Design and Evaluation of the Asynchronous Voice Meeting System AVM

Takuya NISHIMOTO, Hidehiro YUKI, Takehiko KAWAHARA, Masahiro ARAKI, and Yasuhisa NIIMI

Abstract

A voice communication system for asynchronous meetings held in non-real time will attract a broad range of users because it is convenient and easy to use with mobile devices. To realize such a system, however, it will be necessary to make possible the natural input of the progressive process of speech at the same time as facilitating the quotation of audio information and outlining of others' references.

This study reports the design and evaluation of a new interface for effective asynchronous voice meetings, the AVM (Asynchronous Voice Meeting System), a client-server type meeting system that uses overlapping speech. Speeches were, in particular, displayed as text information on the client side for the effective visualization of comments made during these voice meetings. The proposed system AVM was compared with an electronic bulletin board system by discussing the same topic. Using the AVM, the total number of speech utterances decreased, and the total amount of letters went down to 48 % compared to the conventional electronic bulletin board system. Subjective evaluations also rated the AVM highly, indicating the usefulness of the proposed system AVM.

Key Words: voice message, asynchronous non-real time media, interactive communication, overlapping speech, barge-in

1 Introduction

Asynchronous, non-real time media, including electronic mail and bulletin board systems, help people communicate whenever they want. These media have several advantages; they have no time restrictions, all the messages are saved, people can take their time to make a reply, a message can be sent to multiple recipients simultaneously, and so on. These advantages encourage many people with various lifestyles to form communities. In addition, searching through the accumulated information makes it easy for people to meet new friends (in the media). We can say that the community forming function is one of the reasons that has made the Internet so popular.

On the other hand, Internet applications with digitized audio and video have been realized these days because there are many improvements in computer performance, multimedia encoding technology, and so on. Especially, when the applications of audio media on the Internet are classified as real time, asynchronous, non-real time, one-way, or two-way, as in Table 1, in order to form communities using

such technologies, asynchronous and bi-directional applications are required. However, though applications such as real-time one-way communication (Internet radio), real-time two-way communication (voice chat, Internet phone), and asynchronous one-way communication (radio archive, music archive) have been proposed, applications for asynchronous two-way communication have not been developed.

Table 1 Applications of audio media on the internet.

	Realtime	Asynchronous
1way	Internet Radio	Radio Archive Music Archive
2way	Voice Chat Internet Phone	(not available)

There are several applications on interactive services for asynchronous audio media, such as HyperAudio [1]. Also, SMIL[(Note 1)] is an Internet standard for describing the layout of sound and animation, synchronization attribute, and hyper-link information. However, though they make the selection of contents interactive, they do not make messages bi-directional.

If community formation is achieved by voice, it will provide user-friendly services that are suitable for mobile devices. Because such services will attract a broad range of users, they will contribute to supporting the social commitment of the elderly and the visually handicapped, and to overcoming the social digital divide (which is caused by the availability and literacy of using the Internet). Voice messages also allow easy identification of the speaker, so it increases the reliability of its content. Consequently, voice messages will give greater security than written messages.

In this study, we proposed a new interface for effective asynchronous voice meetings, and designed and evaluated the client-server type voice meeting system, AVM. We particularly investigated and evaluated the storing, recording and replaying of voice messages.

The organization of this study is as follows. The second section describes the progressive process of conversations. The third section explains our method for the bi-directional voice meeting system. The forth section outlines the design of the proposed system. The fifth section describes the evaluation system. The sixth section discusses the experiment to evaluate the performance of the system. The seventh section states the summary and shows the problems that need to be solved.

2. The Progressive Process of Conversation and Barge-in

Speech synthesis software for electronic mail and web information and e-mail software with voice recognition have already been commercialized. They are providing a better communication environment for unskilled users and visually impaired people. However, in a system where the voice interface has been

added to a communication procedure involving written words, the para-linguistic information such as tone and emotion of the original utterance will not be conveyed. That means the system does not succeed to create a potential fulfillment of verbal communications. Consequently, in this study we designed the proposed system to record and play the voice message itself, and used speech recognition only as a subsidiary measure for browsing and searching.

