Admux Communication Protrocol For Real-time Multimodal Intreaction

Mohamad Eid and Abdulmotaleb El Saddik
Division of Engineering, New York University Abu Dhabi, United Arab Emirates
School of Electrical Engineering and Computer Science
University of Ottawa, Ottawa, Canada
mohamad.eid@nyu.edu; elsaddik@uottawa.ca

Abstract **—In our previous work [1], we proposed an adaptive application layer communication framework, named Admux, for multimedia applications incorporating haptic, video, auditory, and graphics information for non-dedicated networks such as the Internet. In this paper, the contribution is two-fold: first, we present a thorough description of Admux communication protocol and content access/communication management. Second, the evaluation of Admux, using an interactive multimodal 3D Office Slingshot game, is described. The 3D Office Slingshot game involves the communication of synchronous haptic-audio-video media – with both tactile and kinesthetic haptic feedback. The performance evaluation shows that Admux is capable of delivering synchronous haptic-video data while adapting to network conditions by allocating proportional resources to various media channels. The usability testing with 20 subjects has shown that players have expressed positive feedback about the game.**

***Keywords:** multimodal interaction, haptic gaming, multimodal communication framework, communication protocol.*

I. INTRODUCTION

Multimedia utilizes multiple sensory channels to exchange information (that is both spatially and temporally correlated) with a user. Traditionally, graphic images, audio, video, text and animations define the contents used in a multimedia system. Recently, researchers have made significant progress in multimedia systems by incorporating virtual reality augmentations (3D virtual objects) as well as advanced media such as haptics and scent into the human computer interaction paradigm. The trend with multimedia applications and systems is the incorporation of multiple media in order to enhance the user's quality of experience [2].

In particular, haptic modality (both tactile and kinesthetic) is considered of crucial importance for a wide spectrum of applications such as medical simulations and training, education and learning, and inter-personal communication and gaming [3]. For example, plenty of game prototypes take advantage of the haptic effects offered by mainstream haptic interface such as Haptic the Tactile Gaming Vest (TGV) game [4], HapticCast [5], and the Haptic Battle Pong (HBP) [6], among others [7-8].

The relationships between the different media, how they correlate and compliment, and their respective contributions to the quality of user experience have been studied by many researchers [9]. Nonetheless, the communication of such multimodal contents over non-dedicated networks remains a challenge. First, each media is characterized by different and sometimes conflicting communication requirements (QoS requirements). For example, kinaesthetic haptic communication requires a 1 kHz update rate with very low packet size whereas video data is only 30 Hz with huge volume of data per frame (compared to haptic frame size). Furthermore, the communication of haptic data is constrained by very strict delay constraints in order to assure local control loop stability [10].

Several multimodal data communication frameworks have been developed to tackle some of these challenges [11-12]. Admux is an application layer framework for synchronous haptic-audio-video communication over the Internet network [1]. Admux has the unique feature of dynamic multiplexing by allocating proportional network resources to various media streams in according to the application needs and the available network resources. Our previous work introduced Admux with its distinguished features and applications [1]. In this paper, we introduce Admux communication protocol and test its performance against the 3D Office Slingshot game which incorporates synchronous haptic-video communication. The game also incorporates both kinesthetic and vibro-tactile feedback.

The rest of the paper is arranged as follows: Section II presents the related work in real-time multimodal communication. In section III, a summary of Admux communication protocol is introduced, including content access management as well as protocol packetization. Section IV introduces the 3D Office Slingshot game and presents the performance evaluation and a brief discussion about the relative findings. Finally, section V summarizes our findings and provides directions for future work.

II. RELATED WORK

The idea of building a communication protocol that facilitates dissemination of multimodal contents found significant interest from both the research and industry communities. Two directions can be identified: (1) transport and network protocols such as such as SCTP, Light TCP, and RTP/I, and (2) application layer protocols to handle specific application requirements.

Few transport protocols have been proposed and evaluated for haptic applications (such as SCTP, Smoothed SCTP, Light TCP, RTP/I, and STRON) [13]. These protocols do not support the strict communication requirements of haptic media. Other protocols were proposed to handle the communication of haptic media (example can be found here [14-15]. For instance, the protocol proposed in [15], named ALPHAN, uses multiple-buffer scheme to prioritize and optimize multimodal data transfer. However,

1550-6525/12 $26.00 © 2012 IEEE
DOI 10.1109/DS-RT.2012.23

ALPHAN is not adaptive to dynamic changes in the network resources.

