RAVITAS: REALISTIC VOICE CHAT FRAMEWORK FOR COOPERATIVE VIRTUAL SPACES

Keiichi Yasumoto

Graduate School of Information Science
Nara Institute of Science and Technology
Ikoma, Nara 630-0192, JAPAN

Klara Nahrstedt

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, IL 61801, USA

ABSTRACT

In this paper, we propose RAVITAS, a framework for realistic voice chat among multiple users in a virtual space reproducing the cocktail party effect. RAVITAS utilizes context-aware voice filtering (CAVF), pub/sub-based locality management, and controlled voice streaming to achieve this effect. Our preliminary experiments show that RAVITAS achieves satisfactory perception-based subjective results for a small group of users.

1. INTRODUCTION

Increase in bandwidth availability, development of Voice over IP (VoIP) technology, and an overall increase in development of multimedia-aware protocols over the Internet such as RTP (Real-time Transport Protocol), RTCP (Real-Time Control Protocol), SIP (Session-Initiated Protocol) allow distant users to communicate with each other via real voices. IP-based phones are replacing existing analog fixed-line phones, and video conferencing systems and instant messengers are becoming widely used. On the other hand, there is a need for more realistic voice communication support in collaborative applications such as multi-player networked games and CSCW (Computer-Supported Cooperative Work), in order to facilitate collaboration/conversation among users in a virtual space. In such applications, 3D sound effects are important and often essential. For example, in a multi-player networked game, each user should be able to hear a voice from a distant user in danger and go to his/her place for help. In a virtual cafe, each user may want to hear voices from multiple conversation groups and join the group with the most interesting topic by moving close to the virtual group.

In order to achieve realistic chat among users via real voices in a virtual space, following requirements need to be considered: (1) Voices should be played back with 3D effects so that each user can recognize from which direction and location the voices come; (2) Users should be able to move around in a virtual space and get closer to someone for conversation; (3) In the real world, each individual can hear multiple voices from simultaneous conversations and extract only necessary voices according to the *cocktail party effect* [1]. It is known that visual information helps to assist in this effect. So, in a virtual space, each user should hear voices from speakers of desired conversations more clearly than other audible voices; (4) Even during the conversation, each user should recognize what others nearby are talking about, especially when the user tries to carefully listen to them; (5) If we want to achieve the above criteria (1)-(4) in a pervasive environment (e.g., PCs with headsets connecting to the Internet), it would be important to consider voice propagation delays and required bandwidth, which depend on resources available in user environments; and (6) The whole system should scale against the number of users existing in a virtual space.

There are several research efforts to achieve voice chat in a virtual space. 3D sound libraries such as OpenAL [5] make it possible to reproduce realistic 3D sound effects by playing back multiple environmental sounds such as explosion and footsteps. Sounds are mixed depending on the relative distance/angle between the listener and sound sources, and their volumes. However, since these libraries aim at playing back pre-recorded sound data, it would be difficult to produce 3D sound effects with real-time human voices.

In [3], virtual chat framework called *voiscape* is proposed. A prototype system is implemented in Java. In the system, user terminals exchange locations of avatars through a centralized server called the *room server*. Voice data is exchanged by direct P2P communication among user nodes. Furthermore, in *voiscape,* the above requirements (1) and (2) are achieved using LWJGL (Light Weight Java Game Library) which includes APIs for OpenAL and OpenGL. However, the requirements (3)-(6) are not considered in detail.

In [2], delays and required bandwidth for voice communication in a virtual space have been measured and compared among various communication architectures such as a centralized server, direct P2P communication and distributed proxies. Also, the authors found an interesting relationship between logical locations of avatars in a virtual space and physical locations of corresponding real Internet users. However, such performance measurement has been conducted through simulations, and the simulations are rather static, i.e., avatars do not move around in the virtual space.