The difference between the written language and spoken language should also be considered. Spoken language for daily conversations and telephone talks has a progressive process. Speakers often convey only the new information or utter fragments of information as they come to mind. This progressive process allows the elimination of shared information between the speaker and the listener, and uses cognitive expressions with simple syntactic structures and less complicated expressions in conversations [2]. In order to utilize voice messaging effectively in communication systems, it is crucial to design it allowing these progressive processes of conversations.

To realize virtual real-time asynchronous conversations, consideration of proceeding progressive speech is required. Accordingly, we studied conversational phenomena such as overlapping utterances with other speakers and a type of barge-in response that helps a conversation go smoothly. Barge-in is a short response given to the speaker; it allows speech to be interrupted freely at any moment, without waiting for full sentences to be completed.

Previously, many dialog studies have treated barge-in as an exceptional phenomenon. However, we examined the corpus of the RWC project and found out that the majority of the replies overlapped over the speakers' talks and that the overlaps saved about 13 % of the total conversation time [3]. Enomoto, M. et al [4] reports that nearly 45% of the speech overlapped with each other in the Corpus of Map Task Dialog. Kawaguchi, Y. et al [5] explains an aspect of overlaps that 'the responder reasoned and understood what the speaker meant to say,' according to Grice's theory on the connotation of talks.

Several studies on speech overlap focused on the generation procedure of a barge-in. The simplest example of barge-in generation occurs when the machine makes a barge-in after a certain period of silence [6][7]. There is also a proposal for a simulation model for barge-in generation [8] based on pitch patterns. Regarding the role of the barge-in, some studies discussed its effects on the alternation and verification of the speakers [9,10,11]. Another study [12] constructed a spoken dialog system that generates barge-in by processing dialog on real time.

Based on these preceding investigations, we modeled overlapping speech not as exceptional but as common phenomena, and also designed an interface from the standpoint that barge-in has an important role to play.

3 Inter-editing of Voice Messages

Compared with written messages, voice messages have some advantages including being highly realistic, containing various para-linguistic information, and allowing simple input without using keyboards and etc. On the other hand, voice messages are difficult to skim through to search, quote, or annotate previous messages for editing. We present here two propositions to overcome these difficulties.

The first one is to utilize voice recognition for browsing and skimming voice messages. That is, the system appears only to record and replay the human voice, but at the same time it allows users to deal with voice messages as if they were written words. By doing this, we intend to combine the various advantages of voice messages and written words.

We also intend to prove that there is some practical application in our proposition, even if the voice recognition is not perfect itself. This proposition is intended to provide users with the voice recognition result as a means to select certain sections from the dialogs, but not as the ultimate goal. Errors in recognition will not seriously damage its practical use, because users can understand the text by listening to the original speech. A ready-to-use application system with existing technology enlightens system users and opens a new market of voice systems.

The second proposition is the inter-editing of speeches that is different from text editing to provide equal functions. This is from the viewpoint that bi-directional discussions with asynchronous accumulating media cannot exist without the inter-editing of speeches. The users of e-mail and bulletin board systems can read, excerpt, and annotate messages easily. They use two major functions: a) indicating the message replied to, and b) indicating the part focused on in the message. A tree view of messages is an interface for function a), and the partial excerption of a message by adding letters such as '>>' to the utterance is an interface to realize function b). It can be considered an inter-editing of speeches. This is what makes bi-directional discussion and chat possible with asynchronous media.

The simplest means to edit written messages are through computers text-editing functions such as cut and paste. For voice messages, however, a simple interface cannot deliver equivalent functions as written ones. Therefore, based on the discussion in Section 2, we propose a new operating system that provides an active role to overlap speeches and their temporal information. It allows responses to barge in at any time, and uses the information on a) the target of the barge-in, and b) the focused part of the target as the alternative for inter-editing. This operating system should be a reasonable interface in comparison to our usual natural conversations. In addition, the system depends on the user for the message-relating operation of the messages, so the system is not required to understand the contents. It is expected to be an interface suitable for telephone and mobile equipments, because the system requires only an audio input and output device.

