ELSEVIER

Computer Communications 19 (1996) 901–914

computer communications

Packet loss and jitter control for real-time MPEG video communications

Inwhee Joe*

Broadband and Wireless Networking Laboratory, School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA 30332, USA

Abstract

Real-time video communication over a packet switching network is subject to packet loss and random delay variation called jitter. For compressed video, packet loss can cause significant performance degradation, and for continuous video regeneration, packet jitter can cause discontinuity and additional packet loss. Therefore, for good reception quality of video, a low jitter and a low loss are required. In this paper, a real-time video transport protocol on top of the UNIX UDP/IP is proposed. To reduce the delay jitter and packet loss effects, the protocol uses new mechanisms of such as selective packet discard, buffering, and constant rate playback for compressed MPEG (Moving Pictures Experts Group) video. Test results over Ethernet and under heavy traffic conditions show satisfactory performance which is unnoticeable from the ideal case.

Keywords: Delay jitter; MPEG; Packet loss; Video transport protocol; Real-time communication; Selective mechanism

1. Introduction

During the last few years we have seen two important technology advances that make cost-effective video communications a reality; high-speed local area networking and low-bit-rate video coding [1]. The availability of these technologies has motivated us to provide multimedia communications over networks. For example, we can provide video conferencing or video telephony by transmitting low-bit-rate compressed video over Ethernet [2], which is very attractive to industry and academia.

When packets are sent through a network, they experience different delays because of unpredictable traffic. The random variation of the delay is called *delay jitter*. For data traffic such as file transfer and remote login, real time transmission is not required. When packets are received, they are stored in the receiver buffer and processed later by the local computer. Therefore, delay jitter in general is not critical.

Real-time traffic such as video, on the other hand, requires a high timing accuracy of reproducing the original signal at the receiver end. For example, in real-time video communications, video frames are displayed at a fixed time interval. When a packet has an excessive delay, the display will be temporarily paused. This results in display discontinuity and has a direct impact on the user's perception.

In addition to discontinuity, jitter can also cause additional packet loss. For example, if a sequence of packets comes to the receiver close in time, the receiver buffer may not have space to hold all of them. As a result, some packets will be dropped. In order to achieve a good quality of reception, jitter is thus required to be kept below a certain upper bound.

In packet switching, packet loss can be also caused from network transmission. As the load on a network increases, packets may be lost due to various reasons such as buffer overflow and traffic congestion. Traditional communication protocols deal with packet loss through time-out and retransmission mechanisms. But these techniques cannot be used in real-time video communications because it introduces unacceptably long delay into the video streams. If a frame is segmented for transmission, loss of a packet can cause the entire frame useless, and result in discontinuities in reproducing the video.

The objectives of this paper are threefold. First, we want to investigate and quantify the effects of delay jitter and packet loss on video transmission. Second, from the study, we want to develop a real-time protocol scheme to

* Email: inwhee@ee.gatech.edu

PII S0140-3664(96)01124-3

minimize the jitter and packet loss effects. Third, we want to implement the protocol in an existing and widely used platform so that most users can benefit from the work.

To achieve the above objectives, we build our real time protocol upon existing User Datagram Protocol/Internet Protocol (UDP/IP) protocols in UNIX systems. Since UDP/IP is widely used and is a hardware independent protocol, the real-time video protocol developed can be used for communications between different systems, such as between UNIX workstations and DOS/Windows-based personal computers. To study the packet loss and jitter effects under various traffic conditions, we perform video packet transmission tests over Ethernet.

Fig. 1 shows the block diagram of key modules in real-time video communications. Some special hardware devices are required to construct the total system: a Moving Pictures Experts Group (MPEG) encoder, an MPEG decoder and a video camera. As soon as the input video signal comes from a video camera, it is compressed by the MPEG encoder, where MPEG is a recently established standard for full-motion video compression [3]. After network transmission through the Video Transport Protocol (VTP), it is decompressed by the MPEG decoder and then displayed in an X-window (in UNIX) or MS Windows (in DOS) of the receiver. In network transmission, there are three software modules involved:

- VTP protocol: a new protocol for real-time video communications running on top of UDP/IP. Jitter reduction and packet loss control schemes are introduced to minimize their effects on the displayed stream.
- UDP/IP protocol: UDP provides a transport-level datagram service. It is preferred to the Transmission Control Protocol (TCP), since it does not require acknowledgement and is thus fast in transmission. Like TCP, it is designed to work with IP, which is almost the standard for internetworking.
- X-video player: once decompression is done by the hardware, the X-video player displays the decoded video stream in an X-window.

The rest of the paper is organized as follows. Section 2 describes the data collection process to model the packet loss and jitter statistics, and to quantify their effects on real-time video transmission. Section 3 proposes some mechanisms to reduce the effects of packet loss and jitter. Also, a new protocol for real-time video communications, and related implementation issues, are discussed. In Section 4, we present the results of experiments through a real-time emulation method, which is used to emulate real-time video transmission due to the limitation of real time decoding by hardware. Finally, we conclude with a discussion of the possible extensions for the proposed mechanisms and future research directions.

2. Jitter and packet loss effect analysis

To design a satisfactory video transport protocol (VTP) for lossy real-time video communications, we

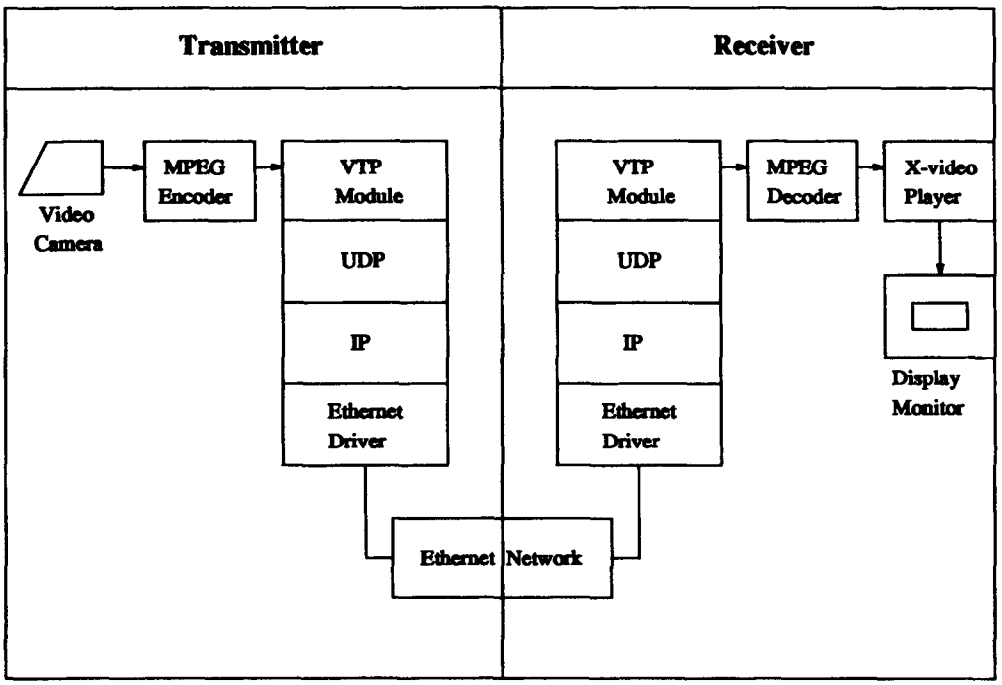


Fig. 1. Overall system structure.

