High Resolution Live Streaming with the HYDRA Architecture

ROGER ZIMMERMANN, MOSES PAWAR, DWIPAL A. DESAI, MIN QIN, HONG ZHU
University of Southern California, Integrated Media Systems Center

Presently, digital continuous media (CM) are well established as an integral part of many applications. With High Definition (HD) displays becoming increasingly common and large network bandwidth available, high quality video streaming has become feasible and novel, innovative applications possible. However, the majority of the existing systems for HD quality streaming are based on offline content, and use elaborate buffering techniques that introduce long latencies. Therefore, these solutions are ill-equipped for interactive real-time applications. Also, due to the massive amount of data required for the transmission of such streams, simultaneously achieving low latency and keeping the bandwidth low are contradictory requirements. Our HYDRA project (High-performance Data Recording Architecture) focuses on the acquisition, transmission, storage and rendering of high resolution media such as HD quality video and multiple channels of audio. HYDRA consists of multiple components to achieve its overall functionality. Here we elaborate on the live streaming capabilities of HYDRA that enables media streaming across an IP based network with commodity equipment.

Categories and Subject Descriptors: C.2.4 [**Computer-Communication Networks**]: Distributed Sytems - *Distributed applications*; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces - *User-centered design*; *Interaction styles*; *Input devices and strategies*.

General Terms: Streaming, high definition media.

Additional Key Words and Phrases: Human-computer interaction, latency, remote performance.

1. INTRODUCTION

Presently, digital continuous media (CM) are well established as an integral part of many applications. With High Definition (HD) displays becoming increasingly common and large network bandwidth available, high quality video streaming has become feasible and novel, innovative applications possible. However, the majority of the existing systems for HD quality streaming are based on offline content, and use elaborate buffering techniques that introduce long latencies. Therefore, these solutions are ill-equipped for interactive real-time applications. Also, due to the massive amount of data required for the transmission of such streams, simultaneously achieving low latency and keeping the bandwidth low are contradictory requirements. Here we present the live streaming component of our High-performance Data Recording Architecture (HYDRA) project. HYDRA has been motivated by our research in the area of Distributed Immersive Perfor-

This research has been funded in part by NSF grant EEC-9529152 (IMSC ERC) and unrestricted cash / equipment gifts from Intel, Hewlett-Packard and the Lord Foundation.
Author's current address: R. Zimmermann, Integrated Media Systems Center, University of Southern California, 3740 McClintock Avenue, Suite 131, Los Angeles, CA 90089-2561; e-mail: rzimmerm@imsc.usc.edu

mances (DIP). DIP investigates the feasibility of achieving a collaborative performance with musicians that are physically located in distributed geographical places. In such a scenario, multiple streams of different media types (e.g., audio and video) must be transmitted between all the participants. At the same time, the events need to be archived in real-time, as they occur. Hence, the need arises to capture and store these streams with an efficient data stream recorder that can handle both recording and playback of many streams simultaneously and provide a central repository for all data. Here we focus on the live streaming capabilities of the HYDRA system that enables HD quality video and multiple channels of audio to be streamed across an IP based network with commodity equipment. This has been made possible due to the technological advancements in capturing and encoding of HD streams. Our goal was to produce an architecture that integrates live streaming along with multi-stream recording by adapting and extending proven algorithms where applicable, and introducing new concepts where necessary. The project raises practical issues such as loss recovery, buffer management, playback latency optimization and multiple stream synchronization.

The rest of this report is organized as follows. In Section 2 we will briefly introduce the overall HYDRA architecture and then detail its live streaming component. In Section 3, we will elaborate on the experiments that we have done so far. Related research work is discussed in Section 4. Finally, Section 5 concludes this report and discusses some of our future research directions.

Figure 1: Two IMSC students, Dwipal A. Desai and Moses Pawar, are watching a live, high definition video stream with a two-way setup in the laboratories of USC.

2. APPROACH

The design and implementation of the recording and playback capabilities that are at the core of the design of the HYDRA architecture are described in detail elsewhere [Zimmermann 2004]. Here, we provide a short introduction that highlights some of the challenges and issues to provide the reader a better understanding of how the storage and archival component of HYDRA relate to its live streaming.

2.1 HYDRA Design Overview

Currently, digital continuous media (CM) form an integral part of many applications. Two of the main characteristics of such media are that (1) they require real time storage and retrieval, and (2) they require high bandwidths and space. Over the last decade, a considerable amount of research has focused on the efficient retrieval of such media for many concurrent users. Scant attention has been paid to servers that can record such streams in real time. However, more and more devices produce direct digital output streams. Hence, the need arises to capture and store these streams with an efficient data stream recorder that can handle both recording and playback of many streams simultaneously and provide a central repository for all data. The HYDRA project focuses on the design of a large scale data stream recorder. Our goal was to produce a unified architecture that integrates multi-stream recording and retrieval in a coherent paradigm by adapting and extending proven algorithms where applicable, and introducing new concepts where necessary. The project investigates practical issues such as the support of multizone disk drives, variable bit rate media, and disk drives that have a different write than read bandwidth.

