Adaptive, Best-Effort Delivery of Digital Audio and Video Across Packet-Switched Networks*

K. Jeffay, D.L. Stone, T. Talley, F.D. Smith
University of North Carolina at Chapel Hill
Department of Computer Science
Chapel Hill, NC 27599-3175 USA
{jeffay,stone,talley,smithfd}@cs.unc.edu

Abstract: We present an overview of a "best-effort" transport protocol that supports conferencing with digital audio and video across interconnected packet switched networks. The protocol delivers the highest quality conference service possible given the current load in the network. Quality is defined in terms of synchronization between audio and video, the number of frames played out of order, and the end-to-end latency in the conference. High quality conference are realized through four transport and display mechanisms and a real-time implementation of these mechanisms that integrates operating system services (*e.g.*, scheduling and resource allocation, and device management) with network communication services (*e.g.*, transport protocols). In concert these mechanisms dynamically adapt the conference frame rate to the bandwidth available in the network, minimize the latency in the displayed streams while avoiding discontinuities, and provide quasi-reliable delivery of audio frames.

1. Introduction

The focus of this work is on the real-time transmission of live digital audio and video across interconnected packet-switched networks. Our goal is to support high-fidelity audio/video conferences, *e.g.*, conferences with high quality audio and full-motion color video. The approach one adopts for the real-time communication of audio and video will depend on numerous factors including the architecture of the audio/video subsystem, the encoding and compression technologies employed in the audio/video subsystem, the available network bandwidth, and the degree of physical layer support in the network for real-time communication. As a starting point we consider existing local area networks (*i.e.*, ethernets, token rings, and FDDI rings) interconnected by bridges or routers, and an audio/video system that acquires and compresses individual frames of video at NTSC rates.

One technical challenge in this environment is to manage and accommodate jitter in the audio/video streams seen by the processes responsible for displaying the streams. In order to sustain a high-fidelity conference, frames of audio/video data must arrive at a displaying workstation so as to be played at a precise rate (*e.g.*, one frame every 33 ms for NTSC video). Although these frames can be generated at the desired rate by the originating audio/video hardware, the exact rate at which frames arrive at a receiver can be grossly distorted by poor operating system scheduling and resource allocation on the transmitting and receiving workstations, and varying load in the network. Jitter is problematic because it can cause discontinuities in the playing of audio/video streams and can increase the latency of the conference. Jitter therefore directly impacts conference quality.

---

* This work supported by the National Science Foundation (grant numbers CCR-9110938 and ICI-9015443), and by the Digital Equipment Corporation and the IBM Corporation.

In: *Network and Operating System Support for Digital Audio and Video*, Proceedings, V. Rangan (Ed.). Lecture Notes in Computer Science, Vol. 712, pp. 3-14, Springer-Verlag, Heidelberg, 1993.

It is instructive to view a video conferencing system as a distributed pipeline. Frames of audio/video data are acquired, digitized, and compressed at one machine, transmitted over a network to a second machine where they are decompressed and displayed. Ideally each stage of this pipeline should operate in real-time, that is, each stage should process a frame before the previous stage outputs the following frame. Through the use of a real-time operating system that provides computation and communication services specifically tailored to the needs of applications that process continuous-time media streams such as digital audio and video [4], we can closely control the processing of audio/video frames on a transmitting workstation to ensure that frames progress from acquisition and compression hardware to the network interface with minimal jitter. Correspondingly, on the receiving machine, we ensure the arriving frames are delivered the audio/video subsystems for display with jitter no worse than in the arriving stream. Unfortunately, in our network environment we cannot control how the network is used by others and hence we cannot provide this level of control over the transmission of individual frames.

We have developed a set of techniques for the transmission and display of audio/video frames that yield the highest quality conference possible given the conditions present in the network. First, in the transport layer, audio is transmitted redundantly (*i.e.*, units of audio data are transmitted multiple times). This allows a conference receiver to function well in the face of packet loss. Second, real-time scheduling and queue management techniques are used in the transport and network interface layers to manage conference latency and automatically adapt the data rate of the conference to varying network bandwidth. At a conference receiver the display of audio and video data follows a protocol to (further) manage conference latency and ameliorate deviations in inter-arrival times of audio/video data. The third mechanism dynamically varies the synchronization between displayed audio and video. This allows a receiver to trade off synchronization between audio and video for low conference latency and the ability to accommodate large deviations in interarrival times of audio/video data. Lastly, a data aging mechanism is used to reduce the latency in a conference after periods of high network load. Together, these mechanisms form a transport and display protocol for conferencing. It is argued that these mechanisms, in concert, provide the best possible conference service given the available network bandwidth at any point in time.