first study the delay jitter and packet loss effects. In the study, we quantify the dependence of jitter and packet loss statistics on the background traffic, which is controlled by a separate traffic generator based on the Poisson traffic model.

As shown in Fig. 2, the traffic generator generates background traffic while video data is being transmitted from the transmitter to the receiver. From the experiment, we can obtain jitter and packet loss statistics at a given background traffic intensity. By performing another subjective test on final video quality, we can identify important conditions that cause poor video transmission quality and design an appropriate VTP.

2.1. Traffic generator

As mentioned earlier, the purpose of the traffic generator is to emulate an actual traffic conditions in Ethernet. For simple design, the packet generation process is based on the Poisson process of a certain arrival rate, and the packet duration follows the exponential distribution of a certain mean packet size. Since the inter-arrival time of a Poisson process follows another exponential distribution with mean equal to the inverse of the arrival rate, the traffic generator mainly generates two random variables for each packet transmission:

- inter-arrival time: T (msecs),
- packet size: L (bytes).

From T and L given, the network utilization in a 10 Mb/s Ethernet is:

$$U = \frac{8L}{10^4 T}. \tag{1}$$

In practice, due to limitations from the UNIX system, T cannot be smaller than 10 msecs. Therefore, we use

$$T = T_1 + 10 \text{ msecs}, \tag{2}$$

where T_1 is an exponential random variable. As a result, Eq. (1) is modified as

$$U = \frac{8L}{10^4 (T_1 + 10)} \tag{3}$$

As a numerical example, Table 1 gives some possible values of L at given values of T_1 when $U = 0.1$.

In experiments, for each utilization value from 10–50%, a set of L and T_1 values are used. Specifically, T_1 used is from 10–100 msecs, from which L can be determined at a given U. Since an Ethernet network usually operates at a utilization around 10% or less, the emulated utilization range from 10–50% is sufficient to cover a wide range of traffic conditions. Due to practical performance limits, each pair of UNIX workstations is restricted to contributing to only 10% of network utilization. As a result, up to five pairs of workstations are used to generate 50% background traffic.

There are two more practical considerations in traffic generation. First, since both T_1 and L are exponentially distributed, there is a possibility that the packet duration (with mean $L/10^4$ in msecs) can be longer than the interval (with mean $T_1 + 10$ in msecs) before the next packet generation. When this happens, the traffic generator sends out the following packet immediately after the current transmission.

Another practical consideration is which transport protocol we should choose for the packet generator. For its reliable data delivery and convenience in handling data fragmentation and reassembly, the built-in TCP/IP suite in UNIX is chosen. Although a certain amount of overhead is added from the TCP/IP protocol suite, its effect on the network utilization can be ignored, since it is less than 3% when the packet size over Ethernet is 1500 bytes long.

2.2. Video transmission model

In this section, a video transmission model working under the background traffic is described. As mentioned

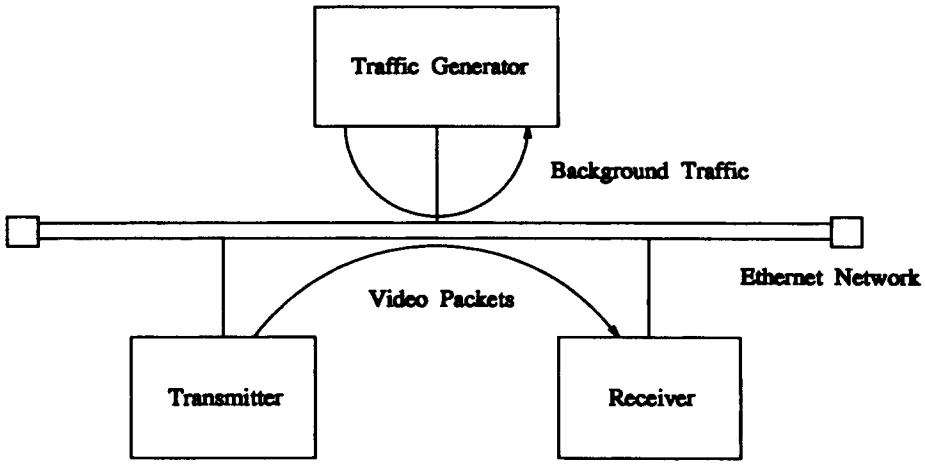


Fig. 2. Experiment setup for delay jitter and packet loss effect analysis.

Table 1
Mean inter-arrival time versus packet size at a 10% network utilization

T_1 (mseces)	L (bytes)	T_1 (msecs)	L (bytes)
10	2500	60	8750
20	3750	70	10000
30	5000	80	11250
40	6250	90	12500
50	7500	100	13750

in Section 1, video transmission experiments considered in the paper consist of three blocks: compressed MPEG video transmission; decompression at the receiver end; and data format conversion and display. This is illustrated in Fig. 3.

To transmit video, a sample MPEG file is chosen, which has a size of 160 × 128 bytes, with an average of 3384 bytes per frame after compression. The UDP/IP protocol is selected for data transmission for its simplified processing and short transmission delay (e.g. no need for positive acknowledgment and no retransmission, as in TCP/IP). For receiver synchronization in the event of packet loss, each UDP packet is used to carry one MPEG compressed frame.

To perform decompression, a public domain MPEG decoder developed by UC Berkeley is used [6]. After decompression, an additional dithering process is used to convert the 24-bit MPEG video to an 8 bit/pixel format for X-window display, where an X-window display routine is called.

To establish a real-time video transmission, we set a target framing rate of 10 frames/sec. This is chosen from a compromise between processing speed and real-time video transmission. To achieve this speed, we performed tests to measure the processing time of each task. The results are:

(1) Packet transmission time: average time is 3 msecs/frame.
(2) Decompression: average time is 166 msecs/frame.
(3) Data conversion from 24-bit MPEG images to 8-bit color pixels:
average time: 34 msecs/frame.
(4) X-window video display: average time: 18 msecs/frame.