Application layer protocols were also proposed and used for multimodal communication – examples are MPEG-4 and CHAI 3D. MPEG-4 is an object-based multimedia framework that supports streaming data for various media as well as interactivity in broadcast multimedia applications [16]. Some researchers have tried to augment MPEG-4 specifications to transmit haptic data [17-19]. For instance, the authors in [18] propose an authoring and editing framework that extends MPEG-4 Binary Format for Scene (BIFS) to conveniently represent haptic data in the scene graph and facilitate haptic data broadcasting. New nodes for tactile and kinesthetic data are added to the BIFS description language in order to attach various haptic properties to the scene. However, all these efforts do not support haptic-video synchronization.

Several frameworks were introduced to handle haptic data communication, such as CHAI 3D [20], Reachin API [21], and H3D API [22]. CHAI 3D facilitates the development of multimedia virtual worlds including haptics, visualization and interactive real-time simulation. However, CHAI 3D does not support any communication and/or synchronization mechanisms for audio/video transmission. Reachin API allows the development of haptic-visual applications with high-fidelity features as well as a complete set of classes, nodes and interfaces for managing and synchronizing the haptics and graphics, and audio aspects of advanced 2D and 3D applications in a hierarchical data structure. However, here is no mechanism defined for live streaming and real-time interactions. The H3D API, based on X3D, is an open-source haptics software development platform that is based on OpenGL for graphics rendering and OpenHaptics for haptic rendering. Similar to Reachin API, H3D does not support any real-time communication mechanism for synchronized haptic-audio-video data.

Other researchers have used statistical multiplexing to improve the efficiency of communication over a limited bandwidth network [23]. One of the few works in this area is to dynamically control the arrival rate of multimedia data by switching the coders to different compression ratio (changing the coding rate) based on the network conditions [24]. The work in [25] investigated the use of self-organizing neural networks to design a statistical multiplexer for video streams. However, haptic media is not considered in both.

However, to the best of our knowledge, there is no framework capable of communicating synchronous haptic (tactile and/or kinesthetic), audio, video, and graphics data. Some frameworks such as MPEG-4 support the communication of audio, video, and graphics data but not haptic media whereas others (such as CHAI 3d, Reachin API, and H3D) focus on haptic data communication only. Admux communication protocol dynamically adapts network resources according to application requirements and network conditions.

III. Admux Communication Protocol

An overview of Admux communication framework is shown in Figure 1. The application generates multiple streams of media data that are compressed using different codecs (depending on the media type). The compressed streams are multiplexed using the Mux block based on the current Quality of Service (QoS) parameters (defined in the HAML-QoS datastore). Based on the available network resources, the multiplexer dynamically re-configures the codecs to comply with the available resources. The multiplexed stream is packetized and transmitted using underlying network interface (implementing UDP transport protocol). Two major features of Admux are described in this section: content access management and protocol packetization.

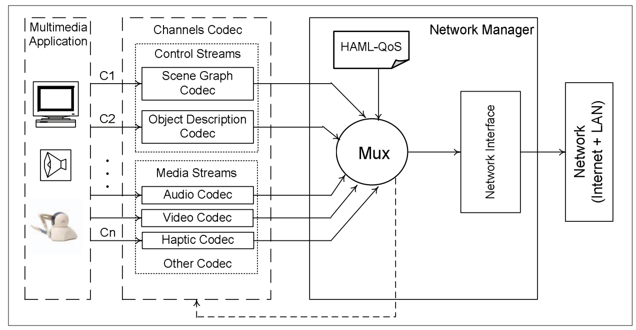

Figure 1. Overview of Admux communication framework.

A. Content Access Management

A major task of Admux specifications is to define the multimodal scene as well as the relationships between media components (objects) of the scene. The description is performed at two levels: structure level and content level. On the structural level, the scene description defines how the media objects are arranged in space and time. On the content level, the object description describes how the various channels that contain the media data relate to each other and how they are configured and synchronized.

The content access procedure always starts with a session description (Figure 2). The session description points to at least two essential channels: the scene description and the object description channels. Notice that the contents and control descriptions are separated to ease content management. The communicating entities, then, establish the two channels (scene graph channel and object description channel). Then they exchange the scene graph description through the scene description channel along with update messages to add their own haptic device object to the scene graph at each end.