These techniques have been implemented in a multimedia transport protocol (MTP) that has been used to transmit live digital audio and video across packet-switched networks using Internet (IP) protocols [7]. We have investigated the effects of these transport and display techniques on the quality of the audio and video streams displayed at the receiver under a variety of network configurations and loading conditions. In this paper we survey two of the techniques we have developed to ameliorate the effects of jitter. These are the dynamic variation of audio/video synchronization and the adaptation of the conference frame rate to that currently sustainable in the network. The following section presents our requirements for audio/video transmission and describes the technologies on which our work is based, and describes the salient communications problems this work addresses. We follow in Section 3 with a description of the transport and display mechanisms that address jitter. We conclude in Section 4 with a review of our results.

2. Requirements and Methodology

2.1 Transport and Display Requirements

Users judge conferences by three metrics: fidelity of, and latency in, the displayed audio and video streams, and the synchronization between displayed audio and video. We measure fidelity of a stream in terms of the number of discontinuities that occur when the stream is displayed. A stream of audio/video is a totally ordered sequence of frames. A *frame* is the logical data unit by which audio/video is acquired and displayed. A *discontinuity* in a displayed stream occurs when frame n is not followed by frame $n + 1$. Since we cannot control how the network is used by others, we cannot bound the number of discontinuities that will occur over any interval in time. The goal of our transport and display mechanisms is simply to minimize the number of discontinuities in the audio and video streams.

Related to fidelity is the synchronization between audio and video frames (*i.e.*, "lip sync"). Our mechanisms bound the amount of synchronization loss between displayed audio and video streams. The third metric used to judge a conference is end-to-end stream latency. Factors such as the lengths of the hardware audio and video pipelines, the latency in the operating system and network influence latency in our environment. Our protocol endeavors to ensure an end-to-end (acquisition to display) latency of no more 250 ms [2]. Our approach to latency management is described in [7].

Jitter control is required to minimize the number of discontinuities in the displayed audio and video streams. The actual requirements for allowable discontinuities differ dramatically between audio and video. As most communication in a conference is spoken conversations, users demand much more of the audio used in a conference than they demand of the video. Moreover, because there is less temporal coherence in the audio stream than in the corresponding video stream, users are quite sensitive to discontinuities in the audio caused by late or lost packets. In contrast, users can tolerate significant discontinuities in the video stream. For example, if video frame n is not present at the receiver when it is time for it to be played, the preceding frame, frame $n - 1$, can be played in its place without adverse effect. By contrast, if a corresponding amount of audio data (*e.g.*, 33 ms worth) does not arrive on time, a replay of the previous 33 ms of audio data (or the playing of silence) is easily noticed.

While audio frames could have a duration significantly less than 33 ms, it is paramount to note that the requirement for reliable transmission of audio is independent of the size of an audio frame. *The important measure is not the duration of an audio frame but rather the total duration of all audio frames that are sent in a single network packet as this defines the amount of audio data that can potentially be lost or delayed in the network.* As the load in a network increases, the critical resource becomes access to the network. If the time to access the network increases beyond the audio frame generation time, a sender must transmit multiple audio frames in a network packet. If a packet is lost or delayed, a discontinuity will occur with a duration equal to the cumulative lengths of the audio frames in the lost packet. In our experiments we have routinely observed network access times exceeding 33 ms. Under these conditions, discontinuities perceivable to the user occur. For this reason we require tighter control of the jitter in the audio stream.

2.2 Experimental Configuration

We have constructed a tested for experimenting with the transmission of live digital audio and video across IP networks. For audio and video acquisition and display we use IBM-Intel ActionMedia 750 hardware and IBM PS/2 workstations. The PS/2s run a real-time operating system we have developed (YARTOS [5]) and support UDP/IP. The network configuration used for experiments consists of two 16 megabit token rings interconnected by a series of 10 megabit ethernets. Packets are initially transmitted on one token ring, routed to an ethernet, bridged to a second ethernet, routed to a second token ring and then displayed.

