# Multi-Context Voice Communication Controlled By Using An Auditory Virtual Space

Yasusi Kanada
Central Research Laboratory, Hitachi, Ltd.
Higashi-Koigakubo 1-280, Kokubunji, Tokyo 185-8601, Japan
kanada@crl.hitachi.co.jp

**ABSTRACT**

A new voice communication medium, which the author calls "voiscape", will probably appear in near future. Voiscape shall have much improved user interface than the conventional voice communication systems, i.e., telephone and conference systems, and be based on the IP-based conferencing and spatial audio technologies. The author has developed a prototype toward voiscape, which has made a step toward solving two problems of the conventional systems i.e., complicated and restricted conference control and lack of crossed-over multi-context support, by introducing two features. The first function is the virtual-location based communication; i.e., the users can talk with other users and move, in a way similar to face-to-face conversation, in a virtual auditory space created by spatial audio technology without explicit session and floor control. The second function is personalized policy-based communication control; i.e., the users can specify communication policies that protects their privacy and reduce required resources. This function is enabled by a distributed policy-arbitration mechanism. Experiments showed that the basic mechanisms and the policy-based control with a simple policy worked well.

**KEYWORDS**

Conference control, floor control, multi-context conferencing, conference policy, IP telephony, virtual space.

## 1. Introduction

The telephone has been in use for about 130 years since A. G. Bell invented it. It is still very important as a human-to-human communication medium. However, the user interface of the telephone has not been much improved since its invention; i.e., to talk by telephone, you must first call the person you want to talk and connect the line, then talk to that person one-to-one (i.e., you can not talk or listen to two persons at once) by using one microphone and *one* speaker, and disconnect the line when you finished. This interface has many drawbacks. For example, it cannot utilize a person's *two* ears that play important roles in direct communications, or the human aural ability with two ears, so it is difficult to use it for multi-user conferencing, and a person may be called when it is most inconvenient because no presence information is propagated while the line is not connected. This means that the telephone has created a very unnatural communication environment that is completely different from real-world face-to-face communication. One reason why this unnatural interface has not been changed is that it has become widely accepted by people. However, the main reason is probably that telephone networks are hard-wired and restricted, especially in the case of line-exchange networks in which continuous connection is not possible, so it has been difficult to change such an interface.

However, telephone networks are going to be replaced by IP networks. This replacement will eliminate the restrictions that have prevented changes in the user interface. It will enable continuous connection (without disconnection): an important feature of an IP (packet-exchange) network. It will also enable communication similar to face-to-face conversation, utilization of the human aural ability, and flexible multi-user conferencing. The author calls this new medium "voiscape" [1, 2] because it should realize a natural soundscape of human voices. The technologies, including conference control and spatial audio technologies, that are expected to be used in voiscape are being developed now.

This paper focuses on a method for session and floor control and multi-context conferencing in small- to medium-sized conferences that should be applied in a voiscape environment. The problems of conference control and multi-context conferencing are explained in Section 2. In Section 3, a solution to solve these problems is explained. The outline of human communication that the author believe voiscape should realize is described in Section 4. The implementation of the functions described in Section 4 is explained in Section 5. The solutions are evaluated in Section 6, and this paper is concluded in Section 7.

## 2. Conference Control and Multi-Context Problems

Two problems of conventional conferencing systems are explained below.

### 2.1 Complicated and restricted conference control problem

In this paper, "conference control" [3] means the whole control and management concerning conferencing systems. Conference control contains four components: (1) room management, (2) user management, (3) session control, and (4) floor control. The room management means creation, destruction, or feature modification of a conference room.[1] A conference room means a virtual place that enables audio and/or video conferences.[2] User management adds or deletes a user to the member list for a conference room or modifies the properties (rights) of a member. These additions or deletions do not really add or delete a user to or from conferences; the session control really adds or deletes a member to or from the conference room. The floor control [4] controls shared conference resources such as the right to speak or the right to change a shared file or camera. This paper focuses on the session and floor controls.