4 The Design of the AVM System

Based on the argument above, we designed the Asynchronous Voice Meeting system, AVM. In this section, we describe the details of the system.

4.1 Server-Client Structures

AVM consists of a message server and a client. Users utilize the client to record their messages and the server stores all of the recorded messages. The server also edits multiple messages and sends them as a serial file in real time upon the client's request. The server carries out this editing, and stores each original message recorded by the client.

4.2 Operating Flow

The following is the operating flow from the view of the user:

(1) Select a group to participate in, and get a message list.

(2) Choose a target message and command to play it.

(3) Record a reply to overlap the target message.

(4) Play back the reply, and to cancel it, go back to 3).

(5) Store the recorded reply in the server.

4.3 Communication Protocol and Data Structure

In the AVM system, AVML files, which are information files based on XML(Note 2), and voice files are transferred between the server and the client. The voice file format is uncompressed (11.025 kHz 16 bit sampling, monophonic) at this point, but we plan to compress it in the future for more efficient transferring.

(1) Communication Protocol

The file transmission protocols are GET and PUT methods of HTTP(Note 3) in WWW. Group name and data are expressed on the path element of the URL. For instance, to designate the AVML information of a group name room1, the path is '/room1/text/avml.' In order to designate the message ID, letters like '? Index=1,2,3' are added at the end of the path.

(2) AVML Information Generated by the Server

AVML information generated by the server is utilized by itself as a message list. Also, it is used as additional information accompanying voice messages when an edited file is sent to the client.

An AVML file is composed of several segment entities. A segment indicates a part of a voice file divided into sections along with temporal information. A segment entity indicates the beginning point and end point of the original message that generated the segment. Attributes of a segment entity include mesid (original message ID), sender (original message utterer), mestime (location of the segment's head on the

original message), playtime (location of the segment's head on the edited file), length (segment length), and indent (layer level on the tree view). Segment entity also includes a text entity, which shows the contents of messages and barge-in made by others to the message. Figure 1 shows an example of AVML information generated by the server.

```
<?xml version="1.0" ?>
<avml>
  <segment mesid="1" sender="nishi" playtime="0"
      mestime="0" length="1.5" indent="0">
    <text mesid="1" begin="0.0" end="1.01">
      Good morning!
    </text>
  </segment>
</avml>
```

Fig.1 An example of AVML generated by the server.

(3) AVML Information of the Client

AVML information generated by the client is transmitted to the server to store an utterance, and each segment of a message corresponds to a message entity. It is intended to provide the utterance with the information of the parent message which was played while recording. The attributes of a message entity include parent (parent message ID), reltime (location of the new message head on the parent message), length (new message length), and overlap (if the new message is Aizuchi or not). The overlap attribute is further described in section 4.5. An example of AVML information generated by the client is shown in Figure 2.

```
<?xml version="1.0" ?>
<avml sender="canny">
  <message parent="2" reltime="0.4" length="0.3"
      overlap="1">
    <text begin="0" end="0.3">
      Yes.
    </text>
```

```
    </message>
</avml>
```

Fig.2 An example of AVML generated by the client.

4.4 Recording and Relating of Messages

The AVM client records a new message in full duplex while playing an existing one. The silent part of the recorded message is detected and trimmed automatically. The new message is marked with the relative time to the existing message based on the segment information. The segment information indicates the location of the original message that corresponds to the specific segment edited by the server. When the recorded message is stored by the server, the temporal relation between the original message and the new one can be preserved. Table 2 shows the database structure of the message relations.

Table 2 Data structure of the message relations.

