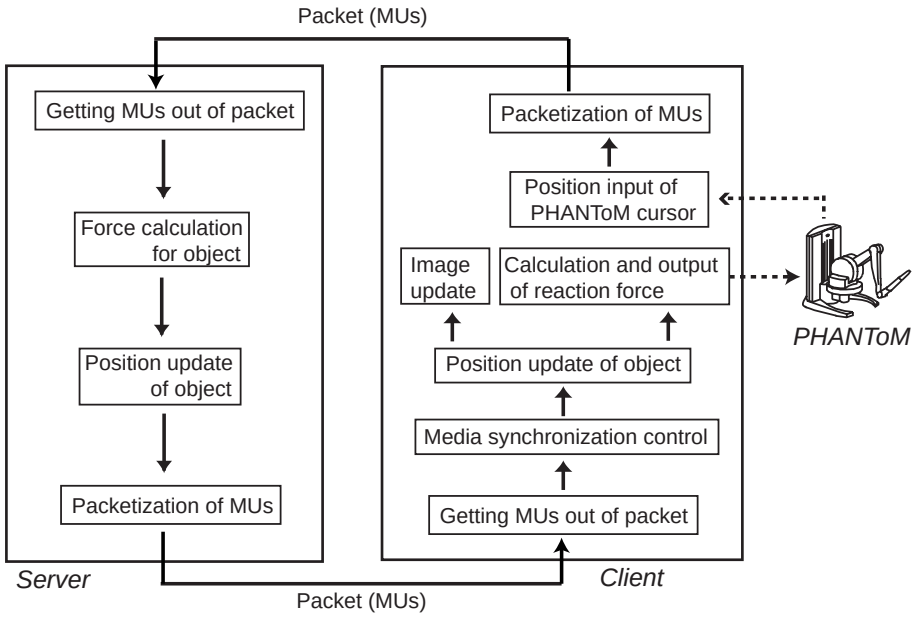

Figure 1: A system model.

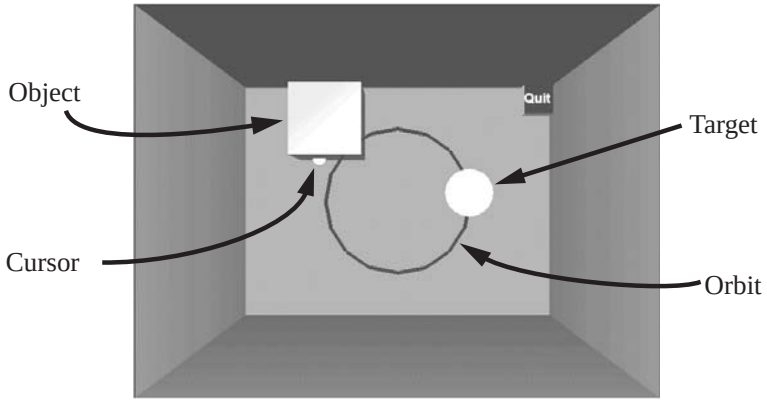

Figure 2: A displayed image of the virtual space.

the object (i.e., the cube) and that of the target (i.e., the sphere). This measure is closely related to the accuracy of the work[4]. As the average distance decreases, the accuracy of the work becomes higher. The average MU rate is the average number of MUs output in a second. This has relation to the intra-stream synchronization quality. Larger average MU rates mean higher quality of intra-stream synchronization.

5.1 Performance Comparison

We show the average distances between the object and the target and the average MU rates of the four schemes as a function of the *data load* (i.e., background load) in Figs. 4 and 5, respectively. The data load is defined as the average number of interference data bits transmitted in a second at WS1 (see Section 4). In the figures, we also display the 95 % confidence intervals; however, when the interval is smaller than the size of corresponding symbol representing the experimental result, we do not show it in the figures. In the proposed scheme and the proposed scheme without IC, we set $P = 8$ ms.

In Fig. 4, we see that the proposed scheme has the smallest average distance between the object and the target when the data load is heavier than around 5 Mbps. In this area, the proposed scheme without IC has the second smallest. The difference between the proposed scheme and the proposed scheme without IC is large at the data load of 8 Mbps. This is the effect of the information compression.

