Retransmission-Based Error Control for Interactive Video Applications over the Internet

Injong Rhee
Department of Computer Science
North Carolina State University
Raleigh, NC 27695-7534
E-mail: rhee@csc.ncsu.edu

Abstract

Retransmission has been known ineffective for interactive video transmission over the Internet. This paper challenges this view by presenting several retransmission-based error control schemes that can be used for interactive video applications. In particular, the schemes do not require any artificial extension of control time and play-out delays, and thus are suitable for interactive applications. They take advantage of the motion prediction loop employed in most motion compensation-based codecs. By correcting errors in a reference frame caused by earlier packet loss, the schemes prevent error propagation. Since a reference frame is arranged to be referenced for the construction of the current image much later than the display time of the reference frame, the delay in repairing lost packets can be effectively masked out. Internet video transmission experiments reveal the superior error resilience of the schemes.

1 Introduction

Video conferencing over the Internet has become increasingly popular because of the widespread use of the Internet and video compression technologies. However, high quality interactive video transmission over the Internet still remains challenging because of frequent occurrences of packet loss and limited bandwidth in the network. The problem mainly arises from the disparity of the operational models of traditional video coding standards and Internet-based video conferencing. Most of standard coding schemes, such as H261, H263, and MPEG, are not designed for transmission over a lossy packet switching network, but primarily for storage (CD or VHS tape). Although these schemes can achieve very high compression efficiency, even small packet loss could severely degrade video quality. This is due to motion compensation employed by these coding schemes to remove temporal redundancy in video stream. Motion compensation removes temporal redundancy in successive video frames (inter frame) by encoding only pixel value differences (prediction error) between a currently encoded image and a previously encoded image (reference frame). A single occurrence of packet loss can introduce an error in a reference frame, which can be propagated to its succeeding frames and gets amplified as more packets are lost.

Error propagation can be controlled by more frequently adding intra frames (which are coded temporally independently). However, the ratio of the compression efficiency of an intra-frame over an inter-frame is as large as 3 to 6 times. Increasing the frequency of intra-frames could increase the bandwidth requirement too much for video transmission over a bandwidth-constraint network. Nonetheless, the severe degradation of image quality due to error propagation has forced several popular video conferencing tools, such as `nv`[5], `vic`[7] and `CU-SeeMe`[4], to adopt an even more drastic approach. Using a technique called *conditional replenishment*, these tools filter out the blocks that have not changed much from the previous frame and intra-code the remaining blocks. Since all the coded blocks are temporally independent, packet loss affects only those frames contained in lost packets. However, this enhanced error resilience comes at the cost of low compression efficiency. Additional compression can always be obtained if temporal redundancy is removed from each coded block (i.e., by coding only their prediction error).

The goal is to find an error recovery scheme that solves the error propagation problem without much increase in the bandwidth requirement.

Retransmission-based error recovery (REC) can provide good error resilience without incurring much bandwidth overhead because packets are retransmitted only when some indications exist that they are lost. However, retransmission has been widely known ineffective for interactive video transmission because of the delay associated with detecting and recovering lost packets. Many researchers proposed to extend control or play-out times in order to allow enough time for retransmitted packets to arrive before their display times [11, 10, 6, 12]. This implies that the display time of a frame is delayed by at least three one-way trip times after its initial transmission (two for frame transmissions and one for a retransmission request). Under the current Internet environment, this delay can be as large as 400 ms to 600 ms. For instance, in a transatlantic transmission experiment, one round trip time delay is usually between 200 and 300 ms. When the network connection gets congested, the delay frequently rises beyond 400 ms. This latency significantly impairs the interact-ability of any real-time video applications.

In this paper, we present new REC schemes that do not require any additional control or play-out delay, and hence are suitable for real-time interactive applications. In addition, the proposed schemes do not require much change in existing standard codecs. We performed extensive transatlantic video transmission tests over the Internet to measure the effectiveness of the schemes. Our experiments indicate that REC can be a very effective error recovery technique for interactive video applications.

Section 2 describes the related work, Section 3 presents our REC schemes, Section 4 contains a discussion of the experimental results, and Section 5 contains the conclusion.

2 Related Work

Dempsey et al.[3] applied retransmission for the recovery of audio packets. They showed that by adding some delay before the play-out of each received audio packet, retransmission can be used to protect audio data from packet loss. Their work hinges on the earlier behavior study by Brady [2] showing that although less than 200 ms round trip delay is required for high quality voice applications, delays up to 600 ms can be tolerable by human ears.

Ramamurthy and Raychaudhuri [11] applied a similar technique to video transmission over ATM. They analyzed the performance of video transmission over an ATM network when both retransmission and error concealment are used to repair errors occurring from cell loss. They analytically showed that for a coast-to-coast ATM connection, 33 ms to 66 ms play-out delay is sufficient to see a significant improvement in image quality.