The ActionMedia system uses a two stage pipeline to acquire and display frames of video. The acquisition pipeline consists of digitization and compression stages; the display pipeline consists of decompression and display stages. The acquisition and display stages are 33 ms long. The length of the compression and decompression stages is a function of scene complexity and the algorithm employed. In our use of the system these stages typically are in the range 20-25 ms. A compressed, color video frame (at 240x256 resolution) is between 6000-8000 bytes. Under program control, each video frame is digitized, compressed, transmitted over the network, received, decompressed, and displayed as shown in Figure 1. Audio frames are digitized and compressed in real-time (*i.e.*, in a single stage). The length of this stage is programmable. To reduce processing overhead, we typically use values of 16.5 or 33 ms. A 33 ms audio frame is a little over 500 bytes. The data rate for a full-fidelity audio stream is approximately 128 kilobits per second.

There is a significant disparity between the lengths of the audio and video pipelines. The acquisition and compression stages run sequentially for each video frame. Therefore, the earliest a video frame can be transmitted is approximately 60 ms after it is introduced to the system (assuming no operating system or application overhead). In contrast, the audio data corresponding to a video frame can be transmitted 33 ms after entering the system. A key facet of our transport and display mechanisms will be an exploitation of this disparity.

3 Transport and Display Mechanisms

We have implemented an unreliable connection-oriented stream transport protocol, called MTP (Multimedia Transport Protocol), on top of UDP/IP. The novel aspects of the protocol are in the way data is managed within the transport layer. We adopt a real-time systems philosophy in the design and implementation of the protocol. By this we mean that we provide the transport layer with information on the timing characteristics and semantics of the data to be transmitted, so that MTP can manage the resources available to it (*e.g.*, CPU cycles, buffers, network transmission slots) to meet the real-time requirements of applications. This implies that the transport layer is more tightly integrated with higher layers (*e.g.,* the application layer) and lower layers (*e.g.*, the network interface layer) than in traditional protocols. For example, when an MTP connection is established, the application specifies (1) audio and video frame times, (2) number of buffers MPT should use for queueing audio and video frames, and (3) the number of times audio frames should be transmitted. The frame times are used by the transport layer to compute deadlines for transmitting packets, and to intelligently combine audio and video data into network packets. Packets are scheduled at the network interface using a deadline-based scheduling algorithm [6].

The application controlling the audio and video generation processes invoke an asynchronous transmit operation whenever an audio or video frame has been generated. Outgoing frames are queued in the transport system. The protocol manages separate queues for audio and video frames. Priority is given to audio frames and, to the extent possible, audio frames are never fragmented.

The essence of this approach is to view the audio/video acquisition processes, the network subsystem, and the display processes as a distributed pipeline as shown in Figure 2. At the sender and receiver, the stages of the ActionMedia pipeline can be tightly synchronized (given sufficient real-time support from the operating system) and data can be made to flow between stages as it is generated (*i.e.*, with 0 or 1 buffers). The network transmission and reception processes cannot be so controlled and hence queues must be maintained in the network subsystem to interface the network with the audio/video processes. The jitter control mechanisms we develop are a set of queue manipulation policies that address the issues of what data should be enqueued/dequeued and when it should be enqueued/dequeued. Given the time-critical nature of audio and video data, queueing decisions should be made as late as possible. For example, the ideal time to decide which audio and video should be transmitted next is to wait until access to the network is actually obtained (*e.g.*, when a free token arrives at a station on a token ring). Similarly, the ideal time to decide which frame of video should be displayed is during the vertical blanking interval of the display. Delaying decisions as long as possible maximizes the amount of data that is available to make a decision. The implication of this is that the transport protocol must be integrated with the network device driver.

The two jitter control mechanisms we present address the issue of *what data* to enqueue and dequeue. There are two queues to manage: a transport queue at the sender that contains compressed audio and video frames that are ready to be transmitted, and a display queue at a receiver that contains complete audio and video frames that are ready to enter the display pipeline. We distinguish between two types of jitter: jitter caused by short-term increases in network load (*e.g.*, bursts), and jitter caused by longer-term increases in network load. In the former case we ameliorate the effects of jitter by buffering audio frames at the display and allowing the audio stream to be played out of synchronization with the video stream (*e.g.*, play the audio ahead of its corresponding video) when frames arrive late. In the latter case, the bandwidth available to the conference may decrease below that required to support full-fidelity audio and video streams. In this case the conference frame rate must be adapted to that currently sustainable in the network.