The environment that we are considering envisions a scalable data stream recorder operating in an IP network environment. Multiple, geographically distributed sources, for example video cameras, microphones, and other sensors, acquire data in real time, digitize it and send it to the stream recorder. We assume that the source devices include a network interface and that the data streams are transmitted in discrete packets. A suitable protocol for audio and video data traffic would be the Real-time Transport Protocol (RTP) [Schulzrinne 1996] on top of the Universal Datagram Protocol (UDP). The client-recorder dialog that includes control commands such as record, pause, resume, and stop is commonly handled via the Real-time Streaming Protocol (RTSP) [Schulzrinne 1998].

The data stream recorder includes two interfaces to interact with data sources: (1) a *session manager* to handle RTSP communications, and (2) multiple *recording gateways* to receive RTP data streams. A data source connects to the recorder by initiating an RTSP session with the session manager, which performs the following functions: (1) admission control for new streams, (2) maintaining RTSP sessions with sources, and (3) managing the recording gateways. As part of the session establishment the data source receives detailed information about which recording gateway will handle its data stream. Media packets are then sent directly to this designated gateway, bypassing the manager. Multiple recording gateways are supported by each stream recorder, providing scalability to a large number of concurrent streams and removing the bottleneck caused by having a single entry point for all packets. Specifically, each recording gateway performs the following functions: (4) handling of errors during transmissions, (5) timestamping of packets, (6) packet-to-storage-node assignment and routing, and (7) storage node coordination and communication. A recording gateway forwards incoming data packets to multiple *storage node*s. Each storage node manages one or more local disk storage devices. The functions performed in the storage nodes are (8) packet-to-block (P2B) aggregation, (9) memory buffer management, (10) block data placement on each storage device, (11) real time disk head scheduling, and (12) retrieval scheduling for outgoing streams.

The HYDRA architecture also includes a mode of operation which is called *pass-through*. The pass-through mode can be enabled for each stream individually. In effect, it combines the recording and playback functionalities and provides an immediate live forwarding of the stream data (the data may still concurrently be recorded). The media data that is transmitted in this manner can be viewed at a remote location with a rendering unit. Figure 1 shows a demonstration of the system in two different rooms on the campus of the University of Southern California. The live streaming capabilities and its challenges are described in the next section.

2.2. HYDRA Live Streaming

Figure 2 illustrates the block diagram of the HYDRA high definition live streaming system (without the storage components). High resolution cameras capture the video that is then sent over an IP network to the receiver. Audio can be transmitted either by connecting microphones to the camera and multiplexing the data with the video stream, or by sending the sound as a separate stream. The transmission subsystem uses the RTP protocol and implements selective retransmissions for packet loss recovery. The streams are decoded at the receiver side to render the video and audio.

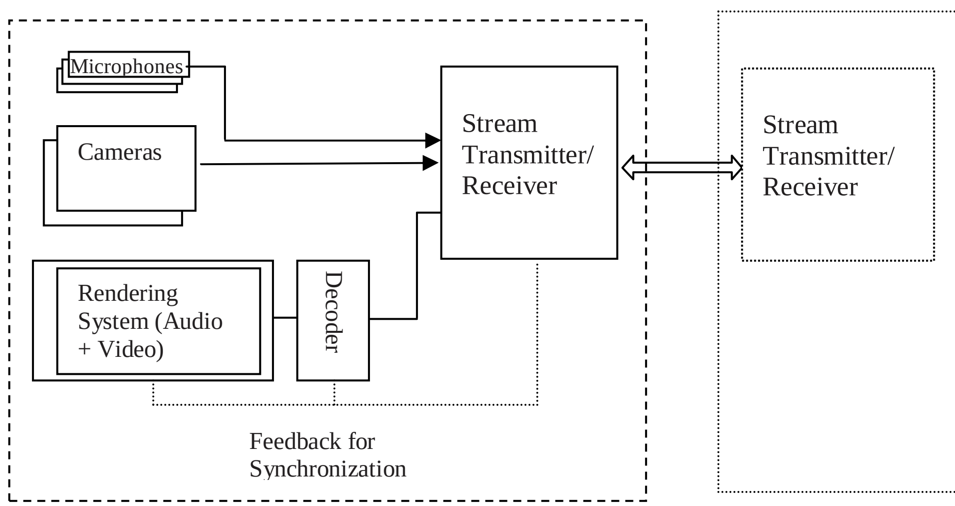

Figure 2: The HYDRA live streaming block diagram. Multiple cameras/microphones are connected to the stream transmitter, which is connected to the receiver node via an IP network. The audio and video streams are then decoded and rendered.