Admux protocol defines several types of messages such as scene graph messages (describing relationships between the populating objects), object description messages, session messages, and media channel messages. For instance, the `InitialSession` message is used to initiate the communication session by defining the scene graph description and object description channels. After agreeing on these two channels, the communicating parties initiate the two channels and communicate the scene graph description and the object description information. One party sends a `SceneUpdate` message with its graph nodes to the other party (including the haptic interface object represented by a haptic node) whereas the other party integrates the received nodes into its own scene graph and sends back a `SceneUpdate` message with the total scene. The first party updates its local scene graph accordingly. The media channels (elementary channels) will be instantiated based on the object descriptions and media communication starts.

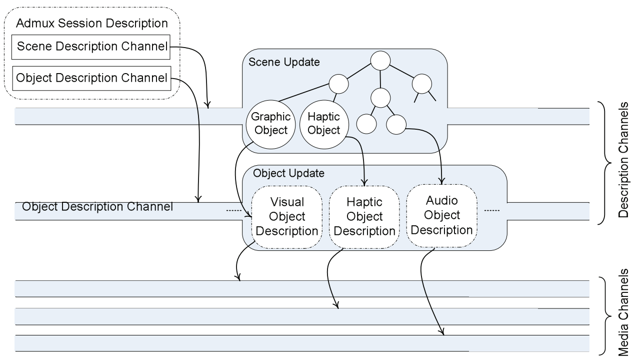

Figure 2. Content access management

B. *Admux Packetization*

Admux packetization of media streams is inspired from MPEG-4 specifications [16] and is performed in two steps: first the Packetized Elementary Stream (PES) packets are composed from the elementary stream data and then PES packets are encapsulated in Transport Stream (TS) packets. The PES packet stores one update/frame of a media stream and thus its size is media dependent (varying). The TS packetization is designed to allow multiplexing of equal size data units and to synchronize the output. Once the TS packet header is added, the TS packets are stored in the channel transmit buffer(s) for multiplexing. Typically, a PES packet may be much larger than a TS packet. Figure 3 shows the packetization process as a two-phase process.

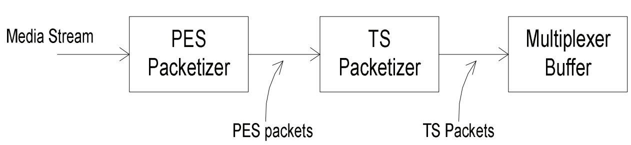

Figure 3. The packetization process.

The PES header carries various rate, timing, and descriptive information as set by the encoder. The PES packet length is described in a field provided for that purpose. The PES packetization interval is application dependent resulting in packets of variable length with a maximum definable size of 216 bytes. The PES packet header is shown in Figure 4. The followings describe each field briefly.

- *Channel_ID:* Each elementary channel is identified by a unique Channel_ID which is carried by the PES packet. This ID is also used by the de-multiplexer to identify the destination channel.
- *PayloadSize:* This field serves to specify the exact size of the PES payload. This field is necessary since each media type has different sizes for their messages.
- *Timestamp:* The timestamp indicates the instant of time where the update must be consumed at the receiver side and is used for synchronization. The field might also be used to establish global ordering among all updates generated by all the participants.
- *Frame Type:* Three bits are reserved to specify the type of the update. Only four types are predefined (see Table 1). Up to four other types of updates can be defined by application developers.
- *ObjectID:* The Object Identifier (ObjectID) uniquely identifies an object in the environment. An ObjectID is persistent for the lifetime of the simulation. This field is optional.
- *ParticipantID:* The Participant ID uniquely identifies a participant who is the original sender of the packet. A ParticipantID is persistent for the lifetime of the simulation. If an application requires multiple Admux sessions, the ParticipantID remains the same for the participant across multiple sessions. This field is relevant to collaborative applications and thus is optional.

Channel ID (1 byte)	Payload Size (2 bytes)	Timestamp (4 bytes)
Frame Type (3 bits)	ObjectID (Optional) (1 bytes)	ParticipantID (Optional) (1 bytes)
Payload		

Figure 4. The PES packet header.