Padopoulos and Parulkar [10] proposed an implementation of an ARQ scheme for continuous media transmission. Various techniques including selective repeat, retransmission expiration, and conditional retransmission are implemented inside a kernel. Their experiment over an ATM connection showed the effectiveness of their scheme.

Most recently, Li et al. [6] and Xu et al. [12] used retransmission in the recovery of lost packets for video multicasting. Li et al [6] proposed a novel scheme for distributing an MPEG-coded video over a best-effort network. By transmitting different frame types (I, P and B frames) of MPEG to different multicast groups, they implemented a simple layering mechanism in which a receiver can adjust frame play-out times during congestion by joining or leaving a multicast group. For instance, consider an MPEG picture pattern: IBBPBBPBBPBB. Their scheme delays the play-out times of frames for one frame interval. They call this delayed play-out time *adaptive playback point.* When a receiver leaves the B frame group, the adaptive playback point is additionally extended by three frame intervals because the next frame to be displayed after a P frame is at three frame intervals away. The scheme is shown effective for non-interactive real-time video applications.

In a video conference involving a large number of participants, different participants may have different service requirements. While some participants may require real-time interactions with other participants, others may simply want to watch or record the conference. Xu et al. [12] contend that retransmission can be effectively used for the transmission of high quality video to the receivers that do not need a real-time transfer of video data. They designed a new protocol called *structure-oriented resilient multicast* (STORM) in which senders and receivers collaborate to recover lost packets using a dynamic hierarchical tree structure.

3 Retransmission-Based Error Control (REC)

Our schemes are based on a careful observation on how video frames are encoded in most motion compensation-based codecs. We base our discussion mostly on H.261 from which many motion compensation-based video standards such as MPEG are designed. In H.261, a video sequence consists of two types of video frames: *intra-frame* (I-frame) and *inter-frame* (P-frame). I-frame removes only spatial redundancy present in the frame. P-frame is encoded through motion estimation using another P-frame or I-frame as a reference frame (R-frame). For each image block in a P-frame, motion estimation finds a closely matching block within its R-frame, and generates the distance between the two matching blocks as a motion vector. The pixel value differences between the original P-frame and a motion-predicted image of the P-frame obtained by simply cut-and-pasting the matching image blocks from its R-frame are encoded along with the motion vectors.

Most of the previously proposed retransmission schemes work as follows. When a packet containing the encoding of a frame is lost at a receiver, the receiver detects the loss after receiving a subsequent packet of the lost packet and sends a retransmission request to the sender. After receiving the request, the sender retransmits the packet. We define the *display time* of a packet to be the time that the frame whose encoding is contained in the packet is displayed at the receiver. If the retransmitted packet arrives before its display time, the frame can be fully restored. Otherwise, it is discarded and the displayed image contains some error. All the subsequently decoded frames will carry the same error unless a new I-frame is received.

Our scheme differs from others in that retransmitted packets arriving after their display times are not discarded but instead used to reduce error propagation. In motion compensation-based codecs, the correct image reconstruction of a currently displayed image depends on a successful reconstruction of its R-frames. The scheme allows that while a frames is being reconstructed, the “late” packets of an R-frame can be decoded and used for restoring the R-frame. This will stop possible error propagation because the next frame reconstructed would not carry over an error from the R-frame.

Figure 1 shows a H.261 decoder modified to handle the recovery of R-frames through retransmitted packets. The only difference from the original H.261 decoder is one additional frame buffer added to handle the recovery. When a packet is received and decoded into an image block, the decoder determines whether the block belongs to the current frame being decoded or its R-frame. If it is for the current frame, then the block is stored into frame buffer CP along with its motion vector. If it is for the R-frame, the block is added with its temporally dependent block in frame buffer R_0 and stored into R_1. Note that CP contains only the prediction error and motion vectors of the current frame while R_1 contains the fully motion compensated image of the R-frame of the current frame, and R_0 contains the R-frame of R_1. We call R_0 a *base reference frame buffer.* At the next display time, ' the current frame is constructed using the information in CP and R_1. After the display, R_1 is copied to R_0 and the displayed image are copied to R_1. In this scheme, as long as the packets belonging to R_1 arrive before the construction of the current frame, the packet can be used to help remove errors in the current frame. The *deadline of a packet* can be informally defined to be the arrival time of the packet at the receiver after which it is not useful for decoding any frame. Note that the decoder in Figure 1 extends the deadline of packets by one frame interval without delaying frame play-out times.

We can easily generalize the above scheme to extend packet deadlines beyond one frame interval. Below, we discuss three such schemes. Many different variations of the schemes are also possible.