From the above results, if we implement all tasks above in software, we can obtain a speed of only about 4.5 frames/sec, which is too low for real time video. If we can use hardware to process decompression, on the other hand, only 55 msecs/frame is needed, which allows us to achieve a framing rate of 10 frames/sec. To compensate a ±20 msec delay jitter as shown in Fig. 4, we use a framing interval of 95 msecs. This gives a framing rate slightly higher than 10 frames/sec. That is, the video source periodically transmits MPEG frames every 95 msec.

From the 95 msec framing interval specified, what is done in the sender application on top of UDP/IP is to send a compressed frame every 95 msec. When the current transmission takes more than its expected transmission time, the corresponding number of next packets expected to be sent should be discarded due to the real time constraints. What the receiver application does is different depending on the purpose of each experiment, so it will be explained case by case.

2.3. Measurement results and analysis

To measure jitter and packet loss of video packets under different background traffic intensities, experiments were conducted from 12:00 to 2:00 am during weekends for minimal uncontrollable traffic. The jitter and packet loss statistics are discussed below.

2.3.1. Jitter behavior

According to the video transmission model, the sender transmits packets every 95 msec, which produces an

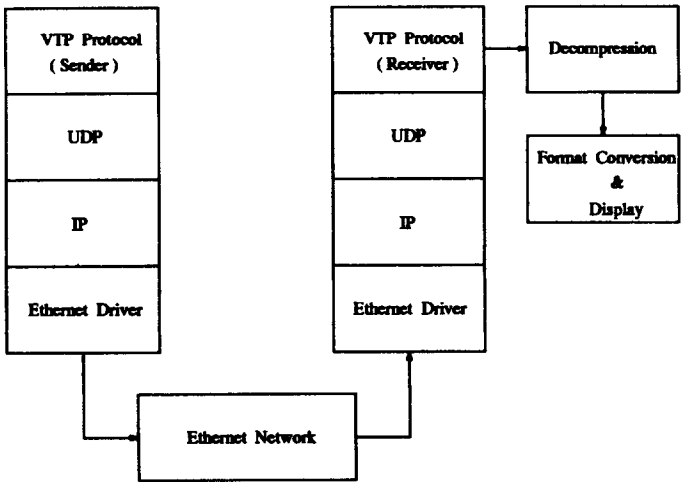


Fig. 3. Protocol structure for the video transmission model.

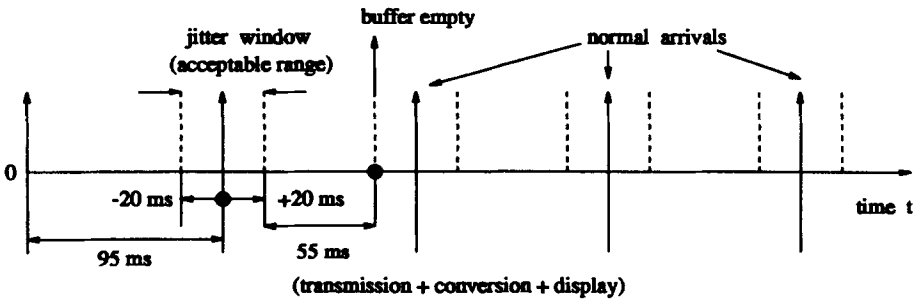


Fig. 4. Jitter timing diagram.

average of 285 Kb/s on the network. When a packet is received, the receiver records its arrival time. To separate the jitter and packet loss effects, each jitter measurement experiment sends only 400 packets. If there is any packet loss, the data will be discarded. In other words, only experiments with all packets successfully transmitted are taken into jitter measurement consideration. The number of 400 packets is chosen, since the probability of at least one packet loss is high as long as more than 400 packets are transmitted at a time.

The above measurement process is repeated 25 times to obtain a sample size of 10,000 (25 × 400). Specifically, from the arrival times measured, delay jitter is calculated as

$$m^{\text{th}}\textit{jitter} = T_m - E[T], \tag{4}$$

where $E[T]$ is the average inter-arrival time and T_m is the mth inter-arrival time.

As illustrated in Fig. 4, from the given 95 frames/sec transmission, if delay jitter is within the range from −20 to +20 msecs, there will be no packet loss. On the other hand, packets with delay jitter outside the range will be considered lost due to either receiver buffer overflow or too late for display in real-time video communications.

Table 2 gives the percentage of packet transmission from the total 10,000 samples that fall in different jitter ranges at a given emulated network utilization. The same results are also plotted in Fig. 5, which shows that the higher network utilization, the more widely the jitter distribution. From Table 2, we see the percentage of packets that have delay jitter within ±20 msecs is 96.2% at 50% utilization. Therefore, even at a very high traffic intensity, we can achieve a packet loss rate of less than 4% due to the jitter effect only.

In addition to the above jitter statistics, we also have the following observations of the jitter correlations for adjacent packets:

- If the jitter of one packet is within one unit time of 95 msecs, the adjacent packet usually arrives on time, as illustrated in Fig. 6(a).
- If the jitter exceeds one unit time, there will be a number of packets arriving in burst. For example, one unit time of jitter causes a bursty arrival of two packets, and two unit times of jitter cause a bursty arrival of three packets, and so on [7]. Fig. 6(b) shows one unit time of jitter case.
- The worst case of jitter observed from experiments is three unit times, which results in a bursty arrival of four packets, as illustrated in Fig. 6(c). By taking an additional unit time from transmission into consideration, the total delay is four unit times.

2.3.2. *Packet loss behavior*

To obtain packet loss statistics another measurement is performed, where 1000 packets are sent under various background traffic conditions according to the video transmission model. Compared to the previous 400 packet transmission experiment, this number of 1000 is chosen to obtain sufficient data on packet loss. This measurement repeats 100 times for a total of 100,000 packet transmissions.

At the receiver end, the receiver records every packet received and checks what packets are lost. Since the received packets are not decompressed at an interval of 95 msecs, packet loss is only due to network congestion that results in buffer overflow at either the transmitter or receiver. This cause of packet loss is defined as the *original* packet loss, in contrast to that due to a large arrival jitter (outside the ±20 msec jitter range). From the video transmission model, we know that the actual packet loss can be due to both buffer overflow and large arrival jitter. Therefore, packet loss due to both cases should be combined to give the *total* packet loss. From the experiment, the packet loss rate for the original case is shown in Fig. 8. By combining the packet loss rate due to jitter shown in Fig. 7, we have the total packet loss rate, as also shown in Fig. 8.