For purposes of illustrating the effect of each mechanism on jitter, we use an audio frame time of 33 ms and a video frame time of 66 ms. For video this means that 33 ms of video (*i.e.*, an NTSC frame) is acquired during each 66 ms interval. At the receiver each video frame will be displayed for 66 ms. This results in a video frame rate of 15 frames per second.

3.1 Display Queue Management

The goal in display queue management is to (1) minimize the number of discontinuities in the displayed audio/video streams, (2) display audio/video data with minimal latency, and (3) bound the synchronization differential between displayed audio and video. These goals typically cannot be achieved simultaneously and hence there are trade-offs between latency, discontinuities, and lip synchronization that must

be managed. Here we concentrate on one queue management technique: dynamically varying audio/video synchronization.

Consider the problem of bursty network traffic. Network bursts increase latency and cause discontinuities by introducing jitter. Figure 3 shows the effect of a network burst on video latency. Figure 3 plots latency versus conference duration (measured in number of frames displayed). Instantaneous increases in latency indicate discontinuities. Latency is broken down into:

- the time spent in the ActionMedia pipeline at the originating machine,
- the time spent in the transport protocol layer (including time spent in the transport queue),
- the time spent at the network interface waiting to access the network,
- the physical network transmission time,
- the time spent queued at the receiver waiting to enter the display pipeline, and
- the time spent in the display pipeline itself.

The time spent at the network interface is combined with the physical transmission time into a measure of network congestion called *per frame network access time*. This is the total time required to transmit an audio or video frame as measured from the time the transport system invokes the network-adapter device-driver to transmit the first packet containing data for the frame until the last packet containing data for the frame has been transmitted. On the receiver, the time spent waiting to enter the display pipeline is further broken down into two components:

- the time spent synchronizing with the display pipeline, and
- the time spent queueing at the entrance to the display pipeline.

We make a distinction between these two components of latency because the former is beyond the control of the display software while the latter is directly controllable. A frame of video is displayed every 33 ms, synchronized with the vertical blanking interval of the display. When a frame arrives at a receiver it may have to wait up to 33 ms before it can enter the display pipeline. Therefore, even if video frames were transmitted instantaneously from sender to receiver, audio and video latency may differ between two otherwise identical conferences by as much as 33 ms. The time spent waiting to synchronize with the display pipeline is called the *display pipeline synchronization time* (DPST). In order to compare the effects of transport and display policies across different executions of the system, the DPST must be factored out. Therefore we explicitly represent this term. The time a frame spends queueing at the entrance to the display pipeline is defined as the time spent queued on the receiver minus the DPST. By definition this time will be a multiple of a frame time.

In Figure 3 video is initially being displayed with latency of 260 ms. After approximately 80 video frames have been generated, load is introduced in the network. The effect of this load is to increase the network access time at the sender and delay the arrival of frames at the receiver. The initial increase in network access time is compensated for by a decrease in the DPST at the receiver and thus the latency in the stream remains constant. As frames arrive later and later, the DPST eventually becomes 0 indicating that frames are entering the display pipeline immediately upon arrival. At this point the next frame that arrives late (*i.e.*, more than 66 ms after the previous frame) will cause a discontinuity in the video stream and an increase in latency. As more traffic is introduced into the network, the network access time

continues to increase. For frames 80-100, the access time has increased from approximately 30 to 70 ms per frame (recall that multiple network accesses are required to transmit a video frame). This increase results in a queue of unsent frames at the sender indicating that the sender is no longer able to transmit data in real-time. If no action is taken and the load in the network does not subside, frames will be queued at the sender for longer and longer durations. This is illustrated by an increase in the time spent in the transport queue starting around frame 70. Eventually a second discontinuity and corresponding increase in latency occurs. By the time the 100th frame is displayed, frames are arriving at the receiver approximately 150 ms after being generated and are displayed with 133 ms more latency than the initial frames.

As the network burst subsides the sender is able to transmit faster than real-time and empties its queue. However, since the receiver can only display frames at a constant rate, a queue forms at the display. Moreover, because the receiver displays frames at a constant rate, the latency in the conference does not change. A temporary increase in the network access time of between 10-15 ms (per packet) results in a sustained increase in latency of 133 ms. However, the conference is now resilient to future network bursts of equal magnitude. After frame 125, the frames queued at the receiver insulate the display process from additional bursts. In general, the increase in network access time required to induce a discontinuity in a displayed stream will be a function of the current display queue length and the amount of time required to synchronized with the display pipeline. If there is no display queue, *i.e.*, frames are displayed with minimal latency, a stream displayed with a DPST of t ms will have a discontinuity if a frame arrives more than t ms late. If one frame is queued at the receiver then the receiver can tolerate the late arrival of a single frame by at most one frame time plus the current DPST.