In H.261, each R-frame temporally depends on the previous R-frames. Thus, by employing more R-frame buffers, the late packets can be decoded and used to restore their corresponding frames which are used as R-frames for the subsequent frames. Since a frame temporally depends on all the prior frames encoded after its immediately preceding I-frame, restoring a sequence of R-frames preceding to a frame contributes to reducing error propagation. Figure 2 illustrates this scheme, called *cascaded buffering.* The shaded squares represent image blocks and an arrow represents the temporal dependency between two blocks. Suppose that the current frame number is f. R_0 is the base reference frame buffer and contains the completely reconstructed image of frame $f-4$ while R_i $(1 \leq i \leq 4)$ contains the prediction error and motion vectors of frame $f-4+i$. The image block that corresponds to b_3 can be constructed by adding b_0, b_1, b_2, and b_3. However this scheme has two drawbacks. First it requires many frame buffers if the

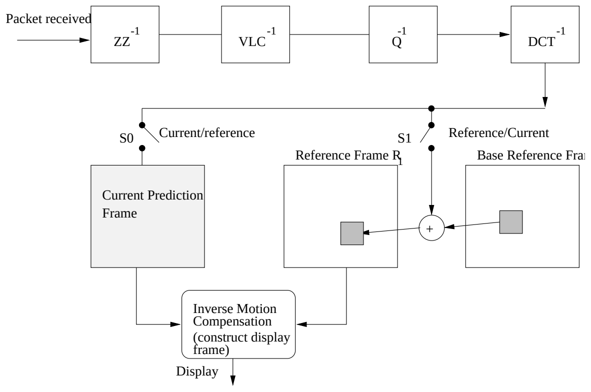

Figure 1: H.261 Decoder modified to handle the recovery of reference frames.

message delay is too long. Second, computational overhead might be large because all the R-frames in the decoder are needed to be added to construct the currently displayed frame.

Another way to extend the deadline is to increase the *temporal dependency distance* (TDD) of a frame which is defined to be the minimum number of frame intervals between that frame and its temporally dependent frame. Figure 3 shows an example where every frame has TDD 3. Each frame temporally depends on another frame sampled at three frame intervals away. The extended TDD essentially stretches the deadlines of packets because each displayed frame is referenced for the reconstruction of another frame only after one TDD period. We call this scheme *extended TDD* (ETDD). Because each frame within a TDD period depends on a frame in the previous TDD period, the receiver has to maintain all the reference frame buffers within a TDD period for the decoding of the frames in the next TDD period. In addition, to restore each R-frame through retransmission, each reference frame R should have one base reference frame. This scheme has another drawback. Since the prediction error of each frame is obtained from the frame that is at a few frame intervals away, it may reduce compression efficiency. However, the ETDD scheme does not require as much computational overhead as the cascade buffering scheme because the current decoded frame can be constructed from its R-frame and base reference frame.

Figure 4 shows another scheme, called *periodic TDD* (PTDD). In the scheme, only every ith frame has TDD i (we call this frame *a periodic frame*) while all the other inter-frames have TDD 1. This frame pattern is very similar to the picture group pattern of MPEG. All the periodic frames can be regarded as P-frames while the other frames as B-frame (except the first frame). Thus, this scheme can be easily incorporated into MPEG. PTDD is both computationally and spatially efficient since only periodic frames require two reference frame buffers. It would also give better compression efficiency than ETDD because every non-periodic frame has TDD 1. However, PTDD does not provide any protection for non-periodic frames from packet loss. An error in a non-periodic frame can propagate until the next periodic frame is received.

In all of the three schemes mentioned above, the packet deadline can be dynamically extended either by adding more frame buffers or extending the temporal dependency distance. For these schemes to be effective, an appropriate packet deadline needs to be selected that allows high compression efficiency as well as sufficient time for a large portion of retransmission packets to arrive before their deadlines. Finding the optimal packet deadline under a given network condition is left as future work.

4 Experimental Result

The main objective of the experiment is to show that a REC scheme is an effective error control scheme for a real-time interactive video transmission over the Internet. We show this through an Internet video transmission experiment between the University of Warwick, UK and Emory University, GA, USA. Among the three REC schemes mentioned in Section 3, we implement the periodic temporal dependency distance (PTDD) scheme by modifying a H.261 coder. The modified coder is called HP.261. The experiments with the other