In addition to the packet loss rate results shown above, we have the following observations from the measured data:

- In a received packet sequence, there is a high probability (97.8%) of single, isolated packet loss from all packet loss scenarios.
- Even though it is rare, there is a probability of 2% or 0.2% for two or three consecutive packet losses, respectively.

Table 2
Percentage of occurrences in jitter range

		Network utilization					
		0%	10%	20%	30%	40%	50%
Jitter range	±10 msecs	95.7%	95.7%	95.6%	94.6%	94.3%	92.8%
	±20 msecs	98.1%	97.9%	97.7%	97.4%	97.0%	96.2%
	±30 msecs	98.7%	98.7%	98.5%	98.4%	98.3%	97.6%
	beyond ±30	1.3%	1.3%	1.5%	1.6%	1.7%	2.4%

- From data observed, the intervals between two packet loss bursts are large enough so as not to affect each other in terms of video quality, so they can be considered independent.
- From Fig. 8, we have approximately 9% of the total packet loss rate at 50% network utilization. This indicates the importance of jitter control for packet loss rate reduction.

2.4. *Subjective video quality experiments*

To further quantify the effects of packet loss and delay on video transmission, subjective tests are also performed. In these experiments, 15 people are asked to evaluate the video transmission quality effects under various conditions. This experiment provides information on how human eyes are sensitive to delay and packet loss. According to their satisfaction, they are asked to give a score from 4 (very satisfactory) to 0 (very unsatisfactory). The average value is then calculated to quantify the effects.

2.4.1. *Loss and delay effect of I-type frames only*

In the experiments, pictures are displayed every 95 msec according to the video transmission model. As mentioned earlier, this display interval can absorb jitter in the range of ±20 msecs. Since a frame loss corresponds to a single packet loss, a simple program was developed to simulate frame loss and delay. When running the program, one can specify the start position of packet loss, burst size, delay position and amount of delay.

(1) Loss effect only

The first subjective experiment takes only frame loss into consideration. In other words, the effect of delay is not considered and frames are displayed every 95 msec.

Under this condition, we first used a fast-motion video in the test. Depending on how many frames in the burst are lost, we have the following results:

- 1–2 consecutive frames loss: not recognizable (average score: 3.8).
- 3–7 consecutive frames loss: a small shift in motion can be noticed, but not bothering (average score: 2.7). This is called a small frame loss case.
- 8–12 consecutive frame loss: a relatively large jump in motion can be sensed and is a little bothersome (average score: 2.1). This is called a large frame loss case.
- 13 or more: a large discontinuity, and totally different after the burst loss, compared to the original picture (average score: 0.3). Therefore, this case is unacceptable.

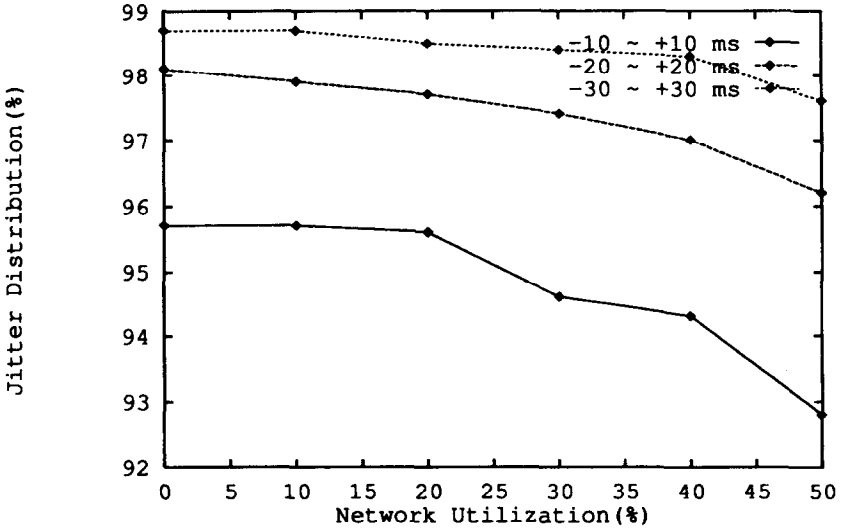


Fig. 5. Jitter distribution vs. network utilization.

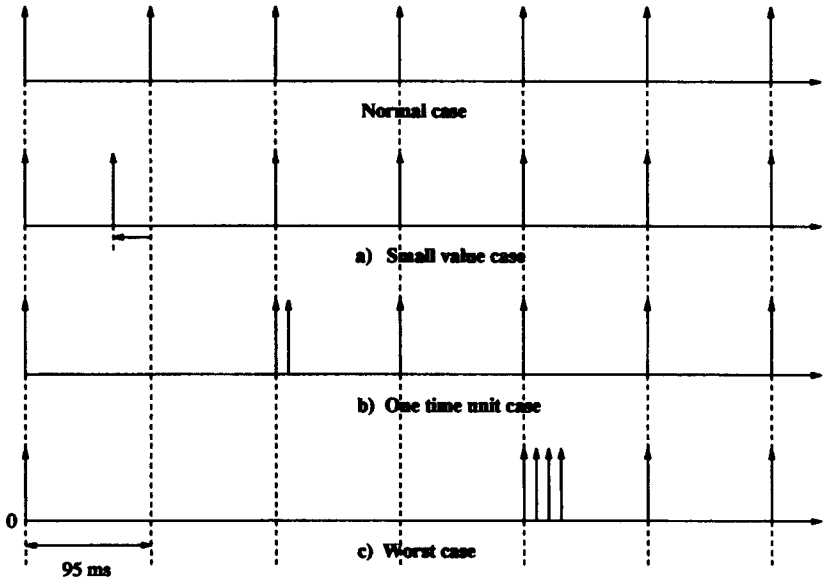


Fig. 6. Jitter correlations.

To make a comparison, we also performed a test using slow-motion video. In this case, only a small shift can be noticed, even we have a large (20) consecutive frame loss in the burst. When this slow-motion video is played at different rates, we found that the higher the rate, the more frame loss acceptable. This is because human eyes become more insensitive to frame loss as the framing rate increases.

(2) Delay effect only

In the second experiment, we consider only the packet delay effect. In this case, the video quality is evaluated at different delay values, where one unit delay considered below corresponds to 95 msecs, and several delay points are introduced randomly in each video stream:

- 1–5 unit delays: a short but acceptable freeze in video display can be noticed. In this case, the video quality is acceptable (average score: 2.9).
- More than five unit delays: a long freeze can be noticed. In this case, the video quality is unacceptable (average score: 1.2).