Field	Content
mesid	Message ID
length	Length of the messages (sec)
parent	Parent message ID
offset	Time between the parent message's start and this message's start
memberid	Member ID
wavefile	Audio file name
overlap	Overlap flag
date	Date and time of recording

4.5 BISP Function

The preliminary testing of this system [13] revealed that when recording a new message while playing an existing one, it is necessary to have a proper control on the client side in order to pause a playback. In other words, recording and playing the messages simultaneously often causes difficulties for the user to understand the contents of the playing message, and he or she is forced to replay the existing message. However, pausing the existing message at every utterance even for barge-in will cause the user to hesitate with barge-in even if occurring as a natural action.

Also, regarding editing by the server, it is difficult to understand the contents of long overlapping messages. Therefore, if the message is long, it should be inserted into the parent message rather than overlapped. On the other hand, for barge-in and shorter messages, they should be overlapped onto the parent message. Hence, the reproduced conversation will become more natural and easier to understand.

Consequently, we designed the server to categorize an utterance according to the overlap attribute

either as a mere barge-in (backchannel utterance) or as a usual conversation before storing it. Based on this information, the server edits barge-in to overlap and usual conversation to barge into the parent message in order to reproduce virtual conversations. The server realizes such processes by following the algorithm below:

(1) Find a root message in the ID list received from the client.

(2) Retrieve the child messages of the root message and select only dialog, non-barge-in messages, and insert them without overlapping recursively into corresponding location of the parent messages. At the same time, generate the corresponding segment information.

(3) After inserting all barge-in messages into the parent messages, overlap barge-in messages onto the conversation. Choose the overlapping points for the barge-in messages by using the segment information, as barge-in messages are managed in relative time to the parents.

In addition, to record a new utterance at the client side, whether it is a barge-in or not is determined by the speech length. A short utterance is treated as a barge-in and recorded with no pausing of the replay and for a speech longer than a certain time (currently 1.0 second), the existing utterance is paused until the end of it, and is stored as a non-barge-in. This function is called BISP (Barge-in to Stop Playing). With BISP, spontaneously uttered barge-in becomes reproducible and existing speeches are no longer missed even if a long speech is given while playing. The client's transition state with BISP is shown in Figure 3.

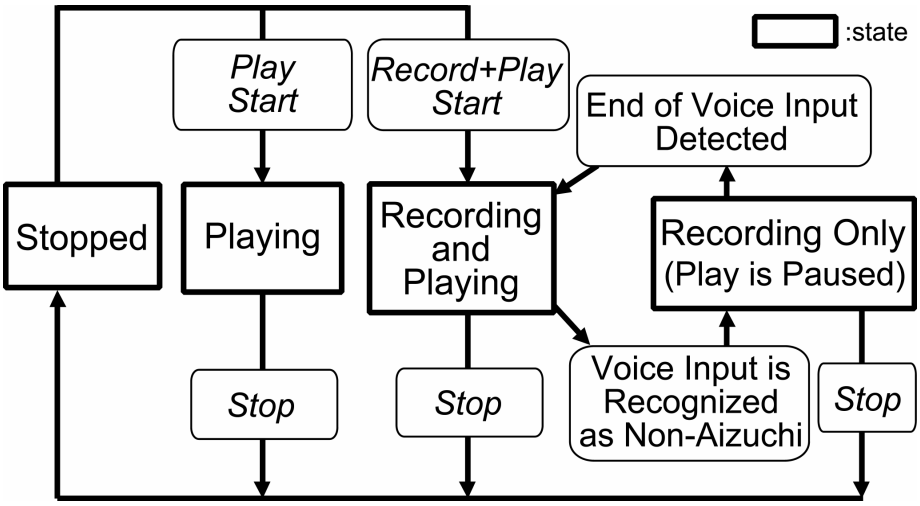

Fig.3 State transision chart of the client.

4.6 Managing Read/Unread Messages

When many messages accumulate in one group, it becomes difficult to understand which remarks the user has already listened to. To eliminate this, the server is designed to manage the IDs of previously

heard messages and to mark them respectively in the list for each user on the client software. This helps users to distinguish the already heard messages in a group, after storing many messages.