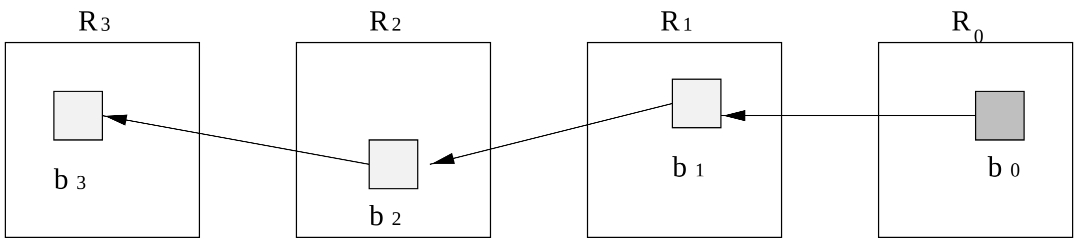

Figure 2: Cascaded Buffering

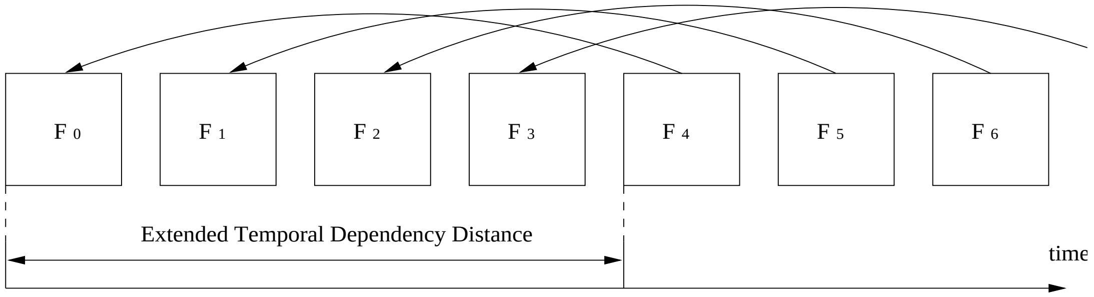

Figure 3: Extended Temporal Dependency Distance

REC schemes presented in this paper are left for future work.

The performance of HP.261 is compared to that of two other codecs. H.261 is used as a base case for the comparison. We implemented INTRA-H261 which is used in `vic`. INTRA-H261 is known for good error resilience under packet loss. INTRA-H.261 intra-codes every image block changed significantly much from the corresponding block in the previous frame. Section 4.1 describes the testing methodology and environment Section 4.2 presents the experimental result.

4.1 Testing Methodology

A test video sequence is obtained from a typical video conferencing session where one typical "talking head" engages in a conversation with the other party. The video is sampled at 5 frames/sec rate and each frame is captured in the color CIF YUV format (352×288). This video sampling rate allows us to achieve a bit rate suitable for a transatlantic transmission without imposing too much load on the network. Considering the long distance and limited bandwidth between the testing sites, this frame rate is not unusual. The target bit rate is around 250 Kbits/sec. In addition to the controlled sample rate, we use a conditional replenishment technique for all the tested schemes to obtain a desired bit rate. Adjusting quantization steps would be a more common way to control the bit rate. However, since INTRA-H261 uses conditional replenishment, we use the same technique uniformly for all the schemes for fairness. Finding the optimal video quality for a given bit rate is outside the scope of this paper.

About 40 second length video sequence (total 190 frames) is obtained. The video sequence is replayed several times for five minute period for each experiment. The replay does not affect the integrity of the experiment because the first frame is always completely intra-coded (without any conditional replenishment). The 95th frame is intra-coded with conditional replenishment to remove any artifact due to the *decoder drift* effect. For all the schemes, we applied a default quantization step size 8, and for all the motion-compensated schemes, a full exhaustive search over search window size 15 by 15 is performed. We chose 5 frame intervals to be the TDD of all the periodic frames in HP.261. Given 5 frames/s frame rate, this TDD extends the deadlines of periodic frames up to 1 sec.

To see the compression efficiency of different schemes. we measure the average peak signal-to-noise ratio (PSNR) of decoded frames over various data rates. The data rate is measured by the average number of bytes required to compress a frame which is plotted in Figure 5. For a given data rate, INTRA-H.261 shows the worst video quality while H.261 shows the best. For instance, to obtain

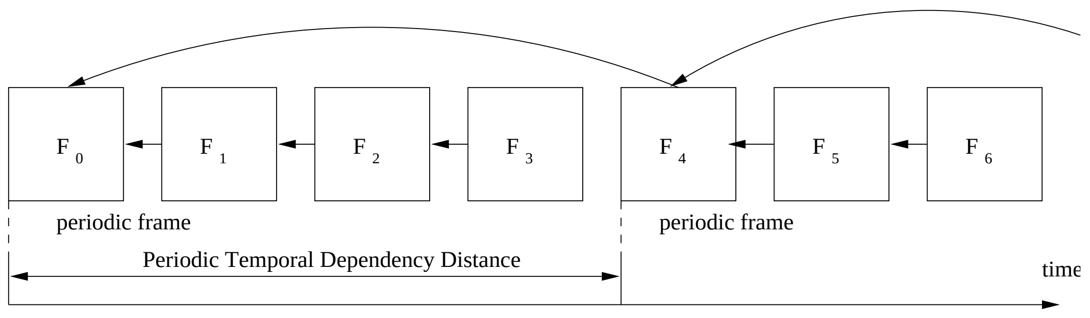

Figure 4: Periodic Temporal Dependency Distance

about 34 dB PSNR, INTRA-H.261 requires 80% (11KB/6KB) more bits per frame than the others.