From the above results, we note that people can have a large range of sensitivity to the delay effect (five unit delays). In our experiment, the tolerable delay is five unit delays, which corresponds to roughly 500 msecs. In interactive audio, on the other hand, human ears can generally tolerate a delay of up to 400 msecs [8], which is almost the same as our video case.

(3) Combined effect

To consider both the packet loss and delay effects, we introduced a burst of packet loss and delay in the video stream:

- 3–7 packets loss and 1–5 unit delays: the combined effect can be noticed separately (average score: 2.5).
- 8–12 packet loss and 1–5 unit delays: the effect of packet loss is stronger to human eyes, and the effect

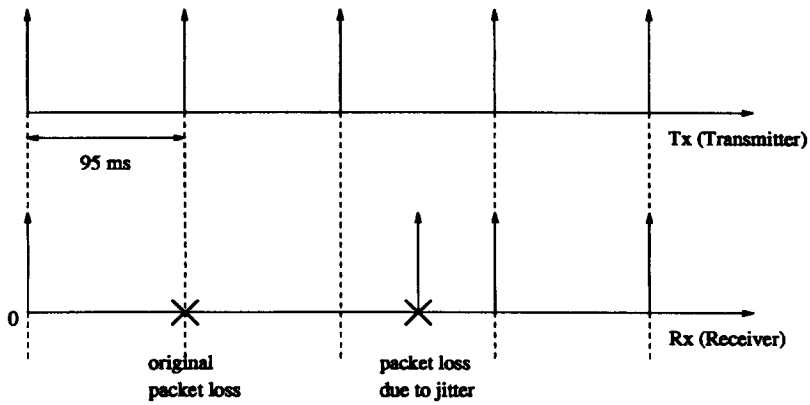
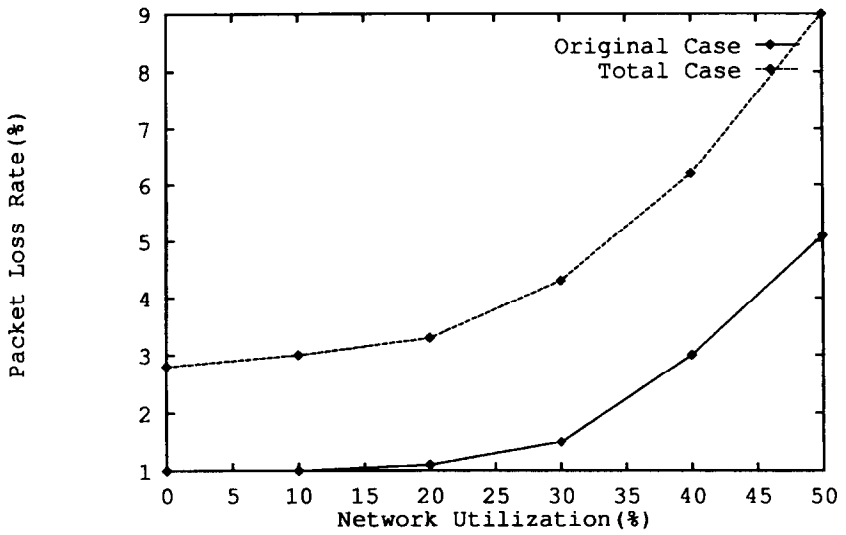


Fig. 7. Packet loss and jitter relation in the worst case.



Figs. 8. Packet loss rate vs. network utilization.

of the delay is relatively unnoticeable (average score: 2.0).

For other combinations, if one effect is much smaller than the other, the combined effect is the same as that of the stronger. On the other hand, if either effect is too strong to accept, the combined effect will be also unacceptable.

2.4.2. Loss effect on MPEG video transmission

Since MPEG has three different types of picture frames, the packet loss effect will also vary for different picture types. To determine the effect, an MPEG video experiment is also performed. In the experiment, the MPEG video has the following periodical pattern I-B-B-P-B-B-P-B-B-P-B-B-P-B-B. That is, we have 14 B-type or P-type pictures between two I-type pictures. A simple program has been written to simulate a single frame loss whose position can be specified by a user. To generate the desirable picture type loss, the user needs to know the picture type of a given position in advance.

(1) Loss of a single I-type picture

It affects the following picture sequence until the next I-type picture. Accordingly, the total number of pictures affected is 14. The resulting video quality is unacceptable.

(2) Loss of a B-type picture

Since its loss affects no other pictures, the video quality is not changed much.

(3) Loss of a P-type picture

It affects the following sequence until the next I-type picture. The number of pictures affected ranges from 2 to 11, depending on which of the four P-type pictures is affected in one sequence period. When the number of packets affected is large, the effect is actually similar to the loss of I-type pictures.

From the above observations, we have the following guidelines to implement a real-time MPEG video transmission protocol:

- Since the effect of B-type packet loss does not propagate, B-type pictures can be dropped if necessary.
- Since a single packet loss of I-type pictures is unacceptable, we need a mechanism to avoid I-type packet loss in transmission.
- Since the loss effect of P-type pictures is similar to that of I-type pictures, we also need a mechanism to protect P-type packets.

3. Video transport protocol design and implementation

From the video transmission studies described in the previous section, we find that the video quality is not strongly dependent on the video packet loss rate. Instead, it is strongly dependent on how many consecutive packets or frames are lost in the case where no compression is employed. However, when MPEG is used to compress video data before transmission, the packet loss effect can be dramatically different. For example, there can be many B (bi-directional) frames between two I (intra) frames. As a result, when an I frame is lost, many surrounding B frames cannot be recovered, causing an unacceptable loss of consecutive frames.

From this observation, to design a video transport protocol for MPEG compressed video transmission, we should ensure no I- or P-type frame loss. For minimal transmission delay, the protocol should also use a minimal buffer size at both the transmitter and receiver. In this section, we describe how these can be done on top of the built-in UDP/IP protocol suite in UNIX.