Each PES packet is fragmented into fixed-sized transport packets (TS packets) to form a general purpose way of combining several channels, possibly with independent time bases. This is advantageous when there is a need to send multiple media channels at a time or/and when there may be potential packet loss or corruption by noise. The header structure of a TS packet is shown in Figure 5. Here is a brief explanation of each field.

TABLE I. PREDEFINED UPDATE TYPES.

Update Type	Description
Key Update	Holds data representing key events that must be conveyed reliably. Note that collision updates are sent reliably regardless of the update type.
Normal Update	Makes up the majority of the updates that are sent over the network. They are sent unreliably.
Incremental Update	Useful when bandwidth is limited, these updates hold incremental information with respect to last received key update.
Control Update	The control update is used for the delay and jitter estimation between two hosts on the network.

<table>
<tr><td>PID
(3 bits)</td><td colspan="2">Payload Unit Start Indicator
(1 bit)</td><td>Adaptation Field Control
(1 bit)</td></tr>
<tr><td colspan="2">Continuity Counter
(3 bits)</td><td colspan="2">payloadByteOffset
(1 byte)</td></tr>
<tr><td colspan="4">Payload</td></tr>
</table>

Figure 5. The TS packet header structure.

- *Packet Identifier (PID):* The PID is used to uniquely identify the channel to which the packet belongs. The PID allows the receiver to differentiate the channel to which each received packet belongs. Some PID values are predefined and are used to indicate various channels of control information. A packet with an unknown PID, or one with a PID which is not required by the receiver, is silently discarded.
- *PayloadUnitStartIndicator:* This field is a flag that indicates the start of a PES packet payload or the continuation of the previous payload for a PES packet. It is used during the reassembly of PES packets to indicate the arrival of the last TS packet of a PES packet.
- *AdaptationFieldControl:* This field is a 1 byte length that represents the number of bytes in the adaptation field immediately following this byte. The adaptation field contains additional optional transport fields that can be used for implementation application layer reliability mechanisms. This is crucial since Admux is based on UDP which does not provide any error detection/correction mechanisms.
- *ContinuityCounter:* This field is a 4-bit counter, which usually increments with each subsequent packet of a frame, and can be used to detect missing packets.
- *PayloadByteOffset:* The byte offset value of the start of the payload or the length of adaptation field is mentioned here.

IV. Evaluation with Office SlingShot 3D Game

We have implemented the Office SlingShot 3D game and used the proposed Admux as the communication protocol. The Office SlingShot 3D game is an interactive one-on-one slingshot game that incorporates synchronous haptic-audio-video interaction. The application uses the Novint Falcon haptic device to launch projectiles against the competitor, a 2.5D z-camera [26] to capture body movement, and a haptic jacket [27] that is capable of simulating hits. The players can effectively launch projectiles at each other while attempting to dodge incoming projectiles (as shown in the application snapshot of Figure 6).

A. Office Slingshot 3D Summary

Figure 6 shows all the GUI components needed for the game. On the top of the page, the local and peer health bars are located; as a player gets hit, her health decreases depending on where the hit has been detected (-1 points for a hit to the shoulders, -2 points for a chest hit and -3 points for a head shot). Also, the player feels the hit using a vibro-tactile haptic jacket (shown in Figure 6). The ball shown is controlled by the users' falcon device and gets launched when she lets go of the button. The velocity of the ball is computed by utilizing physics principles and according to its starting position and force. Finally, when a player's health runs out, the game is over and either the winning or losing screen will appear.

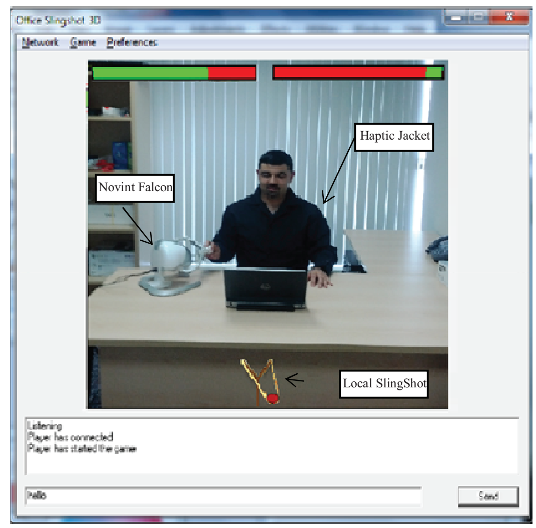

Figure 6. A snapshot of the Office Slingshot 3D game.