Our current implementation includes a camera interface that acquires digital video from a JVC JY-HD10U camera via FireWire (IEEE 1394) in HDV format (1280x720 pixels at 30 frames per second). The MPEG-2 data produced by the camera is encapsulated on an IEEE 1394 isochronous channel according to the IEC61883-4 “Digital Interface for Consumer Audio/Video Equipment” standard and must be extracted. The resulting MPEG transport stream is packetized and can then be transmitted at approximately 20 Mb/s over traditional IP networks such as the Internet. At the client side, the received data stream is displayed through either a software or hardware decoder. Section 2.3 elaborates on the decoding challenges.

The system uses a single retransmission algorithm [Papadopoulos 1996; Zimmermann 2003] to recover lost packets. Buffering is kept to a minimum to maintain a low transmission and rendering latency. Our design can be extended to support multiple simultaneous video streams along with multi-channel sound (i.e., immersive 10.2 channel audio). The issues related to the synchronization of such streams are currently being investigated. As noted earlier, the system integrates with the HYDRA recording system, which focuses on recording of events that produce a multitude of high bandwidth streams.

Figure 3: High definition video streaming equipment. Top: A Shuttle XPC computer complemented with a JVC JY-HD10U camera and a DLP projector are the minimally required equipment at each user location. (In the picture, the local video of the camera is displayed.) Bottom: The setup is quite portable – from left to right: the camera, the projector, and the computer bag.

2.3 HD Video Rendering

We are currently experimenting with the JVC JY-HD10U camcorder to achieve high definition resolution. This camera includes a built-in MPEG-based codec capable of both encoding and decoding approximately one megapixel (i.e., 1280x720) images at a rate of 30 frames per second. The compressed data rate is approximately 20 Mb/s and hence can be stored onto a DV tape in real-time. This format is called HDV (www.hdv-info.org) and a number of manufacturers have announced their support for it.

The decoding functionality of the JY-HD10U electronics is used to display recorded tapes on the camera LCD viewfinder and also to produce the analog Y-Pb-Pr signals that allow the connection of an HDTV directly to the camcorder. The digital data stream from the camera is only available in compressed form on the built-in FireWire port. As a consequence, if the media stream is transmitted over a network, the rendering component requires a MPEG-2 HD decoder. Various hardware and software options for decoding of streams are considered to achieve the best quality video with minimal latency. HYDRA currently employs the following two solutions:

1. Hardware-based: When improved quality and picture stability are of paramount importance we use the CineCast HD decoding board from Vela Research. An interesting technical aspect of this card is that it communicates with the host computer through the SCSI (Small Computer Systems Interface) protocol. We have written our own Linux device driver as an extension of the generic Linux SCSI support to communicate with this unit. An advantage of this solution is that it provides a digital HD-SDI (uncompressed) output for very high picture quality and a genlock input for external synchronization.
2. Software-based: We use the libmpeg2 library – a highly optimized rendering code that provides hardware-assisted MPEG decoding on current generation graphics adapters. Through the XvMC extensions of Linux' X11 graphical user interface, libmpeg2 utilizes the motion compensation and iDCT hardware capabilities on modern graphics GPUs (e.g., Nvidia). This is a very cost effective solution. For example, we use a graphics card based on an Nvidia FX 5200 GPU that can be obtained for less than $100. In terms of performance this setup achieves approximately 70 fps @ 1280x720 with a 3 GHz Pentium 4. Figure 4 shows the fan-less graphics card in our Shuttle XPC computer.

Figure 4: MPEG-2 decoding is achieved with a Nvidia FX 5200 based graphics card. The Linux supplied drivers accelerate the iDCT and motion compensation steps required for MPEG-2 decoding.

Table 1 illustrates the measurements that we performed with the software decoder based on the libmpeg2 library. Two subalgorithms in the MPEG decoding process ─ motion compensation (MC) and inverse discreet cosine transform (iDCT) ─ can be performed either in software on the host CPU (labeled SW in Table 1) or on the graphics processing unit (GPU, labeled HW in Table 1). The tests were performed on a dual Xeon 2.6 GHz processor Hewlett-Packard xw6000 workstation with an NVIDIA Quadro NVS 280 AGP graphics accelerator. The grey fields indicate real time (and better) performance. As can be seen from the results, real time decoding is possible with hardware assist.

Video Format	ATSC 1080i	ATSC 720p	HDV 720p
Frame resolution	1920 x 1080	1280 x 720	1280 x 720
Frames per second	30 (60 interlaced)	60 progressive	30 progressive
Compressed bandwidth	40 Mb/s	40 Mb/s	20 Mb/s
Rendering parameters:			
SW MC & iDCT	17.90 fps	30.37 fps	31.35 fps
HW MC & iDCT	33.48 fps	63.28 fps	67.76 fps

Table 1: Software decoding of MPEG compressed HD video material with the fast libmpeg2 library. Two subalgorithms in the MPEG decoding process ─ motion compensation (MC) and inverse discreet cosine transform (iDCT) ─ can be performed either in software on the host CPU (SW) or on the graphics processing unit (GPU; HW). Grey fields indicate real time or better performance.