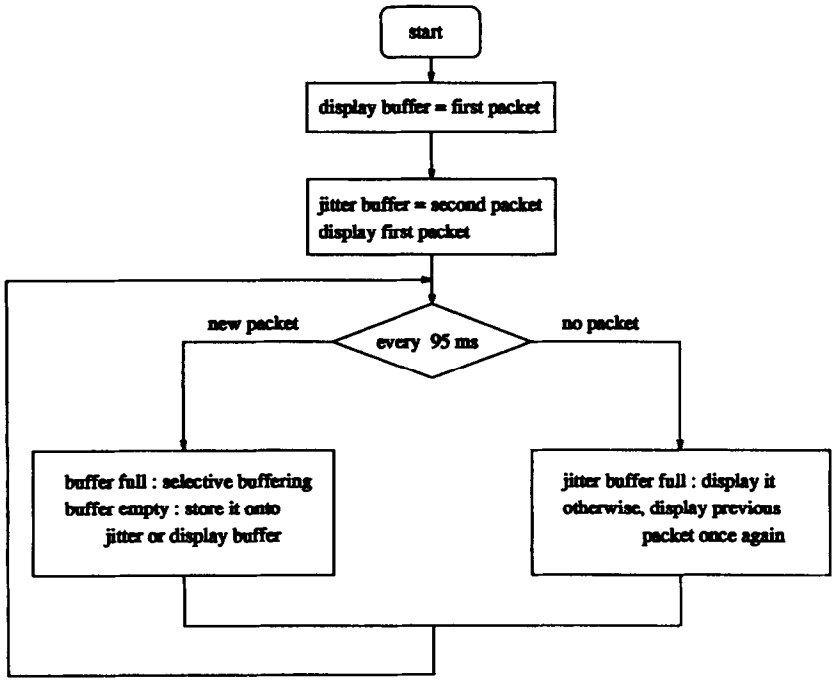


Fig. 9. Flowchart of the algorithm.

3.1. Packet loss control at the receiver side

3.1.1. Design rationale

According to the video transmission model described in Section 3, packets whose arrival jitter falls beyond the ±20 msec range will be lost. Under this jitter range, we found from experiments that there can be up to four consecutive packet losses. From this observation, we can allocate a buffer size of four video packets or frames to ensure no frame loss of I- or P-type packets. One primary disadvantage from this, however, is a long buffer delay, which is undesirable in real time communications.

Based on the following observations that we can tolerate some uncompressed video frame losses and MPEG B-type frame loss will affect no other frames, we can develop a technique called *selective buffering* to prioritize the packet loss. That is, when the receiver buffer is full and a new packet is arriving, we will store the packet that is more important (such as I- or P-type packets) and throw away the one that is less important (i.e. the B-type packet).

This selective buffering is not restricted to the Ethernet network. Because of the general nature of the technique it can be applied to different types of networks or interconnected networks to reduce the system resources, as well as to minimize the impact of packet loss in real-time video communications.

3.1.2. Algorithm of selective buffering

Below we describe the algorithm that performs selective buffering at the receiver. To implement this, we have the following two premises:

- Two frame buffers are used, one for jitter control and one for display. Each of the buffers has a flag indicating whether it is full or not.
- A separate hardware device for real-time MPEG decompression is assumed to be available. This allows the software to process only the protocol and video display.

With the above assumptions, we have the following selective buffering algorithm:

1. Initialize the receiver and wait for incoming packets.
2. Store the first arrival packet in the display buffer and set the corresponding flag.
3. When the second packet arrives, store it temporarily in the jitter buffer, and display the first packet stored in the display buffer.
4. Afterwards, display the packet in the display buffer every 95 msec, which is the same packet transmission period used at the transmitter. During this 95 msec period, one of the following two events will occur:
   (a) one or more new packets arrive,
   (b) no packet arrives.
5. If one or more new packets arrive, perform the following at each arrival:
   (a) If the jitter buffer is occupied and the display buffer is empty, pass the packet in the jitter buffer to the display buffer. This is the normal case where there is no transmission jitter and loss.

(b) If both the jitter and display buffers are empty, store the packet in the display buffer.
(c) If the jitter buffer is empty and display buffer is full, store the new packet in the jitter buffer.
(d) If both the jitter and display buffers are full, we have to drop one frame according to the following three cases:

- If the arrival packet is I-type: overwrite it onto the jitter buffer.
- If the arrival packet is B-type: drop it.
- If the arrival packet is P-type: drop it if the picture type of the stored packet is I-type; otherwise, overwrite it onto the jitter buffer.

6. If no packet arrives during the 95 msec interval, we will perform one of the following tasks:

- If the jitter buffer is occupied, the packet in the jitter buffer will be passed on to the display buffer for display.
- If the jitter buffer is empty, repeat the same frame display once.

3.1.3. Analysis of the algorithm

From the above description, we can see the selective buffering has the following advantages:

1. The selective buffering reduces the buffer size needed for acceptable video quality. This is critical in real time communications.
2. With the use of one jitter buffer and one display buffer, the maximum possible number of consecutive packet losses is reduced from 4 to 2 (under the packet loss statistics measured). As a result, the packet loss effect in un-noticeable.
3. From the use of one jitter control buffer, the packet loss rate can decrease from 4% to 1% under 50% or less network utilization (under the packet loss statistics measured).
4. The total transmission delay is reduced from four frame intervals (one interval is 95 msecs) to three frame intervals, which consists of one interval for display buffering, one interval for jitter buffering, and one interval for hardware decompression.

3.2. Packet loss control at the transmitter side

In addition to packet loss at the receiver side due to arrival jitter, we can also have packet loss control at the transmitter side during network congestion. In the case of Ethernet, when the carrier is sensed busy due to a high traffic intensity, packets stored in the transmitter buffer may not be sent to the network before the next arriving packet, which will result in a packet loss. To minimize this packet loss effect on the final video quality, we use a similar packet loss control called *selective discard*. In this approach, a transmitter buffer of one frame size is allocated. When a new compressed video frame is generated and the transmitter buffer is still busy, we will throw away the B-type packet, since its loss will not affect other picture frames.

3.3. VTP protocol implementation

From the above packet loss control, below we describe the overall VTP protocol based on the UDP/IP protocol suite [10].

(1) Packet header format

A VTP packet consists of two segments: packet header and video data. As shown in Fig. 10, the packet header consists of four bytes partitioned into three fields: packet type, block number and frame number. The one byte packet type field is used to indicate four possible packet types used in VTP, described below, and the block and frame number fields are used for frame sequence synchronization, described later.

The data segment in a packet is variable to allow for a variable video data size of up to 8192 bytes, which is the upper bound of the UDP datagram protocol in UNIX. This variable packet size feature allows us to use one packet to carry one MPEG compressed frame, which is variable in size.

(2) Packet types

There are four packet types, described as follows:

- XREQ (Transfer Request): connection setup request,
- ACCEPT: connection accept acknowledgment,
- DATA: video data,
- FINAL: connection tear-down request.

(3) Connection setup and tear-down

A video connection is set up by an exchange of two VTP packets between the sender and receiver. In implementation, the sender behaves as an active entity (client) and the receiver behaves as a passive entity (server). To start, the active end first sends an 'XREQ' packet. After arriving at the destination, the passive end responds with an 'ACCEPT' packet. Once the active end receives the acknowledgement, the transfer can begin. After data transmission is finished, the connection can be torn down by sending a 'FINAL' packet from the active end to the passive end.