For fairness, each scheme should use a similar bandwidth for transmission. Since H.261 gives the best compression efficiency, we chose the bandwidth requirement of H.261 under the given compression parameters (e.g., search window and quantization steps). H.261 gives the maximum PSNR around 240 Kbits/s.

We set the bit rates of other schemes to this bit rate. Figure 6 shows the chosen data rates. INTRA-H.261 is given a higher data rate because we could not get a finer precision on its data rate by varying only the conditional replenishment threshold. However this higher bit rate does not unfavorably affect the INTRA-H.261 because INTRA-H.261 is not actually transmitted. Instead, only the video sequence of HP.261 is transmitted and the other sequences are simply mapped to the obtained transmission traces of HP.261. More details about this mapping are given in the next section. HP.261 is given a bit lower data rates because the associated retransmission of lost packets would increase the actual data rate. We anticipate the actual data rate during a transmission test would be similar to the others. Figure 7 shows the PSNR of each frame compressed by three different schemes under the data rates given in Figure 6.

The test video sequence is first compressed using each of the three compression schemes and then packetized into approximately 512-byte packets. The packetized sequence of HP.261 is transmitted over the Internet. For each transmission test, we obtain a 5 minute trace that records the packet sequence numbers and arrival times of all the received packets. The transmission tests are conducted at every hour for 8 days between Oct. 20 to Oct. 27.

Each packet of a frame is transmitted at a regular interval determined by the given frame rate (5 frames/s) and the number of packets within the frame. For example, for the frame interval of 200 ms, if one frame consists of 10 packets, a packet in the frame is transmitted at 20 ms interval. Each transmitted packet is assigned a unique sequence number. Retransmitted packets are given the same sequence numbers as their original packets.

The ARQ scheme employed during the experiment works as follows. The receiver sends one acknowledgment to the sender for each received frame. An acknowledgment contains information about the missing packets of the last periodic frame that the receiver received. After retransmitting a packet, the sender does not retransmit the same packet for about three frame intervals. It may retransmit the packet if it receives another acknowledgment after the period indicating that the packet is lost. The receiver also does not request for the retransmission of packets whose deadlines are expired. These mechanisms reduce the number of unnecessary retransmissions.

Each trace is fed to an off-line decoder to measure the signal-to-noise of the received frames. To simplify the experiment, we did not add any jitter control time for frame play-out. Each frame is considered to be displayed at the arrival of the first packet of its next frame if that packet is received. If that packet is not received, the frame is considered to be displayed at 200 ms after its previous frame's play-out time. If no packet for the frame is received, any frame displayed last will be displayed for that frame. Retransmitted packets are not used for the display of their frames, but used only to restore their corresponding reference frames.

For a comparison purpose, we map each of the obtained traces T to the packetized sequence of H.261 and INTRA-H.261 as follows. We first obtain a 5 minute length of the packetized sequences

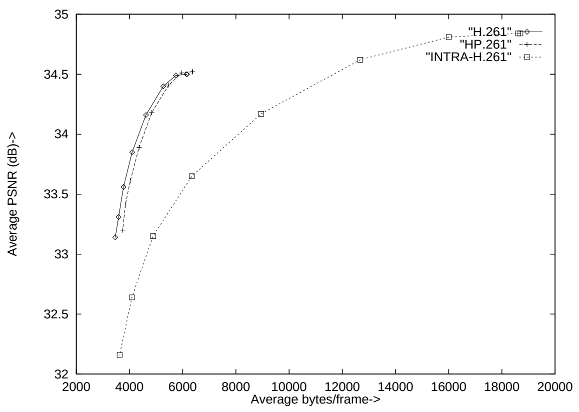

Figure 5: Compression Efficiency of Various Schemes

Compression scheme	Avg. bit rate Avg. Kbits/s	PSNR
H.261	240.6	34.50
HP.261	232.6	34.51
INTRA-H.261	247.77	33.65

Figure 6: Chosen data rates for network experiments, and their average PSNR's

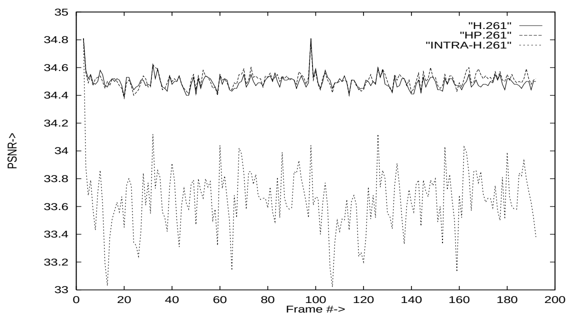

Figure 7: Video quality of encoded sequences

S of H.261 and INTRA-H.261 as if the sequences would have actually been transmitted. Each packet p in trace T is mapped to a packet q with the same sequence number as p. If packet p is received, we record q as received and assign the receiving time of p to q. Otherwise, we record that q is lost.