2.4 Audio Rendering

The JVC JY-HD10U cameras include a built-in stereo microphone as well as a stereo jack that allows external microphones to be connected. The acquired audio signals are compressed according to the MPEG-1 layer 2 standard (ISO/MPEG IS-11172). The resulting audio bitrate is 384 kbps at a sampling frequency of 48 kHz. We currently use an open source software decoder, mpg123, to render the audio channels via a soundcard.

2.5 Delay Measurements

One of the most critical aspects when transmitting live, interactive media streams is the end-to-end latency that is experienced by the participants. The technical tradeoffs are between the following parameters: (1) video quality (this is mostly determined by the resolution of each video frame, the number of bits for each of the three transmitted colors, and the number of frames per second), (2) transmission bandwidth, and the (3) end-to-end delay. The ideal transmission system would achieve very high quality while requiring minimal bandwidth and producing low latency. However, in a real world design, a system must find a good compromise among these three parameters. Sometimes the available devices will constrain some of the design space. For example, the JVC JY-HD10U camera produces good picture quality. The built-in MPEG compressor is very effective considering that the camcorder was designed to run on battery power for several hours. The unit produces a reasonable output bandwidth of 20 Mb/s. The high compression ratio is achieved by removing temporal redundancies across groups of six video frames (also called inter-frame coding in the MPEG standard with a group-of-picture (GOP) size of 6). Therefore, video frame data must be kept in the camera memory for up to six frames resulting in a (6 f)/(30 fps) = 0.2 second camera latency. Figure 5 illustrates the complete end-to-end delay across the following chain: camera (sensor & encoder) – acquisition computer – transmission – rendering computer (decoder) – display. For this experiment we directed the camera at a stop watch application running on a computer screen. We then took still pictures showing both the stop watch application and the rendered video of it. The achieved latency for video only is approximately 310 milliseconds, which, considering the complex computations involved in coding/decoding HD MPEG-2 is surprisingly good. We repeated this experiment a number of times. Note that the monitor refresh rates of approximately 60 Hz (i.e., 16.7 ms per frame) can sometimes result in blurry snapshots of the stop watch with a resolution of 10 ms.

The bottom picture in Figure 5 illustrates the video delay when audio rendering is enabled at the receiver. The audio data, coded in MPEG layer 2 format, must be extracted from the combined MPEG transport stream. In our current implementation, this stream splitting introduces additional delay for the video. As shown in the photograph, the video delay rises to about 440 milliseconds. We believe that with further optimizations of the rendering code the video latency can be reduced to a delay closer to video-only decoding.

The above solution is probably suitable for applications where interactive communication involves discussions between two or more people, e.g., business meetings. However, for other scenarios, such as a musical interaction between remote locations as described in Section 3.2, the delay is too long. Our preliminary experiments show that players experienced the highest difficulty in creating a tight ensemble at 50 ms and above. Therefore, we are currently investigating other design points in the parameter spectrum. For example, uncompressed video acquisition and rendering would drastically reduce the delay at each end point. On the other hand, the transmission bandwidth required will be vastly increased.

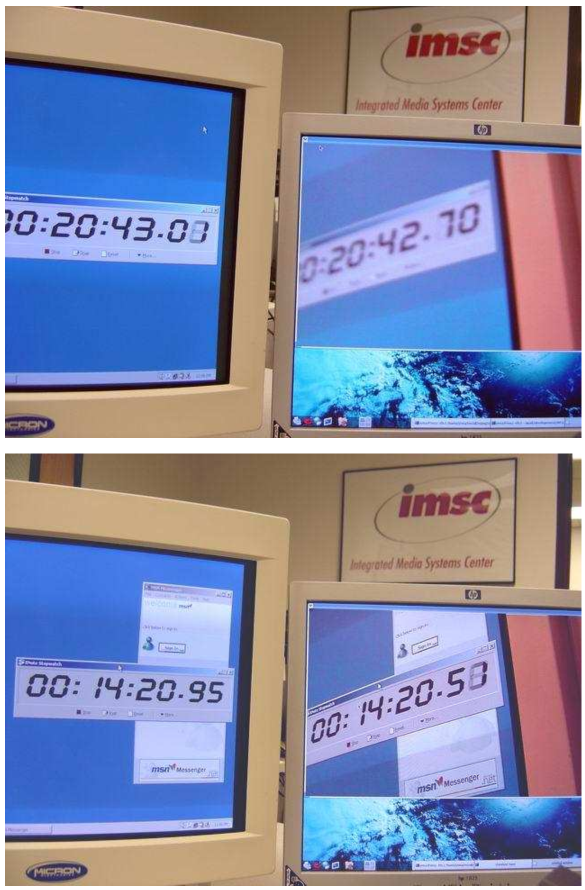

Figure 5: The top picture shows the latency of the complete video chain: camera – acquisition computer – transmission – rendering computer – display, with audio rendering disabled. For this experiment we directed the camera at a stop watch application running on a computer (left). The camera rendering is shown on the right display. As can be seen, the stop watch reads 20:43:01 while the transmitted time shows 20:42:70. The difference of 0.31 seconds represents the end-to-end video latency.