We propose a framework for realistic voice communication among multiple users in a virtual space, called RAVITAS. To reproduce realistic situations in a physical shared space (e.g., chat during a banquet, chat in a bar), RAVITAS emulates the cocktail party effect in a virtual space. For this purpose, RAVITAS deploys context-based voice filtering called CAVF, pub/sub-based locality management and voice streaming. The very preliminary perceptual results indicate that the RAVITAS framework and its internal algorithms do make a difference and allow listeners to join automatically different conversational groups easier than other systems such as *voiscape*.

The rest of the paper is divided as follows: Section 2 presents the basic models of the framework and Section 3 goes into details of the RAVITAS framework. Section 4 addresses RAVITAS' experimental validation. Section 5 concludes the paper.

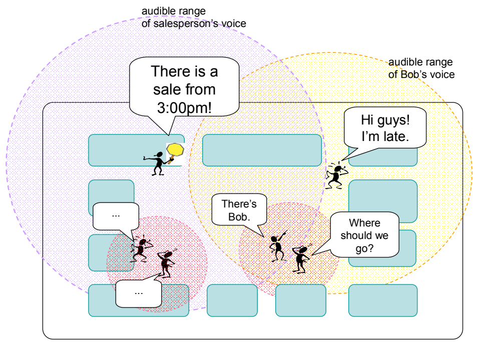

audible range of salesperson's voice
audible range of Bob's voice
There is a sale from 3:00pm!
Hi guys! I'm late.
There's Bob.
Where should we go?
...
...

Figure 1: Virtual Space and Sound Attenuation

2. RAVITAS MODELS

RAVITAS consists of the virtual space model, implemented via the user interface, and the physical space model, implemented via techniques and protocols over the overlay Internet network.

The *shared virtual space* model, shown in Figure 1, consists of multiple *avatars* (i.e., objects representing real users) that can move around and the corresponding real users can communicate with each other using their voices. We represent the audible ranges of voices as circles centered around the sound sources with different radii (depending on their volume). The users whose avatars are out of the *audible range* cannot hear the voices in the circle. For easy implementation, we do not consider reflection or refraction of sounds occurring due to obstacles located in the virtual space. A technique to deal with obstacles can be found in [4]. To accommodate arbitrary number of avatars, we divide the virtual space into subspaces in the form of *hexagons* as shown in Figure 2. The reason for the hexagon shape is that is easy to model the voice propagation (no holes).

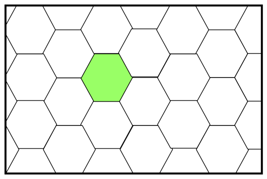


Figure 2: Division of Virtual Space

The *shared physical space* model consists of overlay nodes that include a set of user nodes and locality servers. The locality servers reside on some of the user nodes. Each locality server captures location of avatars moving in one hexagon of the virtual space.

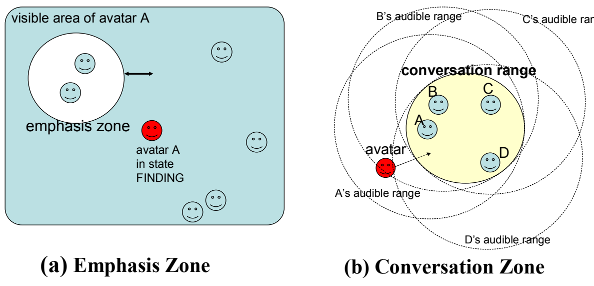
3. RAVITAS FRAMEWORK

For RAVITAS to realize the cocktail party effect in virtual space, and solve the challenges given by the requirements (1)-(6) in Section 1, we adopt the following components:
(1) *context-aware voice filtering (CAVF)* allows each user to hear voices more clearly of other users/avatars that belong to the same conversation group or are in the listener's interest range, and mimics the cocktail party effect in the virtual space.
(2) *distributed publish-subscribe (pub/sub)-based locality management* allows users to efficiently exchange their avatars' logical location information.
(3) *voice management* allows the system to do voice source association and voice streaming to users.