5. Construction of the Evaluation System

Based on the design in the fourth section, we constructed a system for evaluation such as the one shown below.

5.1 The Server

Voxer, the AVM server, is written with the Perl and C languages in order to make it portable, and processes HTTP requests from the client software. It has a database function to store information, including voice files and relations between messages, and an editing function for reproduced speech. Voxer runs on Windows for the following experiment, though it can be used on Linux. Speech recognition is integrated to the server but was not used this time.

5.2 The Client

Voyager, the AVM client software, is implemented with Microsoft Visual C++, which has full-duplex audio input-output. This software runs on Windows98. It has the functions of HTTP message transmission, voice recording and playing including BISP function, and displaying messages as the tree-view. To prevent the playing message from being recorded in full duplex, a headset is used for recording and playing. Figure 4 shows the screen image of the client. The left pane is tree-view for message selection. The right pane displays the message contents based on text entity (see 4.3) generated by speech recognition. The upper tool-bar has buttons for “play and record”, “play”, “stop”, etc. , and the lower gauge shows the level of input from the microphone.

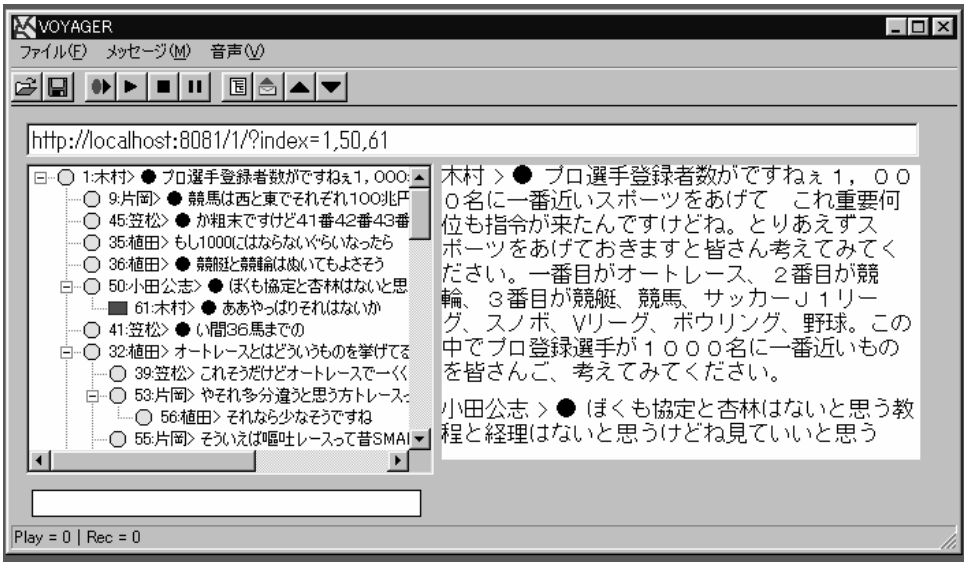

Fig.4 Screen image of the client.

6. Experiment

6.1 Experiment Method

The performance of the AVM system was evaluated against the conventional electronic bulletin board system (BBS) by discussing the same topic: ‘Which sport has registered professional players of close to 1000 people: Auto racing, boat racing, horse racing (The Central Horse Racing Association), J1 league of soccer, snowboarding, V league of volleyball, boxing, bowling, or baseball?”

Usually with this type of topic, the participants can complement their own limited information among themselves, while no one covers everything, and it can be expected that interactive discussion will lead the group closer to the right answer. Activity of the discussion can be evaluated by comparing the conclusion with the right answer, and also the system can be quantitatively analyzed by the number of accesses by the users and the number of utterances etc. in the discussion. This experiment was planned from these viewpoints.

To encourage serious discussion, the subjects were notified that the team with the closest conclusion to the correct answer was to be paid a higher amount. As the AVM system, the client software and the server system were implemented on the same computer with Windows98. For the BBS software, WebForum[(Note 4)] was selected. This software runs on the WWW server and can manage the written utterances with a tree structure.