We also observe in Fig. 4 that the conventional scheme and the dead-reckoning scheme have the largest or the second largest average distance for the data loads heavier than approximately 5 Mbps. In the figure, the average distance of the conventional scheme at the data load of 8 Mbps is

[4]In [7], the authors investigate the relation between the measure and MOS. They show that the two are closely related to each other.

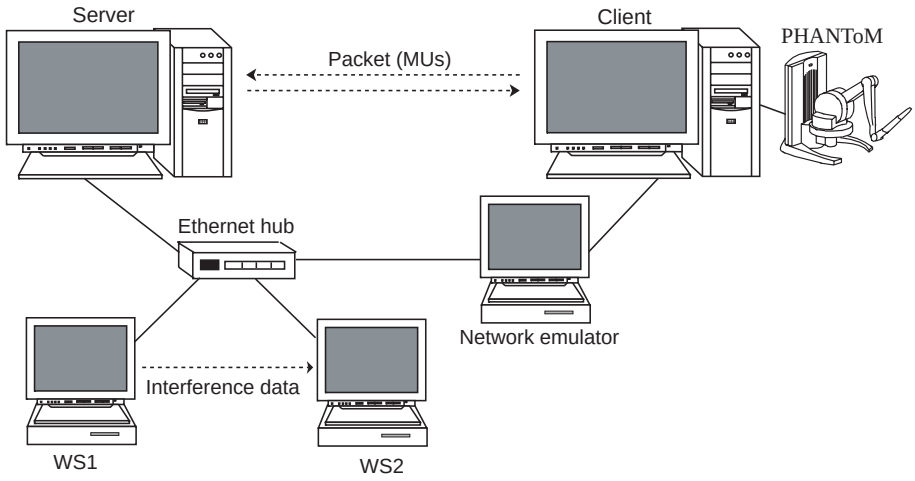

Figure 3: Configuration of the experimental system.

smaller than that at the data load of 7 Mbps. The reason is as follows. When the data load is 8 Mbps, we could hardly move the object; on the other hand, we more frequently dropped the object at the data load of 7 Mbps since we could move the object barely.

In Fig. 5, we notice that as the data load increases, the average MU rates of the four schemes become smaller. This is because the VTR algorithm skips MUs which have arrived largely late [7]. We also see that the proposed scheme has the largest average MU rate in the whole range of the data load considered in this paper. For the data loads heavier than around 6 Mbps, the proposed scheme without IC has the second largest. We further note that the dead-reckoning scheme has the smallest when the data load is lighter than about 6 Mbps. Beyond this range, the conventional scheme has the smallest.

5.2 Influences of Packetization Interval and Network Latency

Figure 6 plots the average distance between the object and the target versus the additional delay for various values of P ($P = 4$, 8, 16, 32, and 48 ms) in the proposed scheme. The additional delay is constant, and it is produced for each packet transmitted from the server to the client by the network emulator. We also show the average MU rate versus the additional delay in Fig. 7. In the figures, we set the data load to 7 Mbps.

From Fig. 6, we find that when the additional delay is shorter than or equal to approximately 15 ms, the differences in the average distance among the values of P are not large. When the additional delay exceeds around 15 ms, the differences become larger. In this area, the packetization interval of $P = 48$ ms has the largest average distance. This is because the first several MUs in each packet are skipped by the VTR algorithm (we can confirm this in Fig. 7), in which we employ the maximum allowable delay (set to 45 ms [7] in this paper); note that the VTR algorithm skips MUs which have arrived largely late. Also, in Fig. 6, the packetization interval of $P = 4$ ms has the second or third largest in the same area, and that of $P = 8$ ms has the smallest. Therefore, the packetization interval of 8 ms is the best in the experiment.

Figure 7 reveals that the packetization interval of $P = 48$ ms has the smallest or the second smallest average MU rate. Also, we see that the packetization interval of $P = 4$ ms has the smallest when the additional delay is 0 ms. On the other hand, the packetization intervals of $P = 8$ and 16 ms produce high average MU rates when the additional delay is smaller than or equal to around 30 ms.

6. CONCLUSIONS

This paper proposed a scheme which lengthens the packetization interval and transmits the haptic media information generated during the interval as a single packet. The proposed scheme also reduces the packet size by information compression. By experiment, we made a performance comparison among the proposed scheme, the proposed scheme without information compression, the conventional scheme, and the dead-reckoning scheme. We also investigated the influences of the packetization interval and the network latency on the performance of the proposed scheme. As a result, we demonstrated that the proposed scheme is superior to the other schemes. When the network load is heavy, the information compression is effective. Furthermore, the packetization interval of 8 ms is the best in the experiment.