We obtained 168 traces of HP.261. Since many traces are obtained, it is difficult to present the result of each trace independently. So we classify the traces into several loss rate groups and present only the average behavior of the traces in each group. Table 1 shows the loss groups and their corresponding loss ranges. Since high loss cases are relatively infrequent, we set a larger range for high loss rates.

4.2 Performance of REC

In this section, we report the result of our experiment. Figure 8 shows the average PSNR's of H.261, INTRA-H.261 and HP.261 for various loss groups. H.261's PSNR drop considerably even under a small amount of packet loss showing the severe impact of error propagation. Both INTRA-H.261 and HP.261 exhibit generally good error resilience. Both show a similar PSNR for all loss groups. Under less than 10 % loss groups, HP.261 shows better PSNR than INTRA-H.261. Between 12% and 20% packet loss, the PSNR of HP.261 drops a little below INTRA-H.261. This is due to the drop in the REC recovery rates (see Figure 9).

Figure 10 shows a portion of one HP.261 trace with 10% packet loss. The figure on the top compares PSNR's of H.261 and HP.261 of all the frames received during the period. The figure on the bottom shows two kinds of data. The impulses indicate the ratio of the number of lost packets over the number of packets in a frame, and the line points indicate the ratio of the number of packets recovered by REC over the number of packets in a frame. When a point reaches the top of an impulse, it means all the lost packets in the frame are recovered. The line points are relatively sparse because only the packets belonging to periodic frames are retransmitted and no points are drawn for the frames that did not lose any packet.

Many packets are lost between sequence numbers 300 and 550. Accordingly the PSNR's of both HP.261 and H.261 drop significantly. How-

Loss group	0.025	0.05	0.075	0.1	0.125	0.15
Loss range	(0, 0.025)	[0.025,0.05)	[0.05,0.075)	[0.075,0.1)	[0.1, 0.125)	[0.125, 0.15)
Loss group	0.175	0.2	0.25	0.3	0.35	0.4
Loss range	[0.15, 0.175)	[0.175,0.2)	[0.2, 0.25)	[0.25,0.30)	[0.3, 0.35)	[0.35,0.40)

Table 1: Loss Rate Groups and their loss ranges

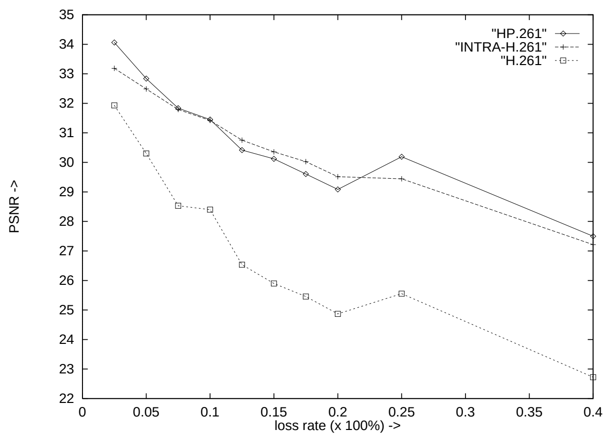

Figure 8: Mean PSNRs

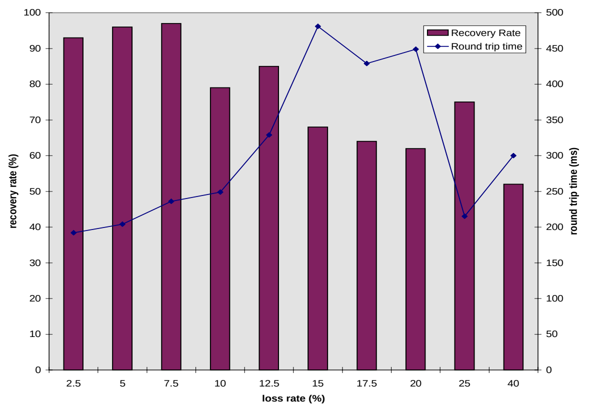

Figure 9: Mean round trip time, and recovery rate

ever, in HP.261, all the packets of periodic frames within that period are recovered by retransmission. Around packet 600, most frames are received without loss. In HP.261, those frames are displayed without an error since all the periodic frames that they temporally depend on are recovered fully before their reconstruction times. On the other hands, H.261 suffers severely from error propagation. The good receiving rate around packet 600 does not improve the PSNR of H.261 during that period.

If a periodic frame contains some error, all the frames before the next periodic frame will contain the same error. However, if the periodic frame is recovered later on before the next periodic frame is reconstructed, the error will disappear from the subsequent frames of the next periodic frame. The many peaks of PSNR shown in the top of Figure 10 illustrate this behavior.