All ten subjects were male students in their twenties from university scientific laboratories (including speech research), and were well-experienced with keyboard operation. They were divided into two groups of five, one for AVM and the other for BBS. The chairperson of each group gave the topic through the system. The users were subsequently chosen randomly to use the system in turn.

Regarding the speech recognition of AVM, the operator repeated the utterance to ViaVoice98 (a speech recognition software from IBM Japan) after every user finished, then he stored the recognized sentences in the server. This allowed the speech recognition to maintain practical level of quality as well as to include a certain level of misrecognition in the obtained text information.

When all the team members agreed on the conclusion, the chairperson orally answered the operator. At the end, all subjects were asked to fill in a questionnaire.

6.2 Result

The winner was the AVM team, although second team reached close to the correct answer. The quantitative analysis of the utterance is shown in Table 3, and the comparison of the lengths of the

messages is in Figure 5. The lengths of the messages on AVM were counted using the manual transcription of voice messages after the experiment. In regards to the utterances on BBS, the number of characters were counted excluding the partial excerption.

ViaVoice98 showed 85.2 % word recognition rate for correctness and 82.5% for accuracy.

The average of the result of the questionnaire is shown in Table 4 (with a five-grade system, including “do not think so at all” as one, and “think so very much ” as five.) The questions, (k) and (l), were asked only to the AVM users.

Table 3 Analysis of the messages on AVM and BBS.

	AVM	BBS
Total count of using the system	20	24
Number of the messages	71	24
Number of the characters of all messages (excerptions are excluded)	2303	5657
Average number of characters of a message	32.4	235.7
Total time of messages (sec)	501.3	-
Average time of a message (sec)	7.1	-
Number of the overlapped messages	12	-
Number of the non-overlapped messages	59	-

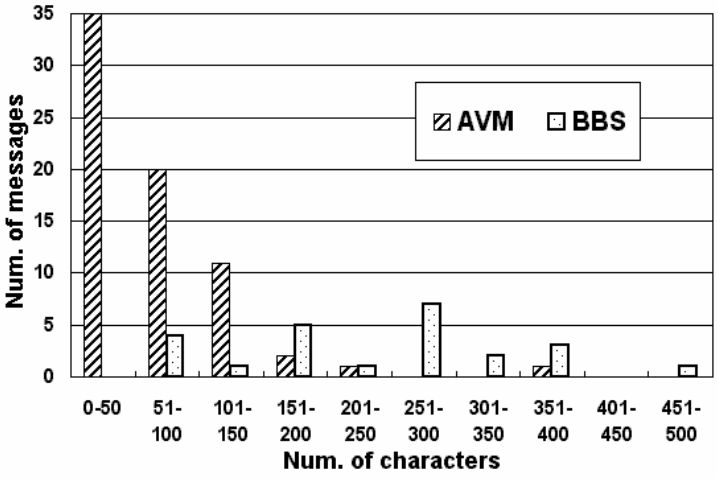

Fig.5 Length of the messages on AVM and BBS.

Table 4 Result of questionnaire.

Items	AVM	BBS
(a) I obtained satisfactory answers	4.4	3.6
(b) I was able to adequately speak my opinions	4.0	4.4
(c) Active discussion was able to be conducted	3.8	3.4
(d) The discussion proceeded in a natural manner	4.0	4.0

(e) The mood was very similar to when people actually congregate to conduct discussions	2.2	2.8
(f) It was easy to expound the pros and cons	4.4	3.8
(g) Easy to input messages	3.4	4.2
(h) Easy to grasp the content of messages	3.4	4.0
(i) Easy to view the discussions of the entire conference	3.4	3.8
(j) The user interface was simple to understand	3.8	4.0
(k) I used the text display more than the voice play	4.0	-
(l) Using the text, including the misrecognitions, was valuable	4.6	-

6.3 Discussion

We shall discuss the result in view of the following hypotheses:

(1) AVM realized an asynchronous voice meeting like spoken dialogs.