The network burst in Figure 3 increased video latency and created discontinuities. If audio is synchronized with video then the audio stream suffers equally. The effects of bursts on the audio stream can be avoided by exploiting the disparity in the lengths of the audio and video pipelines and allowing the receiver to play audio frames before their corresponding video frames are displayed. Figure 4 shows the effect of the network burst on audio latency. Note that audio frames are queued at the display as each frame must wait for its corresponding video frame to be decompressed. Because of this, there are a sufficient number of audio frames in the display queue to play audio without any discontinuities during the burst. Note that after a network burst, audio is being played with 133 ms less latency than the video, *i.e.*, the audio is 2 video-frame times ahead of its corresponding video.

By dynamically varying the synchronization between the audio and video streams we are able to minimize discontinuities in the audio stream due to network bursts. This technique has two limiting effects. First, assuming a situation with stable network load and audio/video displayed with minimum possible latency, audio can be played ahead of video by no more the difference in the lengths of the combined acquisition/display pipelines for audio and video plus the difference in the network access times for audio and video frames. Audio can be played ahead of video by at most 66 ms when the video stream is displayed with minimum latency. Therefore, if audio and video are initially synchronized, the audio stream can tolerate temporary increases in network access times of at least 66 ms per frame (plus the current DPST) without creating an audio discontinuity and corresponding increase in latency. If there

is additional latency in the video stream due to queueing or high per frame network access (as in Figure 3) then audio can be played ahead of video by more than 66 ms.

The second factor limiting the deviation in audio video synchronization is the limit of human perception and tolerance of synchronization between audio and video. Audio should be played ahead of video only to the extent that audio discontinuities are perceived as more annoying the than loss of audio/video synchronization. We conjecture that users will tolerate audio being played by 100 ms ahead video. Informal user studies indicate that this is the level at which the loss of synchronization is first noticeable in our system (given a particular video compression algorithm and its resulting image quality).

3.2 Transport Queue Management

The goal in transport queue management is to (1) adapt the outgoing frame rates to the bandwidth currently available to the sender, (2) deliver audio and video data with minimal latency, and (3) ensure reliable delivery of audio frames. Again, there are trade-offs between these goals. For example, one can choose to maximize throughput or minimize latency. Here we concentrate on one transport queue issue: the transmission of audio frames multiple times to ameliorate the effect of packet loss in the network.

Ideally audio and video frames should be transmitted as they are generated. When the network access time to send a frame becomes larger than the frame time, queues form in the transport system. The lengths of these queues and the policies for managing them can dramatically effect the throughput and latency of the conference. For example, if a frame cannot be transmitted when it is generated, enqueueing the frame only serves to increase its latency. However, if frames that cannot be transmitted immediately are discarded, the throughput of the stream may unnecessarily decrease (with a corresponding increase in the number of discontinuities). When the network access time exceeds the frame time, transport queues of length greater than 1 cannot increase the throughput of frames (unless multiple frames can be transmitted in a single network packet) and only serves to increase the latency in the displayed stream. A transport queue of length greater than one is useful only if the additional latency it introduces can be tolerated by users of the system. If this is the case and if increases in network access times are relatively short lived, then longer queues are of some utility as they reduce the likelihood that the sender will drop a frame. More frames will be sent to the receiver who is in a better position to decide whether or not frames should be displayed or skipped (to reduce latency). Our experiments indicate that a transport queue of length 1 is useful in all cases and that there are (rapidly) diminishing returns to lengthening the queue. The conference quickly reacts to changes in available bandwidth without increasing latency.

The audio stream is not as susceptible to the effect of queue length since, for the networks we consider, multiple audio frames can be transmitted in a single packet. Should the available bandwidth decrease below that required for the full audio frame rate, more aggressive application level mechanisms, such as changing the coding or compression scheme, will have to be employed to sustain the conference.