B. Experimental Test Bed

The experimental test bed is composed of two Pentium 4 PCs with 2 Gb RAM and 100Mbs Ethernet cards, two Novint Falcon haptic devices, two depth camera, and a haptic jacket (one prototype was available). The experiment was conducted over the Internet network between two hosts in the same city (Ottawa, Canada). The average delay between the two hosts was computed to be 40ms and the jitter was 6ms during the time of performing the experiment. A snapshot of the experimental game setup is shown in Figure 7.

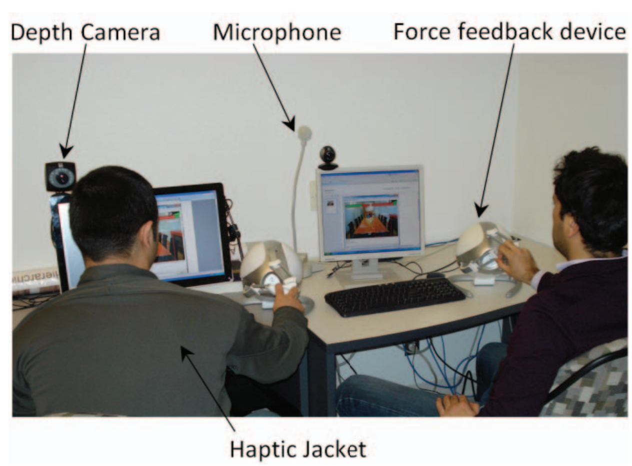

Figure 7. Setup of the experimental evaluation.

C. Results

Three experiments are presented in this section: time complexity analysis to measure the computational delay due to Admux processing, an analysis of the relationship between TS packetization and error rate, and finally an analysis for TS packet size optimization.

C.1 Time Complexity Analysis

One critical factor for real-time communication is the time complexity of the protocol. We used a high precision timer to measure the computation time for a complete multiplexing cycle (including packetization). Figure 8 demonstrates that the computation time is converging to 2.005 µs which is comfortably below the 1 ms delay needed for the haptic modality. The time overhead caused by the multiplexing scheme is comfortably negligible and eventually has no tangible impact on the communication quality.

C.2 TS Packet Size and Error Rate

One observation that we noticed during the implementation was the interdependence between the TS packet size and the error rate for each media channel. Therefore, we conducted an experiment to examine that relationship by measuring the per channel error rate as function of the TS packet size. The results are shown in Figure 9. The error rates for the Video channel and the depth channel data are decreasing as the size of the TS packet increases. This is because the packetization is resulting in less number of fragments per frame, and thus a complete frame is interleaved with less number of UDP packets.

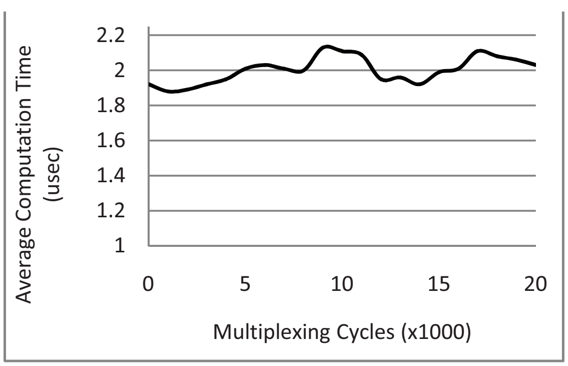

Figure 8. Time complexity analysis with Office Slingshot 3D game.

On the other hand, the haptic channel showed a different relationship; the average error rate was not affected by the change in the TS packet size. The reason is because the haptic frame is very small (only 32 bytes) and there was no fragmentation for this media. It is worth mentioning that even though smaller size TS packet has resulted in smaller values for the average error rate, but the video channel and depth channel have suffered larger delays.