Context-Aware Voice Filtering (CAVF): CAVF assists users to feel the cocktail party effect in the virtual space, and allows each user to hear voices from desired users more clearly. The CAVF mechanism is based on a *two-state per-user finite state machine*. In the state FINDING, the user is looking for other users to talk to. In the state TALKING, the user is engaged in the conversation with other users.

To transit from the state FINDING to TALKING, the user must differentiate interesting voices from other multiple simultaneous voices. To facilitate this kind of transition, we allow a user in his/her FINDING state to specify the *emphasis zone (user-specified audible range)*, as it is shown in Figure 3(a)[1]. The radius and position of the circle can be changed by the user, by dragging the circle with a mouse. We keep the volume of voices from the emphasis zone loud and make the volume of other voices less loud. The volume strength depends on each user's preference.

To transit from TALKING to FINDING state, we introduce the concept of *membership awareness* and a mechanism to automatically control each user's membership in a conversation group. We call a set of users, who are currently talking to each other, the *conversation group*. A new conversation group is created when a user starts to talk to other independent users. For each conversation group, the *conversation zone* is calculated as a circle which is contained in the intersection of standard audible ranges of all members, as shown in Figure 3(b)[2].



(a) Emphasis Zone **(b) Conversation Zone**

Figure 3: Context-aware Voice Filtering

If a new avatar finds and enters into the conversation zone of a group, he/she automatically joins the group, his/her state automatically changes to TALKING, and he/she can hear the

[1] The emphasis zone must be within the audible range and it can span across multiple hexagon virtual subspaces.
[2] The emphasis zone may cover several conversation zones or the emphasis zone may be included in one of the conversation zones.

voices from the members more clearly in his emphasis zone. If the avatar moves out of the conversation zone, the corresponding user's membership is canceled.

Since multiple users/avatars can be in TALKING state at the same time and can belong to the same and/or different conversation groups, this can cause environmental noises, i.e., users may have hard time to hear each other. Therefore, we introduce a *policy* that per conversation group only one user /avatar can be in the TALKING state. This policy is enforced using a simple *token-based session management* scheme. Here we discuss only the token-based scheme under the following assumptions: (1) one token is used per conversation group and (2) each conversation group is confined to one hexagon subspace. Under these assumptions, as shown in 4(a), users (avatars in one hexagon) within the conversation group who want to talk send a request to the corresponding locality server of the hexagon.

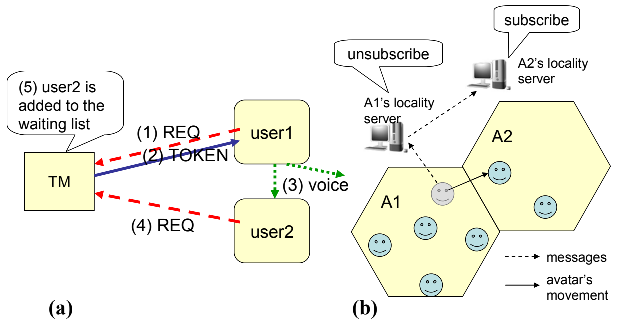


Figure 4: Control Management

Initially, the locality server has a token, and the token is sent to the user (and his/her user node) who first sent the request. The locality server retains the per-conversation group waiting list (i.e., a FIFO queue) of users who requested to talk. The token is returned to the locality server, when the user finishes talking, silence is detected, or certain time expires. Then the locality server sends the token to the next user node in the list.

Distributed Pub/Sub-Based Locality Management: The pub/sub approach assists RAVITAS in keeping (a) logical locations of avatars in virtual space and (b) mappings from virtual locations to physical locations consistent. The pub/sub-based locality management system runs on location servers. The *selection policy* of locality servers could be (a) random, (b) latency-driven, i.e., based on shortest delays from user nodes in the hexagon, or (c) resource-availability driven, i.e., based on available resources of the user nodes. One location server is assigned per one virtual hexagon and it maintains (1) logical locations of avatars in the virtual space, (2) mappings between the virtual locations of avatars and physical location addresses of user nodes, (3) physical location addresses of neighbor locality servers for mobility purposes when an avatar crosses the border of hexagons, and (4) avatar states (TALKING or FINDING).