3.3 Performance

We have used the transport and display techniques outlined above to transmit live audio and video across interconnected packet-switched networks. Here we describe

the results of one experiment: a conference spanning two token ring and two ethernet networks. We compare the performance of MTP to a straightforward use of UDP to transmit audio/video frames for the three measures of conference quality. The experiment uses an audio frame time of 16.5 ms, a video frame time of 33 ms, and a packet size of 5000 bytes. This gives a full 30 (video) frame per second conference with a uni-directional data rate of 2 megabits per second. In the MTP case, an MTP connection is established with a video transport queue length of 1 frame and an audio queue of 3 frames. Each audio frame is transmitted twice. At the receiver, audio and video is initially display in exact synchronization. Audio is never allowed to get ahead of video by more than 100 ms. In the UDP case, the conferencing application combines audio and video frames and presents them together to the transport system. This is done to minimize the number of packets required to transmit audio/video frames and thereby increase the resiliency of the transmission process to network bursts. This has the effect, however, of effectively requiring the receiver to display audio and video in exact synchronization. For the UDP conference the network device driver at the sender uses a queue of length 3.

Figure 5 shows the performance of UDP and MTP in the face of congestion on the sender's token ring. Figures 5a and 5b show latency and discontinuities. Vertical lines are plotted at 10 second intervals indicating the minimum and maximum latency during the interval. A line connects the mean in each 10 second interval. Figure 5c shows the synchronization loss in the displayed streams. It shows the amount (in 16.5 ms frame times) that audio was played ahead of video. Vertical lines again indicate the minimum maximum synchronization loss in a 10 second interval. A line connects the mean loss. The congestion in this experiment is due to file transfers and the simulation of additional conferences. UDP experiences large fluctuations in latency and hence has a high number (thousands) of audio/video discontinuities. Latency averages 300ms. MTP experiences much smaller fluctuations in video latency and has an average latency of approximately 250 ms. However, significantly more video discontinuities are experienced. This is because the overhead of transmitting audio redundantly occasionally increases the network access time per video frame beyond 33 ms and frames are dropped at the sender. However, this is not noticeable. (The average video frame rate is 28 frames per second.) MTP audio fidelity is quite good with only a small number discontinuities and an average latency of 225 ms. Audio is played an average of 16 - 33 ms ahead of video and is occasionally 100 ms ahead. The UDP conference is aesthetically quite poor; primarily because of the number of audio discontinuities. UDP drops 1600 audio frames at the sender while MTP drops only 1. This is because MTP treats audio and video separately and adapts its packetization scheme to available bandwidth. In spite of the fact that MTP has more video discontinuities than UDP, the MTP conference is comparable to a conference on an unloaded network.

4. Summary and Conclusions

Our goal is to support real-time communication of audio/video frames across interconnected packet switched networks. Our approach has been to understand the cost, and feasibility of, supporting multimedia conferencing without admission control, resource reservation, or real-time support from the physical layer. To this end, we have developed a "best effort" transport protocol that provides the highest quality conference service possible given the bandwidth available to the conference.

Quality is defined in terms of synchronization between audio and video, the number of frames played out of order, and the end-to-end latency in the conference.

We described two transport and display mechanisms that provide a measure of jitter control. The mechanisms are: a facility for varying the synchronization between displayed audio and video to achieve high-fidelity audio, and a network congestion monitoring mechanism that is used to control audio/video latency and dynamically adapt the conference frame rate to the bandwidth available in the network.

Our work demonstrates empirically that it is possible to establish and maintain high-fidelity conferences across congested packet-switched networks without requiring special services from the network such as admission control, jitter control [3], resource reservation [1], or isochronous transmission. While such services are clearly useful and important, they equally clearly come at an additional cost; be it the installation of new wiring or other network hardware, the modification of routing software at all network interconnection points, or a reduction in the total bandwidth available conferences. It is therefore useful to understand the cost of supporting high-fidelity conferencing (as measured in terms of the required workstation communications and application software) in the absence of such services. While it is not possible to precisely state this cost, we have provided evidence that, even in the absence of direct network support, high-fidelity conferences are achievable at modest cost.

5. References

[1] Anderson, D.P., Herrtwich, R.G., Schaefer, C., 1990. *SRP: A Resource Reservation Protocol for Guaranteed Performance Communication in the Internet*, University of California Berkeley, Dept. of Electrical Eng. and Computer Science Technical Report, TR-90-006, February 1990.

[2] Ferrari, D., 1990. *Client Requirements for Real-Time Communication Services*, IEEE Communications, (November), pp. 65-72.