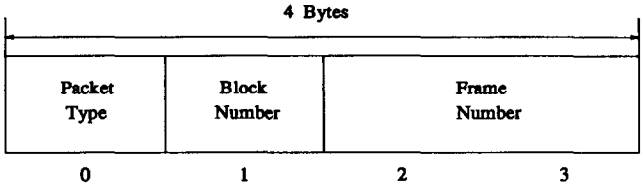


Fig. 10. VTP packet header format.

(4) Flow control
There is no flow control employed in the VTP protocol. Instead, from the video transmission model the transmission rate is fixed at 95 msecs per frame, which is designed to handle the jitter range of ±20 msecs and to achieve at least a 10 framing rate. Since the average compressed frame size is 3.5 Kbytes, the average bandwidth required comes to only 300 kb/s, which is small compared to the 10 Mb/s of Ethernet. This allows us to have multiple video transmission sessions.

(5) Error detection and control
Error detection allows the receiver to detect the corrupted packets and drop them. Since VTP is based on the UDP/IP suite, error detection is simply done by the UDP checksum mechanism, which covers both the VTP packet header and data. Since we are concerned with real-time video transmission, there will be no retransmission for error detected packets.

(6) Sequence number
The frame number and the block number fields in the VTP packet header together form a VTP sequence number. The frame number is used to represent one video frame in the sequence. Since it has a size of two bytes, it permits a cyclic sequence up to 64 K frames. When a compressed video frame exceeds the maximum UDP packet size of 8192, multiple packets will be used to carry the same video frame. In this case, all the packets have the same frame number, but their block numbers are sequential from 0 to $M-1$, with M being the total number of packets used to transmit the video frame.

The VTP sequence number can be used to detect frame loss, out of sequence, and duplication. Therefore, it is important to reproduce a correct sequence video at the receiver.

(7) Timing issue
There is an additional implementation issue of the proposed VTP protocol that should be addressed. The issue is that the frame interval of 95 msecs used at the transmitter and receiver is not exactly the same due to different system clocks. When the small timing difference is accumulated over a long interval, we can have additional packet loss [11]. To solve this, we can use the estimated transmitter frame interval by counting the number of received packets over a long interval. If the number of packets actually received is larger than the expected number, we need to decrease the interval. On the other hand, if the actual number is smaller, we need to increase the interval. When there is a packet loss, we can use the sequence number to make necessary counting adjustments.

4. Test results

In this section, we describe the test setup, procedures, and results based on the VTP protocol described in the previous section. The performance is found to be satisfactory using the proposed schemes.

4.1. Test environment

A test setup is shown in Fig. 11 to evaluate the proposed mechanisms discussed in the previous section for real-time video communications. In the test, two Sun/Sparc workstations with the SunOS release 4.1.3c operating system are used. Since the hardware devices for real-time video encoding and decoding are not available, a real-time emulation algorithm is devised.

As shown in Fig. 11, the video source on the sender side comes from a pre-compressed MPEG file instead of a video camera and MPEG encoder. To improve the speed and avoid the file read/write delay, the whole file is first loaded into real memory before the transmission starts. On the receiver side, all arrival timings of received packets are first recorded. After non-real time software decoding, we can play back the video according to the arrival timings.

Below we describe some important routines used to facilitate the tests:

- *Typeck*: this is a routine to identify the MPEG frame boundary and the picture type by matching the picture start code and checking the picture type field.
- *Lossck*: this is a routine to calculate what frames are lost, and what their types are, by comparing the received frame sequences with the original frame sequences.
- MPEG decompression: this is a software version of MPEG decoding modified from the UC Berkeley version [6].
- X-video player: also modified from the UC Berkeley MPEG program, this routine plays back the decompressed MPEG file in an X window. While playing, the log file is read to emulate the real time transmission according the recorded arrival timings.

4.2. Test procedures

Below we describe the test procedures in detail.

A. Sender side

1. An MPEG encoded file is first loaded into the real memory of the sender workstation.
2. The type of each MPEG video frame is identified before packetization.
3. Each video frame is stored in the VTP frame buffer according to the selective discard algorithm described earlier. To further protect I-type frames, they are transmitted in duplication.

B. Receiver side

1. Whenever a VTP layer packet is received, its data and arrival time are stored in a data file and a log file, respectively.

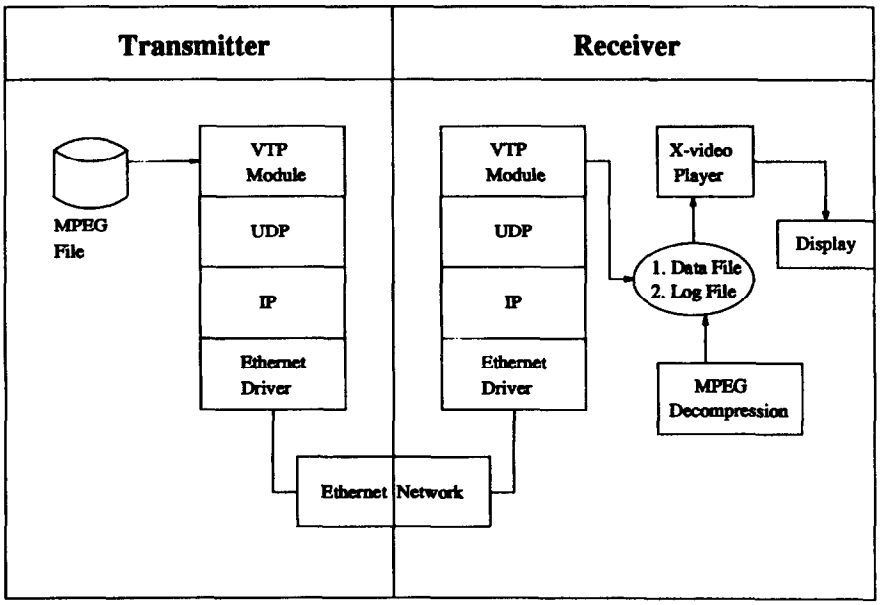


Fig. 11. Test environment.

2. When the transmission is done, the compressed MPEG data are decoded.
3. To provide the transmission statistics such as the packet loss rate and delay jitter, a supplementary tool has been written.
4. The decompressed MPEG file can be played back according to the arrival timings, where the jitter reduction scheme and the selective buffering algorithm described in the previous section are also taken into consideration.

4.3. Test results

To see the improvement from using the selective discard and buffering algorithms proposed, we performed the following combinations in the tests.

A. Sender side

- Using the selective discard scheme.
- Without using the selective discard scheme.