The session control function of a conferencing system is usually performed by a session-control protocol such as SIP (Session Initiation Protocol) [5]. There is currently no widely used protocol for floor control. However, Handley,

[1] Although there are many models for conferencing [3], this paper focuses on one model that covers most applications.
[2] A conference may be identical to the conference room in certain types of conferencing systems.

et. al. [6] developed a protocol called CCCP (the Conference Control Channel Protocol), and the IETF (Internet Engineering Task Force), in particular, the XCON Working Group, is currently working on the standardization [7].

Although the session- or floor-control function is performed by the conferencing system, it is necessary to be usually controlled by a person, i.e., a user or a manager. The simplest session-control functions are addition or deletion of a conference member as described above; they are invoked by the user's join or leave operations. However, there is usually only one and binary way to join or leave a conference. In a direct communication (i.e., a real-world communication without electronic media), there are many and continuous ways to join or leave a chat or meeting. For example, one can (continuously) walk up to a person before beginning to talk, or call a person from a distant place.

In addition, there are many more types of session and floor-control functions, such as merging or splitting conferences, although each conferencing system implements a limited set of such functions. These functions may also be related to the room and user management, and they require much more complicated control both by people and the conferencing system. They also have many restrictions on the people; it is difficult for people to satisfy their desire concerning personal preferences or privacy needs, and difficult to realize social behaviors which are very important in the real world. Most of the session and floor control functions are not generally applicable and do not have a wide range of applications. It is thus difficult to replace poor but general-purpose communication media such as the telephone by a conference system. In a face-to-face communication, there are many ways to reorganize a meeting or chat by using simple and general-purpose methods such as gathering several persons in one place.

### 2.2 Multi-context conferencing problem

In most current IP telephone and conferencing systems, a conference allows only one context, i.e., two persons cannot hear two or more different communication streams at the same time. However, in a direct communication, we can often observe multi-context conferencing, especially in medium- to large-scale meetings; that is, the participants talk locally, apart from the global meeting context. This type of talk is often very important for the participants.

It is very important not just to coexist but to be able to crossover multiple contexts. To crossover means that a user can join two or more contexts at once and can transfer information from one context to another. Merging and separating contexts, which should also be types of crossovers in a wider sense, should also be possible. Such context crossovers are important because a multi-context conference without such a mechanism is equivalent to multiple conference rooms without multi-context mechanisms. Such separate contexts exist in conventional systems.

There are two causes that make multi-context conferencing (with crossover) difficult in conventional systems. One cause is that all the participants hear the same voices; i.e., the (relative) volumes and other properties of voices in a conference are shared by all the participants. The other cause is that the cocktail-party effect [8, 9] is not utilized because the relative direction and distance among the participants, which enables people to distinguish the sources of mixed voices, cannot be expressed in conventional conferencing systems.

In conventional conference systems, a "conference within a conference" can be created and is called a side conversations [10] or sidebar [11]. However, a side conference needs to be explicitly controlled by participants, and the sidebar context is separated from the main context of the conference.

## 3. A Solution to Multi-Context and Conference Control Problems

The author introduced two features to solve the above two problems. The features are virtual-location-based communication and personalized policy-based communication control, which the author believes that voiscape should have.

### 3.1 Virtual-location based communication

In conventional conferencing systems, participants have no ways, or have very limited ways, for expressing their intention and desire regarding who and how they want to listen or talk except the above-explained floor and session control methods. It is important for human users to be able to express their intentions and desires by a communication system. This is because human communication is not merely transmission of communication content; people satisfy their desire by communication and the intensions of both the speaker and listener are important for understanding the conversation. Ringing a bell may be a method of expressing the caller's intention and desire. However, it is not necessarily interpreted so because it is the only way to start a conversation by telephone. The lack of such methods may cause users' frustration.

In face-to-face communications, space and location are general-purpose means that can be used for expressing intentions and desires. For example, if a person want to talk to another person, he can walk up to her, i.e., he can shorten the distance between them. She can probably see his intention to talk or listen. This can be simulated by using virtual-space technologies, i.e., 3-D graphics technologies and spatial audio technologies [12]. A pointing device such as a mouse or cursor keys can be used for moving around the virtual space that is not usually bound to the real location. This virtual-location-based communication concept can be illustrated as in **Figure 1**. The users can talk to each other in a shared virtual graphics or auditory space. It is less easy for a user to see the intention and desire through a communication medium than through direct communication environments, but it is still probably possible because there are many ways to start a conversation in the virtual space.