There is a clear correlation between the round trip time and REC recovery rates. As the round trip times increase, the recovery rates of periodic frames also drop. When round trip times increase beyond 250ms, the recovery rates by REC are significantly reduced. This is because the increased network delay reduces the probability for retransmitted packets to meet their deadlines. However, the packet loss rates do not necessarily correlate with the round trip times. The trace in 25% has the mean round trip times less than 250 ms, which contributes to the increases of the recovery rate up to 75% and PSNR to 30 dB. The trace contains a bulk loss in the middle of the experiment during which no packets are received.

A large amount of packet loss causes many retransmissions increasing the data rates. In Table 2, the data rates for HP.261 between 12.5% and 20% go beyond 260 Kbits/s, which means on an average about 5 to 6 packets/s are retransmitted. These data rates are slightly higher than those of INTRA-H.261 under some loss groups. We believe, however, that this is mainly due to the crudeness of the retransmission decision protocol (i.e., when to retransmit) we employed in the experiment. Use of a more sophisticated ARQ method such as the one used in [10] should reduce the data rates further.

5 Discussion

Retransmission has widely been known ineffective for recovering lost packets in real-time interactive video applications. This paper challenges the conventional wisdom and presents new retransmission-based error control techniques that do not incur any

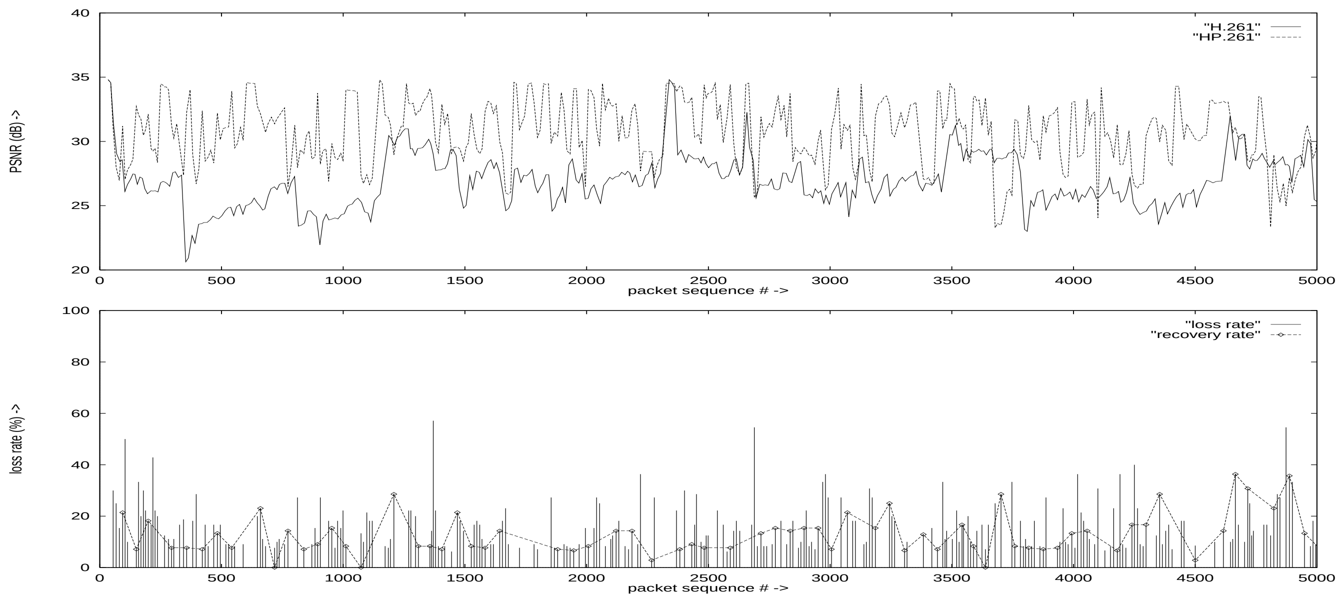

Figure 10: One trace of HP.261 with 10% packet loss – PSNR, and Loss and Recovery Rates

Loss Rate (%)	# of traces	H.261	HP.261			INTRA-H.261
		Data Rate (Kbits/s)	Data Rate (Kbits/s)	Recovery by REC(%)	RTT (ms)	Data Rate (Kbits/s)
2.5	54	237.2	246.4	93.39	192.17	258.1
5.0	40	245.0	247.2	96.84	204.97	256.4
7.5	35	244.6	248.0	97.01	236.71	256.7
10.0	14	245.6	252.8	79.94	249.19	256.4
12.5	9	249.1	262.7	85.67	329.75	259.9
15.0	8	251.5	266.0	68.07	481.28	260.2
17.5	3	251.4	268.0	64.60	429.75	259.8
20.0	2	252.0	267.0	62.38	449.28	260.6
25.0	1	243.6	241.1	75.53	215.8	255.4
40.0	2	248.2	263.6	52.44	300.6	257.9

Table 2: Experimental Data based on HP.261 traces

additional latency in the frame play-out time, and hence are suitable for interactive applications.