The bottom picture shows the same setup with audio rendering enabled at the receiver side. Because the audio data must be extracted from the arriving MPEG transport stream via a splitter, the video is slightly more delayed. In the shown experiment the delay was approximately 0.44 sec.

The audio delay was measured with the setup shown in Figure 6. An audio source produced a sound signal which was then split with a Y-cable into two paths. The signal on the test path was sent to the microphone inputs on the camera and then passed through the HYDRA HD live streaming system. Note that the camera compressed incoming audio signals into a perceptually coded MPEG layer 2 format at 384 kbps. The audio signal was converted back to its analog form in the rendering computer via a regular sound card. The measuring device recorded two audio signals arriving via the path directly from the source and the test path.

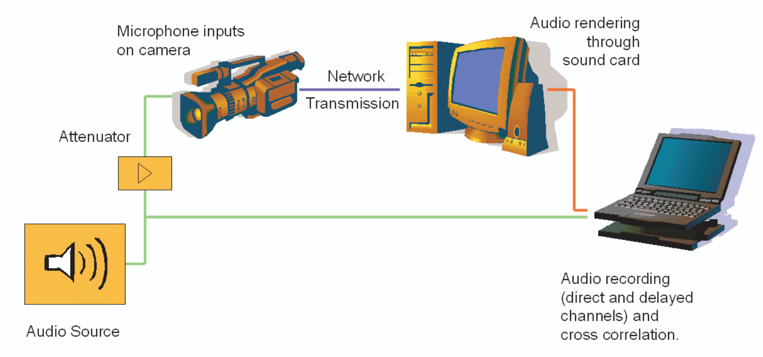

Figure 6: The setup for audio delay measurements.

Figure 7 shows the original (top) and delayed (bottom) audio signals acquired by the measuring recorder. Due to effects of compression/decompression, the delayed audio signal looks slightly different than the original waveform. To compare the two signals we used the cross-correlation estimate function for WAVE files available in Matlab. The program is shown in Figure 8. By repeating the measurements several times we obtained delays of between 17,000 to 18,000 samples. With a sampling rate of 48 kHz, these results correspond to audio delays of 354 to 375 msec.

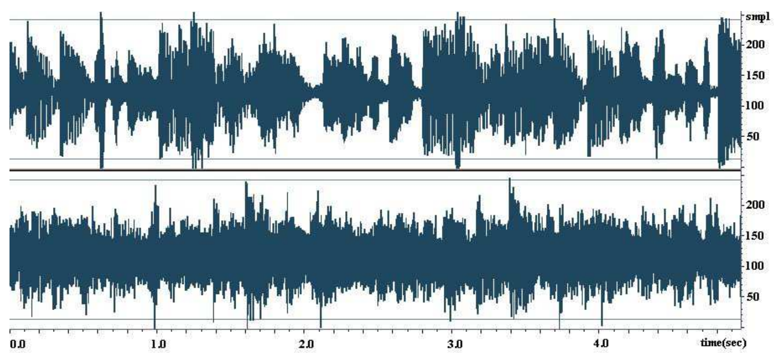

Figure 7: The original signal (top) and the transmitted (delayed) audio signal.

With both audio and video rendering enabled, the audio delay was slightly less than the video delay: 375 msec versus 440 msec. This translates to about 2 frames of lip-sync mismatch. We believe that the HYDRA software can be further optimized such that the video delay is very close to the audio delay. Recall that video only decoding can be achieved with 310 msec latency. For comparison purposes we also investigated other solutions that are available, such as the VideoLAN and XINE media players. However, because these applications were not designed for live streaming and employ quite significant buffering, latencies in excess of 1 second were measured. This makes them quite unsuitable for our purposes.

```
chorLR = wavread('wave_filename');
samplerate=48000;
chorL=chorLR(:,1);
chorR=chorLR(:,2);
cL=chorL(1e4:1e5);
cR=chorR(1e4:1e5);

[xL,iL]=max(xcorr(cL,cR));
[xNew,iNew]=max(xcorr(cR,cR));
distance=iL-iNew;
delay=distance/samplerate;
```

Figure 8: The Matlab program used for delay measurements via cross-correlation function estimate.