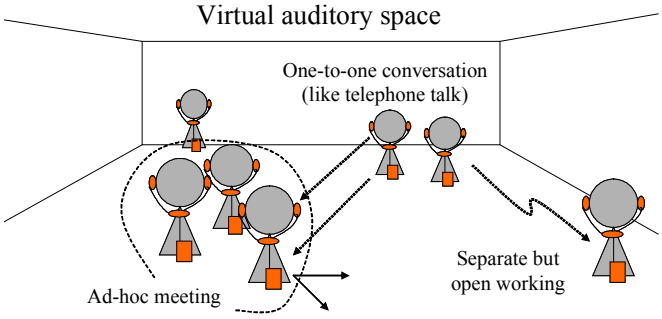


Figure 1. An image of virtual-location-based communication

Graphical-virtual-space-based communication has been studied by many researchers [13, 14, 15, 16, 17]. In the case of MASSIVE [14], the desire is expressed as *nimbus*, which is a range of cognition. Graphical virtual spaces have been often used for entertainment such as games; however, they

have not been commonly used for real-world communications.

There have also been studies of spatial-audio-based communication [18, 19, 20, 21]. A variety of interactive virtual auditory environments were developed [22]. Especially, Savioja [23] enabled an auditory environment in which the user can quickly move around. However, most of them were focused on re-creating room acoustics or human-to-object interactions, but not on human-to-human voice communication in the virtual auditory space. The possibility of virtual-location-based communication using spatial audio technologies has been less extensively studied than that only using 3-D graphics and/or video. This paper focuses on spatial audio because it can be used more easily than 3-D graphics while performing other tasks such as office work or walking.

The following characteristics of spatial audio technologies enable multi-context communication. First, because sounds attenuate according to distance, people at close range can talk to each other even if conversations in different contexts are going on at more distant places. The other conversations are still audible; thus, it is possible to crossover the contexts. Second, because of the cocktail-party effect enabled by spatial audio, a person can switch to a context in an environment in which many other conversations are going at similar volume levels.

The application of spatial audio technologies also partially solve conference-control problems. People can choose a context by themselves without any special control; i.e., there is no need for explicit session and floor controls. Such implicit floor control methods were developed in FreeWalk1, which was a video conference environment, and analyzed by Nakanishi [15]. The present study developed such methods in a spatial-audio-centered environment.

### 3.2 Personalized policy-based communication control

Although location-based communication eliminates explicit floor controls, there are two reasons that some type of automatic conference-control mechanism is required. One reason is that communication resources such as network bandwidth or packet buffers must be controlled because they are limited resources. If all the people in a large conference room send their voice to all the other people, too much network bandwidth will be consumed.

The other reason is that unlimited transmission of voices causes privacy problems. For example, in a real-world conversation, if the number of people in a room is large, a person can hear only a limited number of voices. They cannot hear a voice of a person who is on the other side of the room. However, if voices are transmitted unlimitedly by the network, one can choose and hear any voice in the virtual conference room. People would not feel comfortable while talking in such a room.

A method of automatic conference control should therefore be developed. Each person has their own needs for privacy; she can choose a level of privacy from many levels or continuous levels. Accordingly, the author has developed a policy-based communication-control method to solve the above problems. With this method, each user has a set of policies for controlling voice communications so that the user's privacy and resources can be managed. A user can write a policy such as follows.

- If another user comes within 5 m from the user in the virtual space, connect to that user (bi-directionally), and if another user goes over 6 m from the user, disconnect that user (both directions).

This policy assures that your voice cannot be heard by any persons 6 m or more away from you.[1] This policy requires that the voice streams of both directions must be symmetric, i.e., the communication conditions must be the same. You can also write a policy stating that a distant (e.g., 10 m apart) person can hear your voice but cannot understand what you say (because your speech is replaced by the meaningless voice of a user agent in your terminal).