One main implication of our work is that many motion compensation prediction-based codecs, such as MPEG, and H.261, are still useful for Internet interactive video transmission over a lossy network. Some of the disadvantages of the motion compensated codecs cited in the literature [8, 9, 7] include (1) computational complexity, (2) error resilience, (3) tight coupling between the prediction state at the encoder and that at the decoder, and (4) compute-scalable decoding. In this paper, we showed that the H.261 equipped with our REC schemes achieves comparable error resilience to that of INTRA-H.261. We also believe that some of the other disadvantages can be overcome with a simple modification to the codecs. For instance, the compute-scalable decoding can also be achieved by decoding only periodic frames and shedding off the computational load for decoding non-periodic frames in PTDD.

Having said some of the disadvantages of motion-compensated codecs, we would like to emphasize one of their advantages over INTRA-H.261, which is high compression efficiency. Although INTRA-H.261 gives good error resilience, the low compression efficiency of INTRA-H.261 will make it very difficult to obtain very high quality video for a low bit rate transmission. As pointed out in [12], in a multicast group, while some receivers want real-time video, others may want to watch or record the transmitted video. These observers may want the highest quality that the video source could provide although they can tolerate a longer play-out delay. At the same time, they may have only a small amount of bandwidth allocated for the video.

Motion-compensated encoding allows much better video quality for a given data rate than intra-coding. Our scheme allows real-time receivers to view video in a comparable quality as intra-coding schemes. At the same time, it allows the same encoded video to be multicasted to other non-real-time receivers. These receivers can view the video in very high quality by adding an additional delay (i.e., control time) before the display of the first frame. On the other hand, for intra-coded video to be sent on a low bandwidth network, its quality needs to be reduced substantially. In this case, although real-time receivers may get the video in similar quality as the motion-compensation scheme with REC, non-real-time receivers will receive a poor quality video. Note that this feature is different from media scaling [8] where receivers with a higher network capacity always get a higher quality image. Here, this feature allows even the receivers with a low network capacity to get a high quality while trading interact-ability. The motion compensated codecs equipped with a REC scheme can provide this feature as they generally give better compression efficiency and lost packets can be recovered through retransmission.

References

[1] J. Bolot and A. Vega-Garcia, " The case for FEC-based error control for packet audio in the Internet," to appear in ACM Multimedia Systems.

[2] P. Brady, "Effects of Transmission Delay on Conversational Behavior on Echo-Free Telephone Circuits," *Bell System Technical Journal*, vol. 50, no. 1, pp. 115 – 134, Jan. 1993.

[3] B. Demsey, J. Liebeherr, and A. Weaver, "On retransmission-based error control for continuous media traffic in packet-switching networks," *Computer Networks and ISDN Systems,* 1996.

[4] T. Dorcey, "CU-SeeMe desktop videoconferencing software," *ConneXions,* vol. 9, no. 3, Mar. 1995.

[5] R. Fredrick, *Network Video(nv)*, Xerox Palo Alto Research Center.

[6] X. Li, S. Paul, P. Pancha, and M. Ammar, "Layered video multicast with retransmission (LVMR): evaluation of error recovery schemes," *Proceedings of the Sixth International Workshop on Network and Operating System Support for Digital Audio and Video,* St Louis, May 1997.

[7] S. McCanne and V. Jacobson, "*vic*: a flexible framework for packet video," in *Proceedings of ACM Multimedia'95*, San Francisco, CA, Nov. 1995, pp. 511–522

[8] S. McCanne, M. Verrerli, and V. Jacobson, "Low-complexity video coding for receiver-driven layered multicast," *IEEE Journal on Selected Areas in Communications*, vol. 16, no. 6, pp. 983-1001, August 1997

[9] S. McCanne, V. Jacobson, and M. Vetterli, "Receiver-driven Layered Multicast". *Proceedings of ACM SIGCOMM*, August 1996, Stanford, CA, pp. 117-130

[10] C. Papadopoulos and G. Parulkar, "Retransmission Based error control for continuous media applications," *Proceedings of the Sixth International Workshop on Network and Operating System Support for Digital Audio and Video,* pp. 5–12, 1996.

[11] G. Ramamurthy, and D. Raychaudhuri, "Performance of packet video with combined error recovery and concealment," *Proceedings of INFOCOM'95*, pp. 753–761, April 1995.

[12] R. Xu, C. Myers, H. Zhang, R. Yavatkar, "Resilient multicast support for continuous-media applications," *Proceedings of the Sixth International Workshop on Network and Operating System Support for Digital Audio and Video,* St Louis, May 1997.