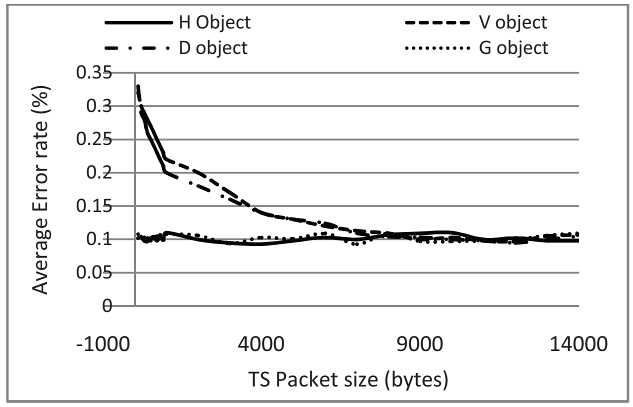

Figure 9. Average error rate (%) versus the TS packet size (H object, V object, D object, and G object refer to haptic channel, video channel, depth channel, and graphics channel).

C.3 TS Packet Size Optimization

We also examine the effects of the TS packet size on the average delays and jitter for the five media channels. The objective here is to find the optimal size of the TS packet in order to tradeoff the network delays and jitters and the average error rate. Figure 10 shows that the network delay and jitter are decreasing as the TS packet size increase since there is less fragmentation/defragmentation delays with larger packet size. This behavior is true until the packet size becomes around 5 Kbytes, after which the delay starts increasing again (shown in Figure 10) as the buffering overhead overweighs the fragmentation delays.

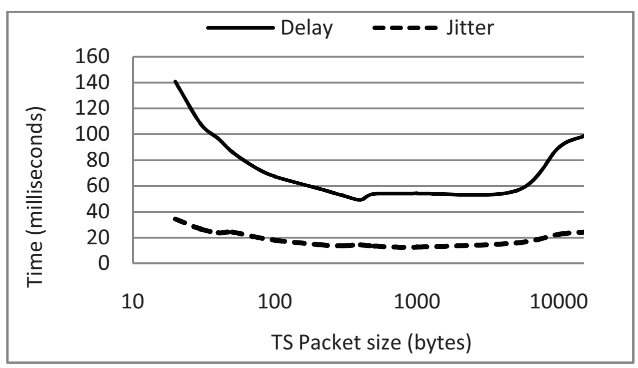

Figure 10. Delay/jitter variations with TS packet size.

V. Conclusion

Our performance evaluation with the Office Slingshot 3D game has demonstrated two distinguished capabilities of Admux. First, Admux provides a mechanism for the synchronization of haptic data with audio/video data with reasonable computation delays. Second, Admux adapts to application level events and interactions as well as to the network conditions. One interesting finding was the error detection and correction for PES packets. Since Admux uses UDP and thus encounters unreliable communication, there is a need for error resilience algorithm that can reassemble PES packets even in the case of loosing TS packets.

As per future work, we plan to investigate the potential for multi-user communication with Admux. In this case, the communication is dependent on synchronous and collaborative interactions from many players. Scalability becomes crucial for particular application domains such as social networking and gaming. Moreover, performing usability testing to evaluate player's satisfaction about playing the game will be part of our future work.

REFERENCES

[1] M. Eid, J. Cha, and A. El-Saddik, "Admux: An Adaptive Multiplexer for Haptic-Audio-Visual Data Communication", IEEE Transactions on Instrumentation and Measurement, Vol. 60, No. 1, pp.21-31, January 2011.

[2] R.L. Koslover, B.T. Gleeson, J.T. de Bever, and W.R. Provancher, "Mobile Navigation Using Haptic, Audio, and Visual Direction Cues with a Handheld Test Platform", IEEE Transactions on Haptics, vol.5, no.1, pp.33-38, Jan.-March 2012.

[3] M. Eid, M. Orozco, and A. El Saddik, "A Guided Tour in Haptic Audio Visual Environment and Applications ", International Journal of Advanced Media and Communication, vol.10, no.1, pp.10-17, Feb. 2007.

[4] iRoboticist: http://iroboticist.com/2010/03/26/tgv/, Tactile Gaming Vest (TGV), demoed at Haptics Symposium 2010.

[5] S. Andrews, J. Mora, J. Lang, and W.S. Lee, 'HaptiCast: a Physically-based 3D Game with Haptic Feedback', Proc. of Future Play 2006, London, ON, Canada, October 2006.

[6] D. Morris, "Haptic Battle Pong: High-Degree-of-Freedom Haptics in a Multiplayer Gaming Environment", Proc. Experimental Gameplay Workshop, GDC, 2004.

[7] W. Park, L. Kim, H. Cho, and S. Park, "Design of haptic interface for brickout game," IEEE International Workshop on Haptic Audio visual Environments and Games, pp.64-68, 2009.