Other users also have their own policies. If a user has the same policy as yours, communication starts and stops as described above. However, if the policies are different but applicable to the current condition, they must be arbitrated, because both policies specify the voice streams of both directions. A policy should specify both directions and should usually be symmetric because the user privacy must be protected. If otherwise, for example, another user who is unrecognizable by the user can hear the voice of the user.

In general, the arbitration is not easy; if the other user and you specify different types of policies that are contradictory, it is very difficult to decide what to do. However, if the policies are just different in strength, the arbitration can be automated. For example, if both are distance-based connection and disconnection policies such as above but the distances are different, a stronger policy (i.e., a policy that protects privacy more strongly) should be applied. For example, if one policy specifies 5 m as the connection distance and the other specifies 4 m, the latter should be taken because it satisfies the privacy requirements of both sides.

The conference system can also have its own policies that manage the system and network resources. The user policies and the system policies can be arbitrated by a decentralized method such as above.

A protocol for carrying conference policies, which is called CPCP (Conference Policy Control Protocol), is discussed in the XCON and SIPPING WGs of the IETF [11]. CPCP can carry policies of a participant from him to the conference system. However, currently it is intended to carry policies that decide how he joins the conference but not intended to carry policies that decide how other people join the conference as described above.

## 4. Outline of Communication Using Voiscape

An outline of a typical sequence of a conversation using the voiscape is as follows. When a user turns on the terminal and invokes the user agent, the agent logs in to the server automatically, and it displays a list of rooms available for the user. When the user selects (enters into) one of them, voice communication with the users in the same room starts, and the room is displayed by sounds and graphics. The user can move freely in the room and rotate himself by using a pointing device. Because spatial audio is used, the voice of a speaker in the room will become louder if the user becomes closer to a speaker, and the direction of the voice is changed if the user rotates. This sequence is explained in more detail below.

The user agent holds the user identifier, which it uses to log into the server. The user-agent window in the prototype is shown in **Figure 3,** for example.

After the user has logged in, the server sends a room list to the terminal. The list box in the upper-left corner in Figure 3 shows the list. There are four room names: Office, Project-X, Meeting room, and Home. The user selects a

[1] The disconnection distance (6 m here) must be larger than the connection distance (5 m here) because otherwise the connection can be unstable at that distance.

room from the room list, and enters the room. The user can enter only one room at a time. The user can also exit from any of the rooms and end the communication.

When the user chooses a room and enters it, they can see who is in the room. Voice communication with the other users in the room starts automatically, and the other users are displayed by spatial audio in the virtual space (i.e., the room); spatialized voices are propagated to both of the user's ears through the headset.

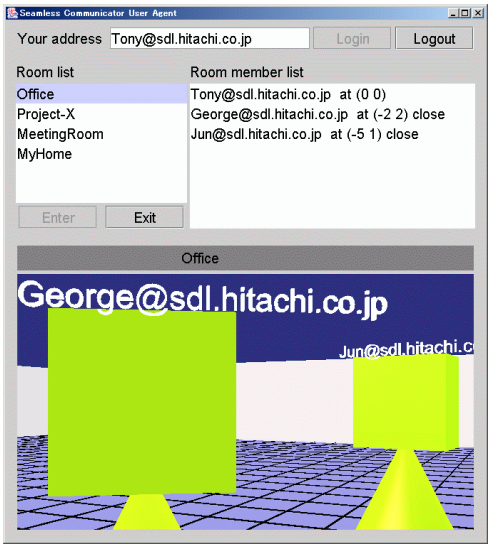


Figure 3. User-agent window in the prototype

In addition to auditory display, the environment around the user should be displayed by 2-D or 3-D graphics because it is difficult for people to exactly grasp the direction of a speaker by voice only, especially the distinction between in front and behind voices and that of above and below voices is vague. In our prototype, the view in the forward direction is displayed by 3-D graphics. The view does not contain the user himself, and the room is expressed by a floor and walls. Because the only communication medium between the users is voice, this view does not contain another user's video image, but another user is expressed by an avatar expressed by a rectangle and a cone as shown in Figure 3. The user identifier is shown above the rectangle, and the direction of the other user can be seen by the direction of the avatar.