As the next step of our research, we will carry out the experiment in various kinds of network environments. We also need to deal with different types of work. In addition, we plan to employ the proposed scheme in the adaptive dead-reckoning.

7. ACKNOWLEDGMENT

This work was supported by the Grant-In-Aid for Scientific Research (C) of Japan Society for the Promotion of Science under Grant 16560331.

8. REFERENCES

[1] M. A. Srinivasan and C. Basdogan. Haptics in virtual environments: Taxonomy, research status, and challenges. *Computers and Graphics*, 21(4):393–1404, April 1997.

[2] C. Basdogan, C.-H. Ho, M. Slater, and M. A. Srinivasan. The role of haptic communication in shared virtual

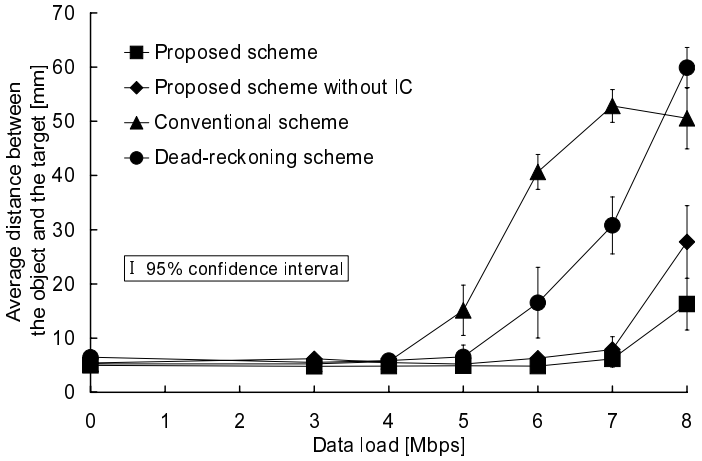

Figure 4: Performance comparison.

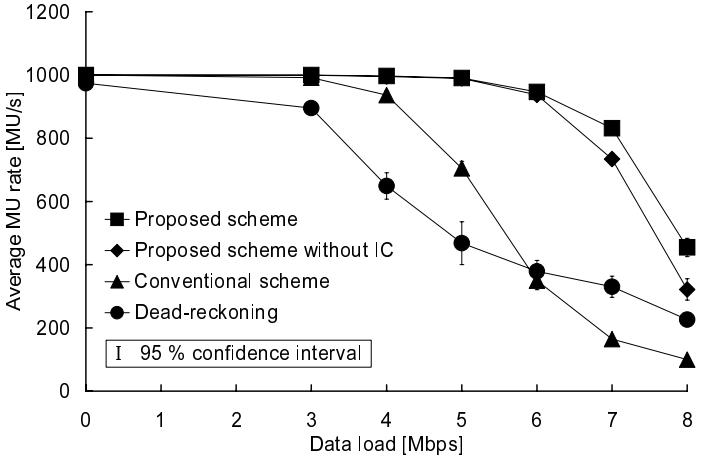

Figure 5: Average MU rates of the four schemes versus data load.

environments. In *Proceedings of the Third PHANToM Users Group Workshop*, December 1998.

[3] Y. Ishibashi and H. Kaneoka. Fainess among game players in networked haptic environments: Influence of network latency. In *Proceedings of IEEE ICME'05*, July 2005.

[4] S. Matsumoto, I. Fukuda, H. Morino, K. Hikichi, K. Sezaki, and Y. Yasuda. The influences of network issues on haptic collaboration in shared virtual environments. In *Proceedings of the Fifth PHANToM Users Group Workshop*, October 2000.

[5] ITU-T Recommendation G.114. Transmission systems and media, general characteristics of international telephone connections and international telephone circuits: One-way transmission time. February 1996.

[6] K. Hikichi, H. Morino, Y. Yasuda, I. Arimoto, and K. Sezaki. The evaluation of adaptive control for haptics collaboration over the Internet. In *Proceedings of the 16th International Workshop on Communications Quality & Reliability (CQR'02)*, pages 218–222. May 2002 (also *Conference Record of IEEE GLOBECOM'02*, November 2002).