[8] L.T. De Paolis, M. Pulimeno, and G. Aloisio, "The Simulation of a Billiard Game Using a Haptic Interface", 11th IEEE International Symposium on Distributed Simulation and Real-Time Applications, pp.64-67, 2007.

[9] R. MacLaverty and I. Defee, "Multimodal interaction in multimedia applications", In proceedings of the IEEE First Workshop on Multimedia Signal Processing, page(s): 25-30, Princeton, USA, 1994.

[10] A. El Saddik, "C-HAVE: Collaborative Haptic Audio Visual Environments and Systems", Multimedia Systems Journal, Volume 13, Number 4, 251, 2008.

[11] S. Lee and J. Kim, "Priority-based haptic event filtering for transmission and error control in networked virtual environments", Multimedia Systems Journal, Vol. 15, No. 6, 355-367, 2009.

[12] A. Boukerche, H. Maamar, and A. Hossain, "An efficient hybrid multicast transport protocol for collaborative virtual environment with networked haptic", Multimedia Systems Journal, Vol. 13, No. 4, 283-296, 2007.

[13] S. Dodeller, "Transport Layer Protocols for Haptic Virtual Environments", M.S. thesis, University of Ottawa, Canada, 2004.

[14] M. Kuschel, P. Kremer, S. Hirche, and M. Buss, "Lossy data reduction methods in haptic telepresence systems", In proceedings of the IEEE International Conference on Robotic Automation, pages: 2933–2938, Orlando, FL, 2006.

[15] H. Al Osman, M. Eid, and A. El Saddik, "Evaluating ALPHAN: A Communication Protocol for Haptic Interaction", In proceedings of the Symposium on Haptic interfaces for virtual environment and teleoperator systems, Page(s):361 – 366, 2008.

[16] F. Pereira and T. Ebrahimi, "The MPEG-4 Book", IMSC Press multimedia series, Prentice Hall PTR, New Jersey, NJ, 2002.

[17] D. Walsh, C. Gunn, M. Adcock, and M. Hotchins, "Haptics Nodes for MPEG-4 BIFS (Object Surface)", ISO/AEC JTC 1/SC 29/IWG 11, MPEG2001im7409, 2001.

[18] J. Cha, Y. Seo, Y. Kim, and J. Ryu, "An Authoring/Editing Framework for Haptic Broadcasting: Passive Haptic Interactions using MPEG-4 BIFS", In proceedings of WorldHaptics, pages: 274-279, Tsukuba, Japan, 2007.

[19] J. Zhou, X. Shen, and N.D. Georganas, "Haptic tele-surgery simulation", In proceedings of the IEEE Workshop on Haptic Audio Visual Environments and their Applications, Ottawa, Canada, page(s): 99- 104, 2004.

[20] M. Fayad and D.C. Schmidt, "Object-Oriented Application Frameworks", Communications of the ACM, Special Issue on Object-Oriented Application Frameworks, Vol. 40, No. 10, October 1997.

[21] Reachin Display official website, www.reachin.se, accessed on March 14, 2010.

[22] H3D official website, www.h3dapi.org, accessed on March 14, 2010.

[23] K. Chandra, "Statistical Multiplexing", Wiley Encyclopedia of Telecommunications, 2003.

[24] M.A. Saleh, I.W. Habib, and T.N. Saadawi, "Simulation Analysis of a Communication Link with Statistically Multiplexed Bursty Voice Sources", IEEE Journal on Selected Areas in Communications, Vol. 11, No. 3, 1993.

[25] M. Zajeganovic-Ivancic, I.S. Reljin, and B.D. Reljin, "Video Multicoder with Neural Network Control", In proceedings of the 9th Symposium on Neural Network Applications in Electrical Engineering, NEUREL, Belgrade, Serbia, 2008.

[26] R. Gvili, A. Kaplan, E. Ofek, and G. Yahav, "Depth keying", SPIE, California, 2003.

[27] M. Eid, J. Cha, and A. El Saddik, "HugMe: A Haptic Videoconferencing System for Interpersonal Communication," IEEE International Conference on Virtual Environments, Human-Computer Interfaces, and Measurement Systems (VECIMS), pp. 5-9, Istanbul, Turkey, July 14-16, 2008.