The user can move and rotate by using a pointing device. This motion and rotation is in the virtual space, so it has no relationship to the location and direction in the real world. The pointing device moves and rotates the user himself; it does not move the room or the other users. In the prototype, a mouse is used as the pointing device; the user can move himself forward by moving the mouse forward and backward by moving it backward, and he can rotate himself to the left by moving the mouse to the left and to the right by moving it to the right. Motion and rotation can be realized by other types of pointing-device operations. The reason why forward/backward motion and left/right turn is used is that when people move in the real world, they do so by walking forward or backward and by changing direction.[1] This interface is similar to that of Digital Space Traveler (http://www.digitalspace.com/traveler/), which is a successor of OnLive Travelor [17].

The graphics is updated every time the user moves or rotates, because the change of location and direction should be immediately fed back to the user. However, the updating frequency for sending the change to the server and other users may have to be reduced to decrease the network traffic and processing time for the server and other user agents.

## 5. Implementation

A prototype for voiscape, with the two features described in Section 2, were developed. The outline of the architecture, the room, location and presence management method, and the session and floor control methods are explained below.

### 5.1 Outline

The architecture of the prototype is explained in **Figure 4**. The prototype consists of terminals and servers. Each terminal contains a user agent for voiscape. Currently, desktop PCs with Microsoft Windows XP and an inexpensive sound card with HRTF (Head-Related Transfer Function) [12][2] are used for the terminals, although mobile terminals will be used in the future because voiscape will probably replace mobile phones and utilize the advantages of the full-time connection feature of IP networks. The structure of a terminal is shown in **Figure 5**. Most parts of the system, including the user agent and the servers, are coded in Java. However, detailed explanation is omitted here.

The SIP proxy server and the room server have been developed, but no authentication or authorization functions have been developed yet. SIP is used for controlling voice-communication sessions between the user agents. The SIP proxy relays SIP messages and receives registration messages (SIP REGISTER) from the terminals and manages the locations (IP addresses) of the users (i.e., the SIP proxy contains a registrar). NIST SIP stacks [24], which are written in Java and are an implementation of JAIN SIP [25], are used throughout the system, and an NIST SIP proxy [24] is used for the proxy. (The SIP messaging is explained in Session 5.3.) The room server manages the virtual conference rooms and their users. (The function of the room server is explained in detail in Session 5.2.)

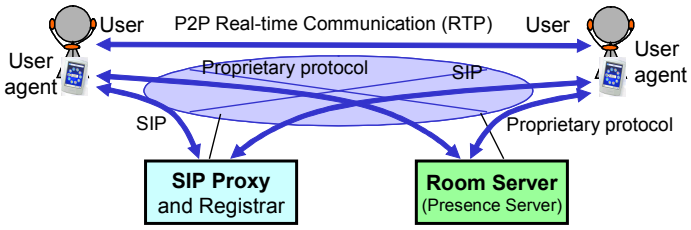


Figure 4. Architecture of the prototype

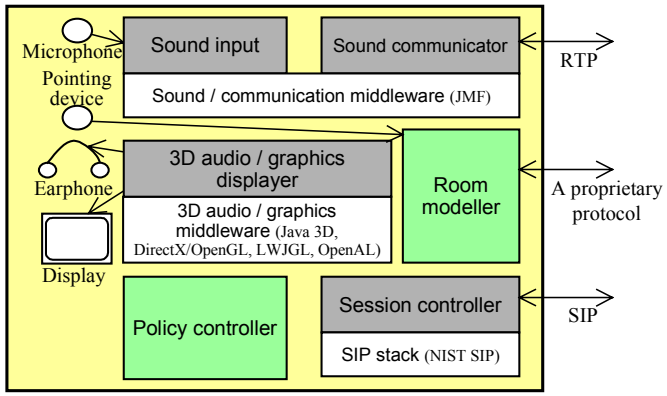


Figure 5. Structure of the terminal[3]

[1] It is unusual, for example, to move like a crab; i.e., to move left or right without changing the direction.