(2) AVM made use of the progressive process of spoken words.

(3) BISP function worked well and contributed to smooth conversation.

(4) The speech recognition result including misrecognitions was efficiently used.

(5) AVM functioned sufficiently as an alternative to BBS.

We will first discuss Hypothesis 1. With AVM, the subjects mostly spoke with conversational and casual expressions such as “Well, talking about soccer now… about 20 teams in a league. And to make 1000 players, 50 for one team. There aren’t so many on a team, are there?” On the contrary, with BBS, the participants mainly utilized written language, for example, “In my interpretation, as others are saying, there should be more than 84 players (1000 ÷ 12 = 84) in the professional baseball league, including the farm teams. However, there is also a limit for the number of registered players for one team, so …” The relations among the team members cannot cause such a difference because all the subjects knew each other well. Accordingly, the result supports Hypothesis 1.

Hypothesis 2 predicts the elimination of shared information, which causes the frequent utilization of simple and fragmented expressions as stated in the Section 2. Several results supported this hypothesis, including the fact that the total length of messages with AVM was 48 % of that of BBS (Table 3), that the length of most messages were relatively short (Figure 5), and that many utterances depended on the context such as “Well, it’s close, as you said.”

Hypothesis 3 was only partially supported. Many of the overlapping utterances, which are 17 % of the total speech, were fragmented because of unsuccessful detection of the speech segment. This also caused the low rating of question (g) in Table 4, indicating insufficient processing of overlapping speeches. However, the high rating of questions (a) to (d) in Table 4 suggests that smooth conversation for this experiment was realized.

Hypothesis 4 was supported by the results of the questionnaire, including high evaluation of questions (k), (l) in Table 4, subjects' comments such as, 'I confirmed the context roughly by the text and minutely by the replay,' and, 'The text information saved time from listening to various utterances.'

Hypothesis 5 was supported by the facts that stated in Table 4, AVM was not rated far lower than BBS, and that both teams reached acceptable answers. However, rating of the question (e) for AVM was as low as BBS, which indicates that AVM cannot replace face-to-face discussion yet.

Accordingly, the experiment results support hypothesis 1, 2, 4, and 5 completely, and hypothesis 3 partially.

7. Conclusion

In this report, we designed and evaluated AVM, an asynchronous, non-real time, and interactive voice meeting system. We were able to confirm that AVM applies the advantages of voice messaging in that it is highly expressive, easy to talk, and does not impair the merits of asynchronous communication of electronic mail, bulletin board systems, etc.

In the future, we intend to advance the performance of speech recognition in the server particularly for spoken language. Improvements for smoother conversation are also required, including avoiding redundant replaying of utterances already heard, and preventing the fragmenting of messages in the middle of a word. In addition, large-scale operating tests with various users, such as people unfamiliar to the keyboard, are necessary. We plan to further investigate the realization of an AVM application on telephony services and handheld devices, to attract a wider range of users.

(Note 1): Synchronized Multimedia, http://www.w3.org/AudioVideo/

(Note 2): Extensible Markup Language, http://www.w3.org/XML/

(Note 3): Hypertext Transfer Protocol HTTP/1.1, RFC2068.

(Note 4): http://www.kent-web.com

References

[1] M. J. Hirayama, T. Sugahara, Z. Peng, J. Yamazaki: Interactive listening to structured speech content on the Internet, Proceedings of ICSLP'98, pp.1627-1630, Dec. 1998.

[2] M. Okada: The mumbling computer, Kyoritsu Shuppan, Tokyo, 1995 (in Japanese).

[3] T. Nishimoto, Y. Niimi: An asynchronous voice messaging system, IEICE Technical Report, MVE97-98, pp.39-46, Feb. 1998 (in Japanese).