3. EXPERIMENTS

During the last year we have concentrated on the design and initial prototype implementation and evaluation of the live streaming system. Some of the highlights of the past year were as follows.

On Thursday January 29, 2004, we used the HYDRA live streaming system to demonstrate a two-way videoconferencing setup between USC and the University of Hawaii (a distance of approximately 5,000 km). During the 10-minute linkup, two of Prof. Zimmermann's students in IMSC's data management research laboratory — Dwipal A. Desai and Moses Pawar — explained the technical details of the system to researchers attending a meeting of the Asia-Pacific Advanced Network (APAN) Consortium in Honolulu. Figure 9 shows two pictures of the event. (Note that these are screenshots taken from an HD video, not still photographs).

Figure 9: Two IMSC students, Dwipal Desai and Moses Pawar, are watching a live, high definition video stream at USC transmitted from the APAN meeting in Honolulu. They later interacted with the audience in Hawaii through this bi-directional setup.

3.1 Distributed Immersive Performance

One of the key initiatives at USC's Integrated Media Systems Center (IMSC) is a real-time and multi-site distributed interactive and collaborative environment called Distributed Immersive Performance (DIP) [Sawchuk 2003]. The DIP project is an embodiment of IMSC's goal to develop the technologies of *immersive* and other *integrated media* systems through research, engineering, education and industrial collaborations [McLeod 1999]. Our vision of *immersive technology* is the creation of a complete audio and visual environment that places people in a virtual space where they can communicate naturally even though they are in different physical locations. The DIP project investigates a comprehensive framework for the capture, recording and replay of high-resolution video, audio and MIDI streams in an interactive environment for collaborative music performance, and user-based experiments to determine the effects of latency in aural response on performers' satisfaction with the ease of creating a tight ensemble, a musical interpretative and adaptation to the conditions. As a system, it facilitates new forms of creativity by enabling remote and synchronous musical collaborations. Musical collaboration demands a level of fidelity and immediacy of response that makes it an ideal testbed for pushing both the limits of human perception as well as technology innovation. The performance of such a system can be measured and quantified through the capture, replay and analysis of musician interaction and the digital music signals they create. The experiments mark the beginning of our efforts to study comprehensively the effects of musical interaction over the Internet in a realistic performance setting.

In our current experiments, illustrated in Figure 9, we use the HYDRA live streaming components to capture and monitor high-definition video of both musicians. In this setup, the full HYDRA architecture with its recording components is employed to archive video, audio and MIDI streams synchronized with common time stamps. The users (and evaluators) of this musical collaboration environment are a professional piano duo, Vely Stoyanova and Ilia Tosheff (the Tosheff Piano Duo, www.tosheffpianoduo.com), award-winning expert musicians who have been performing together since 1997. By engaging a professional piano duo for the experiments, we can forego the effects of learning in the analyses. In the first set of experiments, the duo performed each of the three movements of Poulenc's *Sonata for Piano Four-hands* with zero visual delay (in the same room, seated across from each other) and varying degrees of controlled audio delay (between 0 ms and 150 ms). In the second set of experiments, the duo swapped parts to test for symmetry in the effects of sensory delay on the performers. Musician satisfaction with the interactive environment is documented through a questionnaire, and their ensemble synchrony will be quantified computationally in future analyses based on the comprehensive set of MIDI data obtained.

Figure 9: Vely Stoyanova and Ilia Tosheff (of The Tosheff Piano Duo) performing audio latency experiments in one of USC's laboratories.

One of the main challenges of synchronous collaboration over the network is the effect of audio and video latency and reduced physical presence on ensemble synchrony as well as musical interpretation. In our preliminary results the users judged that, with practice, they could adapt to audio delays below 65 ms. In our first two experiments the piano duo was seated directly across from each other and hence were experiencing no visual delay. The HYDRA live streaming system was primarily used for stream monitoring, since its inherent end-to-end delay is too high to make it suitable for video transmissions between remote performers. However, it would be very appropriate for recreating the performance event for an audience situated at a third location. Our future plans include the development of a fully networked DIP version that uses low latency video between performers and high resolution, higher latency video to the audience.

4. RELATED WORK

The goal of this project is to achieve high quality streaming with minimal latency such that it can be used for interactive applications. There are commercial systems available to stream media content, however, they are either more focused on low resolution or content that is stored offline. Such systems use very large buffers and other mechanisms to reduce the bandwidth requirement that increase the latency and are unsuitable for interactive applications. Microsoft's Windows Media, Apple's Quicktime and RealNetwork's RealOne are examples of such systems. Cable companies currently broadcast high definition TV programs. However, they have a very limited support for extensions that Internet based streaming can achieve, such as interactivity and high quality multi-channel audio. Our streaming system is specifically designed to integrate with the large-scale HYDRA recording architecture. Hence, the combination of low latency and high quality media streaming offered by HYDRA is not currently available in any other existing commercial systems.