[7] Y. Ishibashi, S. Tasaka, and T. Hasegawa. The virtual-time rendering algorithm for haptic media synchronization in networked virtual environments. In *Proceedings of the 16th International Workshop on Communications Quality & Reliability (CQR'02)*, pages 213–217. May 2002.

[8] J. K. Salisbury and M. A. Srinivasan. Phantom-based haptic interaction with virtual objects. *IEEE Computer Graphics and Applications*, 17(5):6–10, September/October 1997.

[9] SensAble Technologies, Inc. GHOST SDK programmer's guide. Version 3.0, 1999.

[10] H. Schulzrinne. RTP profile for audio and video conferences with minimal control. *RFC1890*, January 1996.

[11] Y. Ishibashi, T. Kanbara, and S. Tasaka. Inter-stream synchronization between haptic media and voice in collaborative virtual environments. In *Proceedings of ACM Multimedia'04*, pages 604–611. October 2004.

[12] K. Hikichi, H. Morino, I. Fukuda, S. Matsumoto, K. Sezaki, and Y. Yasuda. Proposal and evaluation of system for haptics collaboration. *IEICE Transactions on Communications (Japanese Edition)*, pages 268–278. March 2003.

[13] T. Kanbara, Y. Ishibashi, and S. Tasaka. Haptic media synchronization control with dead-reckoning in networked virtual environments. In *Proceedings of the 8th World*

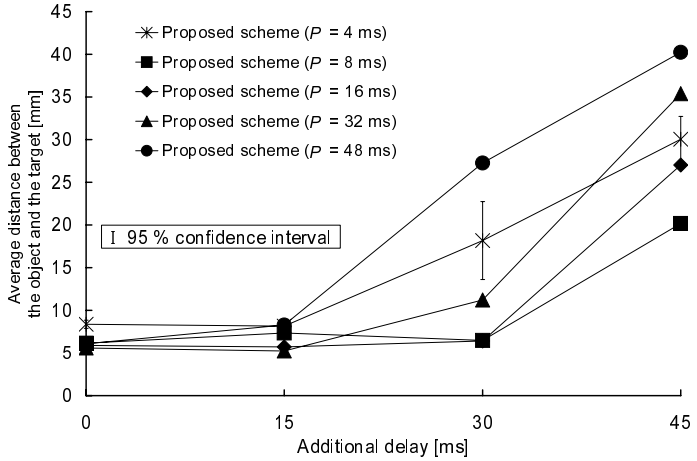

Figure 6: Influences of packetization interval and network latency.

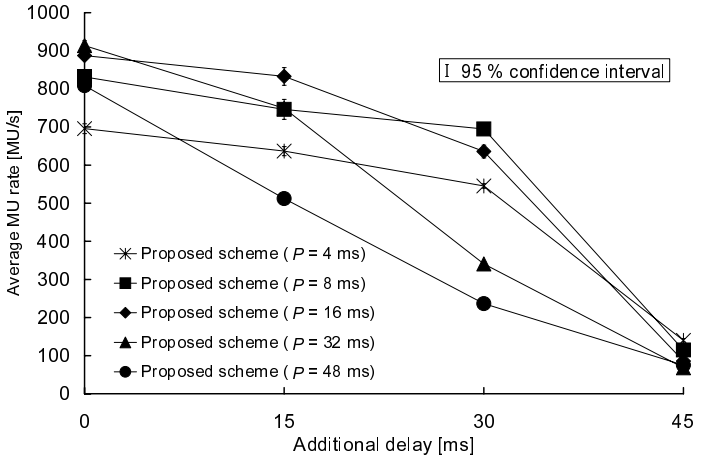

Figure 7: Average MU rate of the proposed scheme versus additional delay.

Multi-Conference on Systemics, Cybernetics and Informatics (SCI'04), vol. III, pages 158–163. July 2004.

[14] S. Singhal and M. Zyda. *Networked virtual environments: Design and implementation*. ACM Press, SIGGRAPH Series, pages 127–144. 1999.

[15] Y. Ishibashi and S. Tasaka. A comparative survey of synchronization algorithms for continuous media in network environments. In *Proceedings of IEEE LCN'00*, pages 337–348. November 2000.

[16] M. Carson and D. Santay. NIST Net – A Linux-based network emulation tool. *ACM SIGCOMM Computer Communications Review*, 33(3):111–126, July 2003.