[3] Ferrari, D., 1991. *Design and Application of a Jitter Control Scheme for Packet-Switch Internetworks*, 2nd Intl. Workshop on Network and Operating System Support for Digital Audio and Video, Heidelberg, Germany, pp. 81-84.

[4] Jeffay, K., Stone, D., Poirier, D., 1992. *YARTOS: Kernel support for efficient, predictable real-time systems*, in "Real-Time Programming," W. Halang and K. Ramamritham, eds., Pergamon Press, Oxford, UK.

[5] Jeffay, K., Stone, D.L., and Smith, F.D., 1992. *Kernel Support for Live Digital Audio and Video*. *Computer Communications*, 16, 6 (July), pp. 388-395.

[6] Jeffay, K., 1992. *Scheduling Sporadic Tasks with Shared Resources in Hard-Real-Time Systems*, Proc. 13th IEEE Real-Time Systems Symp., Phoenix, AZ, December 1992 (to appear).

[7] Jeffay, K., Stone, D.L., and Smith, F.D., 1992. *Transport and Display Mechanisms For Multimedia Conferencing Across Packet-Switched Networks*. In submission.

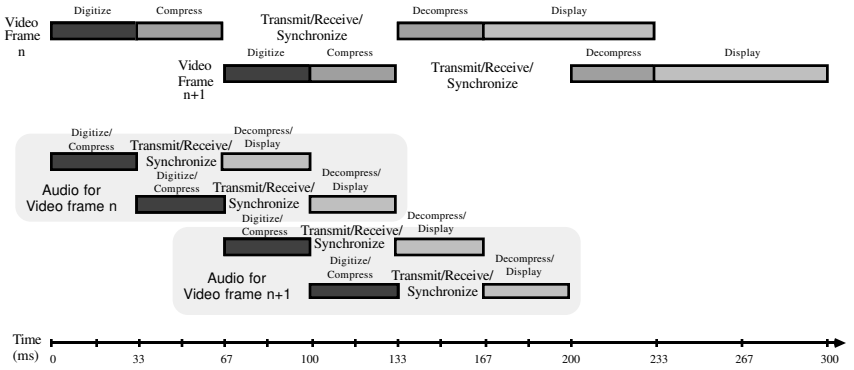


Figure 1 Hardware limits on audio/video synchronization (66ms video frame times, 33 ms audio frame times).

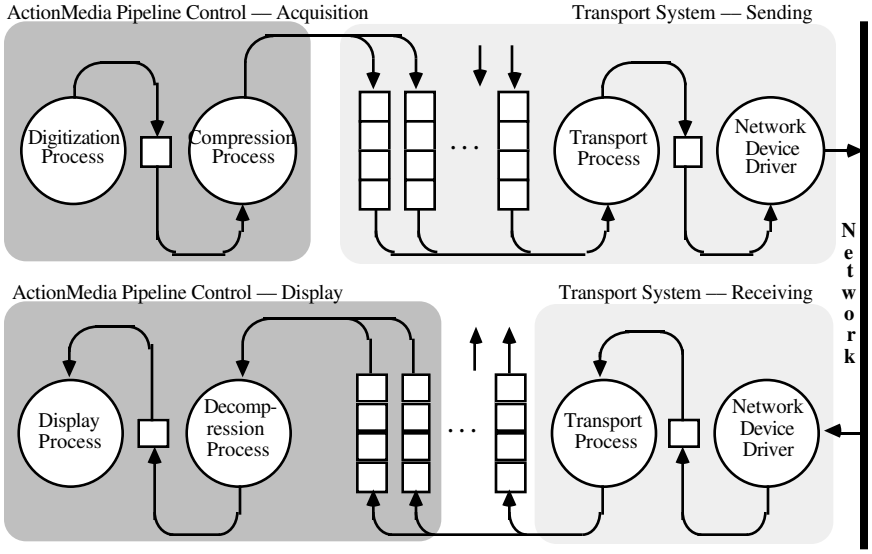


Figure 2 Delivery system architecture.