On the research front, several implementations and demonstrations of HDV over IP were discussed as part of the HDTV video and HDTV Birds-of-Feather sessions at the 17th APAN (Asia-Pacific Advanced Network) Meetings/Joint Tech Workshop held on January 28 and 29, 2004. The workshop venue was the University of Hawaii in Honolulu, and Michael Wellings and James W. DeRoest from Internet2/ResearchChannel demonstrated HD and SD video-on-demand applications from Seattle to Hawaii, using the popular VideoLAN (www.videolan.org) media player. Over the last few years, the ResearchChannel has performed several advanced streaming demonstrations such as a 270 Mbps HD video transmission to Busan, Korea, a 19.2Mbps ATSC 1080 / 60i HD transfer from Seattle to Los Angeles, and an uncompressed HD transmission, featuring 1080/60i HD video at 1.5 Gigabits as Packet over SONET IP from Seattle to Denver. Most of these demonstrations required very specialized and costly equipment.

At the same workshop, K. Okamura from Kyushu University and Reiji Aibara from Hiroshima University presented a HDV live streaming system that also uses Victor's JY-HD10 camera. The implementation was Linux based and the receiver used special PCI cards to decode MPEG2 to HDTV. The talk presented the implementation details and a local demonstration with both the camera and the display at the workshop venue was shown. The HYDRA system was also demonstrated with a two-way transmission between Los Angeles and Honolulu. Several other groups in Korea (based at K-JIST and KAIST) and Japan are working on HDV streaming systems and on integration issues related to the AccessGrid project (www.accessgrid.org).

Other research groups have experimented with distributed collaborative performance environments. One of the latest initiatives takes the form of a Berlin-Paris network concert to take place at the eighth International Cultural Heritage Informatics Meeting (ICHIM) in August, 2004 [ICHIM 2004]. Previous experiments include a Network Jam session between Stanford's SoundWIRE Group and McGill University on 13 June, 2002 [Stanford's SoundWIRE], and live demonstrations (a distributed Trio) and presentations documented by Eve Schooler [Schooler]. As a departure from the previous one-time demonstration/performances and a step towards rigorous study of performance in the time-delayed environment, the Stanford SoundWIRE group has conducted several experiments to quantify the effects of collaboration over the Internet by analyzing the ensemble accuracy of two persons clapping a short but inter-locking rhythmic pattern [Schuett 2002, Chafe 2004]. As far as we know, our experiments involving professional musicians performing complex composed pieces is the first such evaluation experiments on a realistic scale.

5. CONCLUSIONS AND FUTURE RESEARCH

We have presented a live streaming framework specifically targeting very high resolution, multi-channel video and audio transmissions to enable activities in immersive environments. We plan to continue working on the technical aspects of our system, specifically targeting (1) the integration of the live streaming components with our high-speed immersive media stream recorder prototype that will allow recording, archiving and playback of multiple streams of different media types, and (2) the reduction of the latency among the participants. By extending its capabilities and evaluate its performance, we expect that HYDRA will be able to support the requirements of the Distributed Immersive Performance project. Up-to-date information can be found on our home page at dmrl.usc.edu/hydra.html.

The Distributed Immersive Performance experiments mark the beginning of a series of experiments to study and understand the effects of network delays and a virtual environment on musical ensemble, interpretation and adaptability. Future studies will incorporate detailed analyses of the users' comments as well as quantitative measures of musical synchronization derived by computational means. Eventually, we plan to extend our experiments to three- and n-way collaborative, immersive activities.

ACKNOWLEDGMENTS

First and foremost, we wish to thank our musical collaborators, Vely Stoyanova and Ilia Tosheff, the Tosheff Piano Duo. The investigators working on the Distributed Immersive Performance project include Alexander A. Sawchuk, Elaine Chew, Chris Kyriakakis, Christos Papadopoulos, Albert "Skip" Rizzo, Alexandre R.J. François, Gerry Kim, Ania Volk, and Victor LaCour. We also acknowledge the help of numerous students who assisted in the project, including, Hui-Yun Frances Kao, Dwipal Desai, Kanika Malhotra, William Meyer, Moses Pawar, Min Qin, Shiva Sundaram, and Hong Zhu. We also thank Seth Scafani and Allan Weber for providing technical support throughout the project. We further would like to credit Dan Dennedy, on whose code our MPEG transport stream extraction module is based. Last but not least, the DIP experiments were made possible by NSF grants MRI-0321377 and Cooperative Agreement No. EEC-9529152. Any Opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the National Science Foundation.

REFERENCES

CHAFE, C., GUREVICH, M., LESLIE, G. and TYAN, S. 2004. Effect of Time Delay on Ensemble Accuracy. Proceedings of the International Symposium on Musical Acoustics, March 31st to April 3rd 2004 (ISMA2004), Nara, Japan.