When a user runs RAVITAS, the corresponding user node sends a subscribe message with its IP address and his/her avatar's initial location in a virtual space to one (e.g., closest neighbor) of the locality servers. Then, the message is automatically transferred to the locality server responsible for the virtual hexagon where the user/avatar wants to reside (participate in the conversation). When the user/avatar moves, there are two *mobility scenarios* that pub/sub-based locality management system will accommodate.

First, when an avatar changes its virtual location within one hexagon, the underlying user node of the avatar sends a publish message to the hexagon locality server. In response, the locality server forwards the message to all subscribers (user nodes) in the hexagon. Consequently, all user nodes know virtual/logical locations of other avatars within the same hexagon.

Second, when an avatar moves across the border of hexagons, the corresponding user node must unsubscribe from the locality server of the current hexagon and subscribe to the locality server of the new hexagon as shown in 4(b). To execute this subscription method, the following protocol happens: since each user node only knows the current locality server, it sends the avatar's new virtual location to the current locality server and asks the current locality server to assist in discovery of the new hexagon locality server. The current locality server checks if the avatar is truly out of its hexagon, and if so, the locality server unsubscribes the avatar and forwards the new avatar’s virtual location to the corresponding locality server. After that, the user node knows the new locality server by receiving new information from the new locality server.

Voice Management: The voice management consists of *voice source association* and *voice streaming*. The voice source association component keeps track of the audible range (emphasis zone). When a user talks in one hexagon, the *voice source association* works with the locality servers to (a) identify avatars and their logical locations within the circle, and (b) translate the logical locations into physical user node addresses. Then, the *streaming module* streams the voice data to user nodes of those avatars with direct unicast communication. IP multicast or application level multicast can be used if available. Note that in the background the following protocol may occur: if a user emits a loud voice or his/her avatar is near the border of the hexagon, the audible range may have intersections with multiple hexagons. In this case, the user node asks the hexagon locality server to discover necessary information about avatars which reside in the audible range but outside of the user’s hexagon. The server, in response, asks the neighbor servers to find the avatars within the circle.

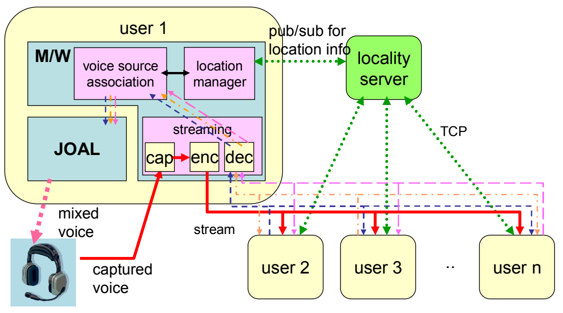


Figure 5: RAVITAS System Implementation

4. EXPERIMENTAL VALIDATION

Implementation: As shown in Figure 5, implementation of RAVITAS includes three modules: a location manager, a voice source association module and a streaming module. 3D sound effects are achieved using JOAL [6] (Java Bindings for OpenAL). The voice source association module maps multiple voice streams onto sound sources for JOAL, and handles the context-aware voice filtering. The streaming module exchanges voice data with other user nodes via network.

We have implemented a prototype application using our framework, J2SDK1.4.2_05 and JOAL1.1.0_04 for Windows. A couple of snapshots of the user interface (virtual space) are shown in Figure 6. Here, a virtual space is drawn as a 2-D space in a window, and avatars are drawn as bitmaps. We used balloons to show who is currently talking. The circles represent either the emphasis zone or the conversation zone which the user is joining.

Experimental Setup: In our preliminary study, we investigate the *impact of the proposed CAVF technique* using our prototype system. The pub/sub locality management system is very simple at this point. Figure 6 shows our virtual space setup, i.e., we put one listener (current user) and three conversation groups A, B and C into the virtual space, where two people talk alternately in each group. We use pre-recorded conversations in English. Our system plays back these conversations simultaneously. For experiments, we use an ordinary PC (Pentium4 1.7GHz and 1GB memory, and Windows 2000) with a headset.