[4] M. Enomoto and S. Tutiya: Overlapping and it's statistical analysis in the Japanese map task dialogue corpus, IPSJ SIG Notes, 99-SLP-29-25, pp.145-150, Dec. 1999 (in Japanese).

[5] Y. Kawaguchi, S. Tutiya: On those aberrations from the turn-taking rules which can be accounted for by conversational implicature, IPSJ SIG Notes, 99-SLP-29-26, pp.151--156, Dec. 1999 (in Japanese).

[6] H. Nishi, K. Gomi, J. Kojima: Stochastic speech-end detection for man-machine speech communication, Transactions of the IEICE, D, Vol.J70-D, No.11, pp.2108-2114, Nov. 1987 (In Japanese).

[7] C. Kougo, J. Yamauchi: An experimental answering machine with backchannel responses, The 8th annual meeting of Japanese Cognitive Science Society, pp.72-73, Jul. 1991 (In Japanese).

[8] N. Ward: Using prosodic clues to decide when to produce back-channel utterances, Proceedings of ICSLP'96, pp.1728-1731, Oct. 1996.

[9] H. Kikuchi, Y. Sugita, K. Shirai: Analysis of influence of time constraint in natural dialogue, JSAI SIG Notes, SIG-SLUD-9702-5, pp.31-36, Oct. 1997 (In Japanese).

[10] Y. Horiuchi, H. Koiso, S. Tutiya and A. Ichikawa: Some utterance-final syntactic and prosodic characteristics relevant to the control of the speaker shift phenomena in spontaneous spoken dialogues, IPSJ SIG Notes, 96-SLP-10-9, pp.45--50, Feb. 1996 (In Japanese).

[11] Y. Okato, Keiji Kato, M. Yamamoto and S. Itahashi: Evaluation of the dialog systems with back-channel feedbacks, JSAI SIG Notes, SIG-SLUD-9804-2, pp.7-12, Feb. 1999 (in Japanese).

[12] J. Hirasawa, N. Miyazaki, M. Nakano, T. Kawabata: Implementation of coordinative nodding behavior on spoken dialog systems, Proceedings of ICSLP'98, pp.2347-2350, Dec. 1998.

[13] T. Nishimoto, H. Yuki, T. Kawahara and Y. Niimi: An asynchronous virtual meeting system for bi-directional speech dialog, Proceedings of Eurospeech'99, pp.2471-2474, Sep. 1999.

Profiles

Takuya Nishimoto

He received B.E. and M.E. from Waseda University, Japan in 1993 and 1995, respectively.

He is a Research Associate at Department of Electronics and Information Science, Kyoto Institute of Technology, Japan. His research interests include spoken dialogue systems and human-machine interfaces.

He is a member of Information Processing Society of Japan, the Acoustical Society of Japan, the Japan Society for Artificial Intelligence, and Human Interface Society.

Hidehiro Yuki

He received B.E. and M.E. from Kyoto Institute of Technology, Japan in 1998 and 2000, respectively.

Takehiko Kawahara

He received B.E. and M.E. from Kyoto Institute of Technology, Japan in 1999 and 2001, respectively.

Masahiro Araki

He received M.E. in information science and D. Eng. from Kyoto University, Kyoto, Japan, in 1990 and 1998, respectively. He is an Associate Professor at Department of Electronics and Information Science, Kyoto Institute of Technology. His research interests include spoken dialogue systems and natural language understanding.
He is a member of IPSJ, JSAI, ANLP, and ACL.

Yasuhisa Niimi

He received the B.E., the M.E. and the Ph.D. degrees from Kyoto University, Japan in 1962, 1964 and 1969, respectively. From 1964 to 1969 he was working at Kyoto University. Since 1970 he has been working at the Department of Electronics and Information Science of Kyoto Institute of Technology, where he is now a professor. His current interests include speech information processing, natural language and artificial intelligence. He published the bool “Speech Recognition” (in Japanese) in 1979. Dr. Niimi is a member of the Information Processing Society of Japan, the Acoustical Society of Japan, and the Japan Society for Artificial Intelligence.