B. Receiver side

- Using the jitter reduction and selective buffering scheme.
- Without using the jitter reduction and selective buffering scheme.
- Playing back according to the arrival time.
- Playing back one video frame every 95 msec.

From the above possibilities, we can have six possible combinations as follows:

1. No selective discard and no selective buffering.
2. No selective discard but with selective buffering.
3. No selective buffering but with selective discard.
4. With both selective buffering and selective discard.
5. With selective discard and display at every 95 msec (neglect the actual arrival timing).
6. Without selective discard and display at every 95 msec (neglect the actual arrival timing).

In the last two cases, since the actual arrival timing is neglected, there is no packet loss due to arrival jitter. Therefore, selective buffering is not needed.

From the test combinations described, we have the following test results:

- No selective discard and no selective buffering. In this case, there is a large probability that I- or P-type pictures can be lost from either transmission or arrival jitter. Therefore, the playback video is messed up.
- No selective discard but with selective buffering. In this case, there is still a large probability of losing I- or P-type packets from transmission. Therefore, there is no significant performance difference compared to the first case.
- No selective buffering but with selective discard. In this case, it is rare to lose I- or P-type pictures from transmission. However, there is a large possibility of losing them in playback due to arrival jitter. Therefore, we have almost the same results as above.
- With both selective buffering and selective discard. In this final case, I- and P-type packets are well protected and not lost. As a result, the subjective performance from experiments is found to be very satisfactory.

4.4. Discussion over internetworking

Since most video communications are not within a single Ethernet, it is important to know what happens

when the same VTP is used and what modifications are necessary for satisfactory video transmission. Below we consider the application of the VTP protocol over two long-haul networks: the Internet and the B-ISDN/ATM (Broadband Integrated Service Digital Network/Asynchronous Transfer Mode).

When the protocol is used over the Internet, we will have a higher packet loss rate for three primary reasons. First, since there are several intermediate routers and bridges in the network, we have a higher probability of having buffer overflow, which results in a higher packet loss rate. Second, because of the more random traffic statistics, there is a larger jitter variation, which can cause a larger packet loss rate if we maintain the same jitter window of ± 20 msecs. Finally, since the Internet is based on the unreliable datagram protocol (IP), packets are likely to arrive out of sequence. As a result, more packets can get lost on re-assembly at the destination due to buffer overflow or being late for display.

Because of the higher packet loss rate and no internal selective control, we can lose I- or P-type packets, which can cause significant video quality degradation, as explained earlier in the paper. Furthermore, when the packet loss rate is high, adjacent bursts of packet losses can be close and affect each other. This is a scenario not considered earlier, and its combined effect on the video transmission is expected to be much more serious.

To reduce the packet loss effect over the Internet, and at the same time maintain the same framing rate or the same 95 msec framing interval, a simple solution is to increase the buffer size for a larger jitter window. If one additional buffer is used, we can have an extra 95 msecs for the jitter window. As a result, we can increase the jitter window to ± 67.5 msecs, which can reduce the packet loss rate significantly due to packet arrival jitter. This approach exchanges packet loss for a longer transmission delay. When a short delay transmission is not an important criterion, this approach is an effective and simple solution.

Since the internal Internet network has no selective loss control as proposed in the VTP, another simple solution to reduce the packet loss effect on video transmission is to duplicate the transmission of important packets such as I- or P-type frames. In general, one critical disadvantage of this approach is a higher average transmission bit rate. When this is a concern, we have to use a different code instead of MPEG for more robust and less redundant transmission.

When transmission over a B-ISDN/ATM is considered, we can also expect a higher packet loss rate. However, due to a better packet loss control and larger transmission bandwidth, the increase of the packet loss is primarily due to a larger jitter distribution at the destination. In contrast to the Internet, congestion and flow control mechanisms used in ATM can significantly reduce the packet loss rate and jitter distribution. Therefore, the simple modifications of the VTP protocol suggested above can be promising for future ATM internetworking.

5. Future work

Some possible extensions of this work are described as follows. First, it would be obvious for the next step to conduct similar experiments over ATM networks so that we develop a video transmission model for demanding video applications with higher quality, by quantifying the jitter and packet loss statistics. From this study, we can identify some modifications of the VTP protocol required for an ATM environment.

Second, since we cannot access the UDP/IP layer in the current implementation, we cannot access the UDP/IP buffer. Therefore, we cannot implement the selective buffering and discard schemes at the UDP/IP layers. This means we can still have a chance of losing I- or P-type pictures, which will cause a serious damage on the video playback. However, if we can implement selective buffering in the driver of the receiver, we can avoid losing I- or P-type pictures when the receiver buffer gets overflow.

Finally, we have only tested the proposed schemes over a single Ethernet. It will be useful to perform transmission tests through a number of Ethernet routers or gateways. Also, we will need to incorporate a separate hardware encoder and decoder to make sure that the current emulation might neglect some problems at the interfaces between the hardware and software modules.

Acknowledgement

I would like to give my particular thanks to my advisor, Dr. Ian F. Akyildiz for his insightful comments and constructive suggestions.

References

[1] M.K. Liu, Circuit-emulated video communications for telecommunications research and curriculum enhancement, Preliminary Report to Advanced Telecommunications Research Project, 1990.
[2] A.H. Pathan, Ethernet interface for real time video communications, MS Thesis, Electrical and Computer Engineering Department, University of Arizona, 1993.
[3] D. LeGall, MPEG: A video compression standard for multimedia applications, Comm. ACM, 34(4) (April 1991) 46–58.
[4] M. Aras et al., Real-time communication in packet-switched networks, Proc. IEEE, 82(1) (January 1994) 122–138.
[5] A.N. Netravali et al., Design and implementation of a high-speed transport protocol, IEEE Trans. Comm., 38(11) (November 1990) 2010–2024.
[6] L.A. Rowe et al., MPEG video software decoder, Berkeley Plateau Research Group, EECS, University of California at Berkeley, January 1993.
[7] Z. Wang and J. Crowcroft, Analysis of burstiness and jitter in

real-time communications, Proc. ACM SIGCOMM, September 1993, pp. 13–19.
[8] C. Partridge, Gigabit Networking, Addison-Wesley, 1994.
[9] S. Lei, Forward error correction codes for MPEG2 over ATM, IEEE Trans. Circuits and Systems for Video Technology, 4(2) (April 1994) 200–203.
[10] W.A. Doeringer et al., A survey of light-weight transport protocols for high-speed networks, IEEE Trans. Comm., 38(11) (Novemebr 1990) 2025–2039.
[11] R.P. Singh et al., Jitter and clock recovery for periodic traffic in broadband packet networks, IEEE Trans. Comm., 42(5) (May 1994) 2189–2196.