Experimental Methodology: The experimental evaluation is based on the *subjective feedback* of people. We asked 6 persons how clearly they can understand conversations. We used scoring approach with 1 to 5 scores (1: cannot understand at all, 3: understandable if carefully listening, 5: easy to understand) for both cases with and without CAVF and for two different virtual locations L1 (Figure 6Figure (a)) and L2 (Figure 6Figure (b)) of the listener. We show the average scores in Table 1.

Table 1: Audibility

	Group A	Group B	Group C
OpenAL (L1)	4.1	2.1	3.5
OpenAL+CAVF (L1)	5.0 (3.0)	3.8 (1.5)	5.0 (2.1)
OpenAL (L2)	3.1	3.6	4.1
OpenAL+CAVF (L2)	2.1	5.0	2.6

In location L1, even without CAVF, most examiners could understand conversations from group A (average score was 4.1) since only this group is located on the left side of the listener, and the quality of the voice is good. On the other hand, conversation from group B (average score was 2.1) was hard to understand since this group is located in the same direction as group C, and its voice quality is not very good. Conversation of group C (average score was 3.5) was understandable. With CAVF, the average scores were improved to a great extent, while scores decreased when out of emphasis zone (scores in parentheses).

In location L2, without CAVF, the voice from group B was still hard to understand (average score was 3.6), since examiners heard mixed voices from groups B and C. With CAVF, the listener could hear the voice from group B much more clearly (average score was improved to 5.0) since our CAVF allows the listener to automatically join the conversation group B and emphasizes the voice.

5. CONCLUSIONS

In this paper, we proposed RAVITAS, a framework for realistic voice chat among multiple users/avatars in the virtual space. Our main contribution is in the context-aware voice filtering (CAVF) technique to help users easily feel the cocktail party effect by controlling membership of each user, and the distributed scheme for location management of users/avatars which is scalable against the number of users in a large virtual space. Through experiments, we confirmed that our CAVF technique greatly increases human perception in a virtual world.

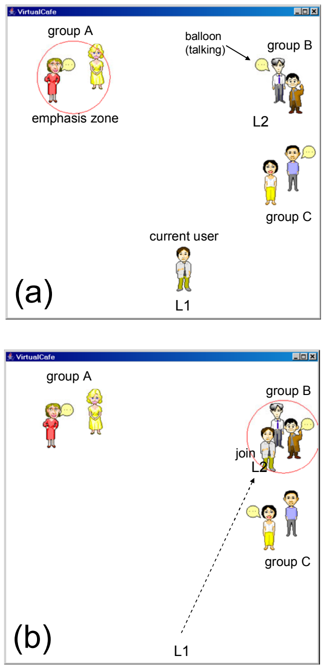


Figure 6: Prototype system

7. REFERENCES

[1] B. Arons: A Review of The Cocktail Party Effect, *Journal of the American Voice I/O Society* 12, pp. 35-50, 1992.
[2] P. Boustead and F. Safaei: Comparison of Delivery Architectures for Immersive Audio in Crowded Networked Games, *Proc. of 14th ACM Int'l. Workshop on Network and Operating System Support for Digital Audio and Video (NOSSDAV2004)*, pp. 22-27, 2004.
[3] Y. Kanada: Multi-Context Voice Communication Controlled By Using An Auditory Virtual Space, *Proc. of 2nd IASTED Int'l. Conf. on Communication and Computer Networks (CCN 2004)*, 2004.
[4] S. A. Maffra, M. Gattass and L. H. Figueiredo: Propagation of Sound in Two-Dimensional Virtual Acoustic Environments, *Anais do II WTDCGPI do Sibgrapi, Sao Carlos, SP*, Brasil, 2003.
[5] OpenAL home page, http://www.openal.com/
[6] Joal home page, http://joal.dev.java.net/