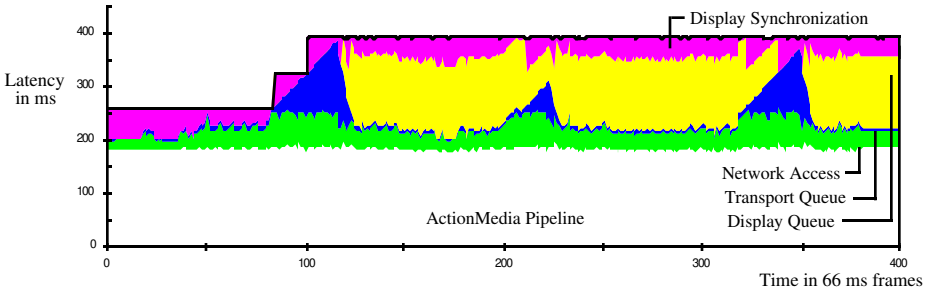


Figure 3 Video latency in the face of several network bursts.

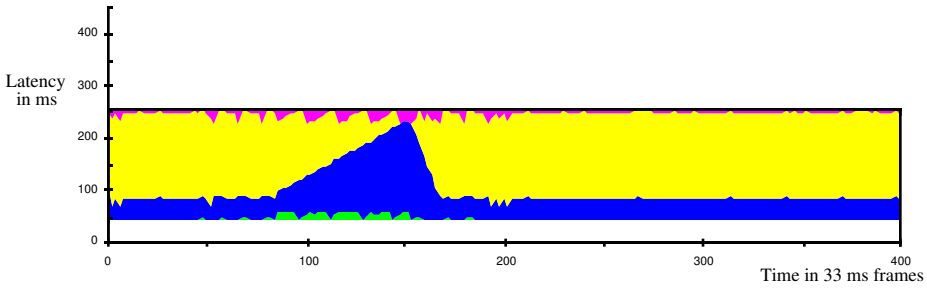
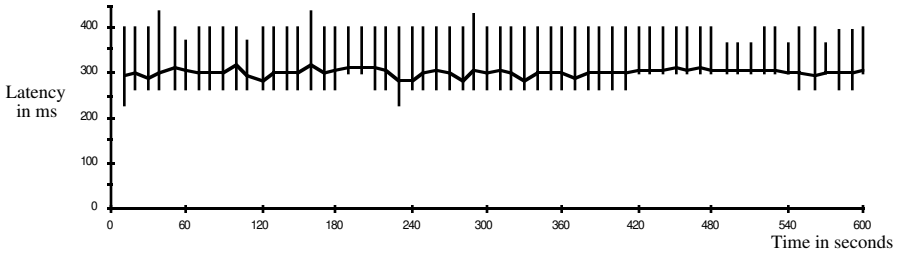
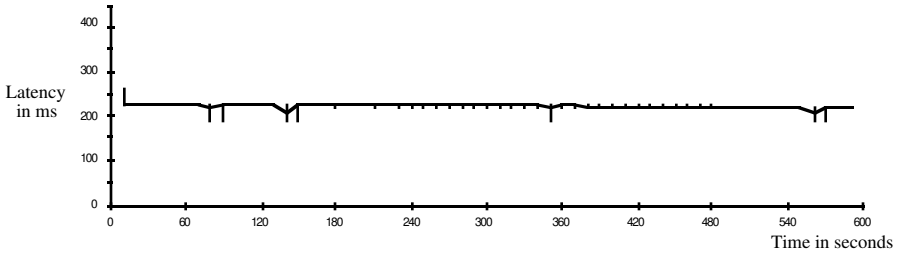


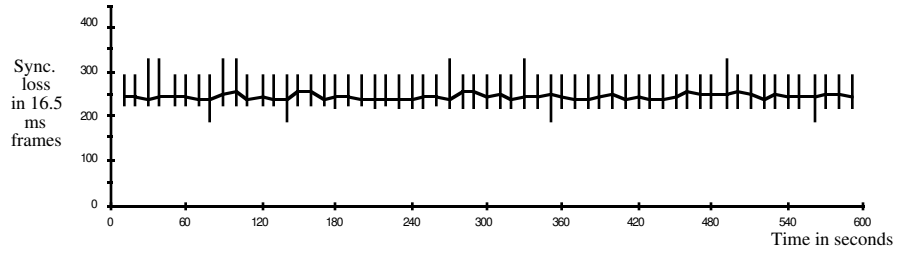
Figure 4 Audio latency in the face of a network burst.



(a) Audio/video latency (UDP).



(b) Audio latency (MTP).



(c) Video latency (MTP).

Figure 5: MTP performance, high token ring traffic.