CHEW, E., ZIMMERMANN, R., SAWCHUK, A.A., KYRIAKAKIS, C., PAPADOPOULOS, C., FRANÇOIS, A.R.J., KIM, G., RIZZO, A. and VOLK, A. 2004. Musical Interaction at a Distance: Distributed Immersive Performance, submitted to the *4th Open Workshop of MUSICNETWORK: Integration of Music in Multimedia Applications*, Universitat Pompeu Fabra, Barcelona, Spain, 15-16 September.

DASHTI, A. E., KIM, S. H., SHAHABI, C., and ZIMMERMANN, R. 2003. Eds., *Streaming Media Server Design*. Prentice Hall IMSC Press Multimedia Series, March 2003. ISBN: 0-130-67038-3.

ICHIM 2004 – Eighth International Cultural Heritage Informatics Meeting. Berlin, August 30 – September 2, 2004. – *http://www.ichim.org/jahia/Jahia/lang/en/*.

MCLEOD, D., NEUMANN, U., NIKIAS, C.L. and SAWCHUK, A.A. 1999. Integrated Media Systems. *IEEE Signal Processing Magazin*e, vol. 16, no. 1, pp. 33-76, January.

PAPADOPOULOS, Ch. and PARULKAR, G. M. 1996. Retransmission-based Error Control for Continuous Media Applications. In Proceedings of the *6th International Workshop on Network and Operating Systems Support for Digital Audio and Video* (NOSSDAV 1996), Zushi, Japan, April 23-26.

SAWCHUK, A. A., CHEW, E., ZIMMERMANN, R., PAPADOPOULOS, Ch., and KYRIAKAKIS, C. 2003. From Remote Media Immersion to Distributed Immersive Performance. In Proceedings of the *ACM SIGMM 2003 Workshop on Experiential Telepresence* (ETP 2003) November 7, Berkeley, California, USA. In conjunction with *ACM Multimedia 2003*.

SCHOOLER, Eve. Musical Distractions – *http://www.async.caltech.edu/~schooler/music.html*.

SCHUETT, N. 2002. The Effects of Latency on Ensemble Performance. Masters Thesis, Stanford CCRMA, May.

SCHULZRINNE, H., CASNER, S., FREDERICK, R., and JACOBSON, V. 1996. RTP: A Transport Protocol for Real Time Applications. URL: http://ww.itef.org/rfc/rfc1889.txt.

SCHULZRINNE, H., RAO, A., and LANPHIER, R. 1998. Real Time Streaming Protocol (RTSP). URL: http://ww.itef.org/rfc/rfc2326.txt.

STANFORD's SoundWIRE Group – *http://ccrma.stanford.edu/groups/soundwire/*.

SHAHABI, C., ZIMMERMANN, R., FU, K., and YAO, S.-Y. D. 2002. Yima: A Second Generation Continuous Media Server, *IEEE Compute*r, vol. 35, pp. 56–64, June.

ZIMMERMANN, R., FU, K., and KU, W.-S. 2003. Design of a Large Scale Data Stream Recorder. In Proceedings of the *5th International Conference on Enterprise Information Systems* (ICEIS 2003), (Angers, France), April 23-26.

ZIMMERMANN, R., FU, K., SHAHABI, C., YAO, S.-Y. D., and ZHU, H. 2001. Yima: Design and Evaluation of a Streaming Media System for Residential Broadband Services. In Proceedings of the *VLDB 2001 Workshop on Databases in Telecommunications* (DBTel 2001), (Rome, Italy), September.

ZIMMERMANN, R., FU, K., NAHATA, N., and SHAHABI, C. 2003. Retransmission-Based Error Control in a Many-to-Many Client-Server Environment. In Proceedings of the *SPIE Conference on Multimedia Computing and Networking 2003* (MMCN 2003), (Santa Clara, California), pp. 34–44, January 29-31.

ZIMMERMANN, R. 2003. Streaming of DivX AVI Movies. In Proceedings of the *ACM Symposium on Applied Computing* (SAC 2003), (Melbourne, Florida), March 9-12.

ZIMMERMANN, R., FU, K. and DESAI, D. A. 2004. HYDRA: High-performance Data Recording Architecture for Streaming Media. Book chapter in *Video Data Management and Information Retrieval*, editor Sagarmay Deb, University of Southern Queensland, Toowoomba, QLD 4350, Australia. Published by Idea Group Inc., publisher of the Idea Group Publishing, Information Science Publishing and IRM Press imprints.

ZIMMERMANN , R., KYRIAKAKIS, C., SHAHABI, C., PAPADOPOULOS, Ch., SAWCHUK, A. A., and NEUMANN, U. 2004. The Remote Media Immersion System. Published in the *IEEE MultiMedia* magazine, special issue on "Digital Media on Demand," April.