[2] We used a sound card based on C-Media's CMI-8738 chip. This chip works with Sensaura's HRTF library.

[3] JMF (Java Media Framework) and Java 3D are optional packages of Java

Voice streams are transmitted peer-to-peer by using RTP (Real-time Transmission Protocol) [26]. The sessions between two user agents are managed one-by-one (pairwise), and the VoIP data sent to one user can be different from the data sent to another user because the privacy conditions may be different.[1] Unicast, not multicast is used for the voice communication.

### 5.2 Location and presence management

The room server performs the room management and part of the user management. It communicates with the user agents. It receives the location and direction of each user and returns the collected locations and directions of other users to the user. In the current implementation, a polling-based proprietary protocol is used for the communication between the room server and the user agents. This is a request-and-response-type protocol. The requests are sent from a user agent to the room server. The basic request message types are as follows.

- **Room add**: This type of message is used for notification of entering or creating a room. If a specified room exists, it is used, but if it does not exist, a new room is created.[2]
- **Room remove**: This type of message is used for notification that a room exists. The room continues to exist even after all the users have exited.
- **Presence refresh**: This type of message is used for notifying the sender's (user's) location and direction in the room to the server and for obtaining the list of rooms and other users' presence in a room (including their location and direction). If the sender is not in any room, the response does not contain other users' presence information.
- **Room destroy**: This type of message is used for destroying a room.

This protocol is Java-based. Reliability was not a key concern in the design of this protocol, so it is easy to generate zombies with this protocol because rooms and users are hard-state objects. It is planned to re-implement a more reliable protocol for this communication by using the event-notification mechanism of SIP [27].

This room-management architecture is centralized. The room server manages all the rooms and the users therein, and all the SIP messages are handled by the SIP proxy. However, as described in Section 3.2, the actual communication between the users is controlled locally, and the room server does not know the communication structure. In addition, the function of the room server can be replaced by a peer-to-peer (P2P) mechanism such as used by Lennox [28]. If security is not a main concern, it is easy to implement the room-management function by a P2P mechanism because the function is simply to obtain the closure of the participant information. It is also possible to replace the proxy-based SIP messaging by a P2P mechanism.

Standard Edition (i.e., J2SE) supplied by Sun Microsystems. DirectX (a trademark of Microsoft) and OpenGL (a trademark of Silicon Graphics) are graphics APIs. JMF has audio capture (by microphones) and transmission (by RTP) functions. LWJGL (Light-Weight Java Game Library) is an API for utilizing OpenGL and OpenAL and is developed at SourceForge.net. OpenAL is the core of the spatial sound libraries. Java 3D has spatial audio function but it does not work with RTP, so I had to connect the stream transmitted by RTP to the spatializer by using LWJGL.

[1] However, currently the stream data is identical to all the destinations.

[2] There is no particular reason why messages for creating a room and entering it are not separated.

### 5.3 Policy-based session and floor control

Each user agent knows the location and direction of all the users in the room. User agent A tests the communication policies and, if a policy condition is met, it takes the communication action that is specified as the policy action. For example, the policy is assumed to be as follows (the same policy as described in Section 3.2).

- If another person comes within 5 m, connect to that person, and
  if another person goes over 6 m away, disconnect from that person.

If user A currently does not communicate with user agent B and now B is 5 m away, A sends an INVITE request to B, who may accept or refuse the request. If the condition of A does not meet the policy condition of B, B refuses to connect, i.e., returns a response "488 not acceptable here". The stronger policy thus applies as described in Section 3.2. There is no centralized mechanism required for this arbitration. If B does not recognize A because B has not yet received the room-user list from the server, B responds in the same way as above.

If A is currently communicating with B and now B is 6 m away, A sends a BYE request to B. The streams between A and B are immediately disconnected. A user agent that has a stronger policy sends a BYE request before the other side sends one. Accordingly, the stronger policy is applied again.

Each of two user agents may send an INVITE request before it receives another INVITE request sent by the other agent; i.e., the INVITE requests may be crossed. In this case, a (double-dialog) glare [28] may occur. A glare means a doubly connected situation. If both agents respond "200 OK" to the INVITE requests, a glare occurs. However, it is easy to avoid the glare in SIP. Because each agent remembers that it has sent an INVITE request when it receives an INVITE request from the other sides, and each INVITE request contains a unique call identifier, they can choose the same request from them, and the agent that received the request returns a response "480 temporary unavailable" to the other. For example, they can compare the call identifiers and choose the larger one.[3]

## 6. Evaluation

The prototype can be used by three or more people to talk to each other and to hear virtual speakers (i.e., prerecorded voices). Although the sound localization depends on each person's auditory characteristics, most people recognized a virtual speaker walking on a circle trace, and roughly distinguished the direction and distance of speakers. However, extensive evaluation on multi-context communication by users has not yet been conducted because the voice quality was not yet sufficient for human evaluation.

The policy-based communication-control function was also tested. It took several seconds to receive the voice of a user since the user comes within the connection distance. In this case, the SIP messaging (i.e., INVITE, 200 OK, and ACK) took less than a second. It took more time (because polling was used but it was usually less than two seconds) to receive the user list, which must be received by both users before the SIP messaging succeeds. It usually took less than two seconds to disconnect from a remote user since the local

[3] This asymmetric solution is possible only if some centralized mechanism is available. If the messages are symmetric, a more elaborate solution [2] must be devised.

user goes over the disconnection distance.

However, in our Java-based implementation (i.e., by using JMF and Java 3D), we could not sufficiently reduce the communication delay and the fluctuation of voice processing caused by other tasks, especially 3D graphics processing.

## 7. Conclusion and Future Work

To solve complicated and restricted conference control and to enable multi-context conferencing, the author proposed virtual-location-based communication using spatial audio and personalized policy-based communication control enabled by a distributed policy-arbitration mechanism. These features were implemented in the prototype. The test results showed that the above basic localization mechanisms worked as expected in spite of low-cost hardware and software, so they should work well with multi-context communication. The policy-based communication control was also tested, and it worked well with a simple distance-based policy. If a good user interface is available, these features will be useful for realizing a general-purpose voice-communication medium.

As our next goal, an improved prototype with wearable terminals will be developed and evaluated with human testers. The new prototype will use a SIP event-notification mechanism for more reliable room and user management.

## References


[1] Kanada, Y., A Virtual 'Sound Room' Based Communication-Medium Called *Voiscape*, *Technical Report of IEICE,* (*SIG MVE*), Institute of Electronics, Information and Communication Engineers (IEICE), October 2003 (in Japanese).

[2] Kanada, Y., Policy-Based Session Control in a Virtual 'Sound Room' Based Communication-Medium Called *Voiscape*, *Technical Report of IEICE,* (*SIG IA/IRE*), Institute of Electronics, Information and Communication Engineers (IEICE), October 2003 (in Japanese).

[3] Koskelainen, P., Schulzrinne, H., and We, X., A SIP-based Conference Control Framework, *12th Int'l Workshop on Network and Operating System Support for Digital Audio and Video* (*NOSSDAV'02*), 53–61, June 2003.

[4] Dommel, H-P., Garcia-Luna-Aceves, J. J., Floor Control for Multimedia Conferencing and Collaboration, *Multimedia Systems*, *5*, 23–38, 1997.

[5] Rosenberg, J., Schulzrinne, H., Camarillo, G., Johnston, A., Peterson, J., Sparks, R., Handley, M., and Schooler, E., SIP: Session Initiation Protocol, RFC 3261, IETF, June 2002.

[6] Handley, M., Wakeman, I., and Crowcroft, J., CCCP: Conference Control Channel Protocol: A Scalable Base for Building Conference Control Applications, *ACM SIGCOMM'95*, 275–287, 1995.

[7] Koskelainen, P., Ott, J., Schulzrinne, H., and Wu, X., Requirements for Floor Control Protocol, Internet Draft, draft-ietf-xcon-floor-control-req-01, IETF, July 2004.

[8] Cherry, E. C., Some Experiments on the Recognition of Speech, with One and with Two Ears, *Journal of the Acoustical Society of America*, *25*, 975–979, 1953.

[9] Stifelman, L. J., The Cocktail Party Effect in Auditory Interfaces: A Study of Simultaneous Presentation, MIT Media Laboratory Technical Report, 1994.

[10] Berc, L., Gajewska, H., and Manasse, M., Pssst: Side Conversations in the Argo Telecollaboration System, *17th ACM Symposium on User Interface Software and Technology* (*UIST 95*), 155–156, November 1995.

[11] Rosenberg, J., A Framework for Conferencing with the Session Initiation Protocol, Internet Draft, draft-ietf-sipping-conferencing-framework-02, IETF, June 2004.

[12] Begault, D. R., 3-D Sound for Virtual Reality and Multimedia, NASA/TM-2000-XXXX, NASA Ames Research Center, April 2000, http://human-factors.arc.nasa.gov/ihh/-spatial/papers/pdfs_db/Begault_2000_3d_Sound_-Multimedia.pdf

[13] Benford, S. D., and Fahlén, L. E., A Spatial Model of Interaction in Large Virtual Environments, *3rd European Conference on CSCW* (*ECSCW'93*), Milano, Italy, Kluwer, 1993.

[14] Greenhalgh, C. and Benford, S., MASSIVE: a collaborative virtual environment for teleconferencing, *ACM Transactions on Computer-Human Interaction* (*TOCHI*), *2*(3), 239–261, September 1995.

[15] Nakanishi, H.: FreeWalk: A Social Interaction Platform for Group Behaviour in A Virtual Space, *Int. J. Human-Computer Studies*, *60*, 421–454, 2004.

[16] Rodenstein, R. and Donath, J. S., Talking in Circles: Designing A Spatially-Grounded AudioConferencing Environment, *ACM CHI 2000*, 81–88, April 2000.

[17] DiPaola, S. and Collins, D., A 3D Virtual Environment for Social Telepresence, *Western Computer Graphics Symposium*, 2002.

[18] Hollier, M. P., Rimell, A. N., and Burraston, D., Spatial Audio Technology for Telepresence, *BT Technical Journal*, *15*(4), 33–41, 1997.

[19] Low, C., and Babarit, L., Distributed 3D Audio Rendering, *7th International World Wide Web Conference* (*WWW7*), 1998, http://www7.scu.edu.au/programme/-fullpapers/1912/com1912.htm.

[20] Singer, A., Hindus, D., Stifelman, L., and White, S., Tangible Progress: Less Is More In Somewire Audio Spaces, *ACM CHI '99*, 104–112, May 1999.

[21] Aoki, P. M., Grinter, R. E., Hurst, A., Szymanski, M. H., Thornton, J. D., and Woodruff, A., *Sotto Voce:* Exploring the Interplay of Conversation and Mobile Audio Spaces, *ACM CHI 2002* (*Conference on Human Factors in Computing Systems*), 431–438, April 2002.

[22] Lokki, T., Savioja, L., Väänänen, R., Huopaniemi, J., and Takala, T., Creating Interactive Virtual Auditory Environments, *IEEE Computer Graphics and Applications*, July/August 2002, 49–57.

[23] Savioja, L., Modeling Techniques for Virtual Acoustics, Helsinki University, 1999.

[24] About the IP telephony project, http://snad.ncsl.nist.gov/-proj/iptel/, National Institute of Standards and Technology.

[25] O'Doherty, P., SIP Specifications and the Java Platforms, Sun Microsystems, 2003, http://java.sun.com/products/jain/-SIPAPIS.pdf

[26] Schulzrinne, H., Casner, S., Frederick, R., and Jacobson, V., RTP: A Transport Protocol for Real-Time Applications, RFC 1889, IETF, January 1996.

[27] Roach, A. B., Session Initiation Protocol (SIP)-Specific Event Notification, RFC 2543, IETF, June 2002.

[28] Lennox, J. and Schulzrinne, H., A Protocol for Reliable Decentralized Conferencing, *13th Int'l Workshop on Network and Operating System Support for Digital Audio and Video* (*NOSSDAV'03*), 72–81, June 2003.