

FEC-based Loss Recovery for Interactive Video Transmission – Experimental Study*

Injong Rhee [†]

Srinath R. Joshi

Department of Computer Science

North Carolina State University

Raleigh, NC 27695-7534, USA

{rhee, srjoshi}@csc.ncsu.edu

November 1998

Abstract

Real-time interactive video transmission in the current Internet has mediocre quality because of high packet loss rates. Loss of packets belonging to a video frame manifests itself not only in the reduced quality of that frame but also in the propagation of that distortion to successive frames. This error propagation problem is inherent in any motion-based video codec because of the interdependence of encoded video frames. Since packet losses in the best-effort Internet environment cannot be prevented, minimizing the impact of these packet losses to the final video quality is important. In this paper, we present a new forward error correction (FEC) technique that effectively alleviates error propagation in the transmission of interactive video. The technique is based on our recently developed error recovery scheme called *Recovery from Error Spread using Continuous Updates* (RESCU). RESCU allows transport level recovery techniques previously known to be infeasible for interactive video transmission applications to be successfully used in such applications. The proposed FEC technique can be very useful when the feedback channel from the receiver is highly limited, or transmission delay is high. Both simulation and Internet experiments indicate that our proposed FEC technique effectively alleviates the error spread problem and is able to sustain much better video quality than H.261 or other conventional FEC schemes under various packet loss rates.

Key words: Packet Loss Recovery, Interactive Video, Error Propagation, Forward Error Correction, Video conferencing.

*This work is supported by in part by Faculty Research and Professional Development Fund of North Carolina State University.

[†]contact author: email: rhee@csc.ncsu.edu, Phone: 919-515-3305, Fax: 919-515-7925

1 Introduction

Transmitting high-quality, real-time interactive video over lossy networks such as the Internet and wireless networks is very challenging. Because of limited bandwidth on networks and the bandwidth-hungry nature of video, video transmission requires extremely high compression efficiency. However, state-of-the-art video compression standards (MPEG, H.261) are not designed for transmission over a lossy channel. Although they can achieve very impressive compression efficiency, even small data losses can severely degrade video quality. A few bit errors in encoded data can cause the decoder to lose synchronization in the encoded stream, and can render useless all the data received until the next synchronization point. Furthermore, motion estimation and compensation in these codecs pose an even more severe problem, namely *error propagation* (or *error spread*). Motion estimation removes temporal redundancy in successive video frames (inter frame) by encoding only pixel value differences (prediction error) between a currently encoded image and a motion-predicted image created from a previously encoded image (reference frame). Image distortion in a reference frame can propagate to its succeeding frames and becomes amplified as more bits are lost.

Most of earlier work on loss recovery focuses on repairing packet losses before the scheduled display times of those video frames contained in lost packets (e.g., [31, 28, 21, 39, 1, 36, 20, 23]). However, this approach is ineffective for interactive video because data losses inevitably occur in packet-switched communication, and detecting and repairing losses incur latency. To handle this latency, existing techniques introduce additional delays in frame display times. However, delaying frame playout times greatly impairs the interactiveness of video communication.

Many researchers [31, 28, 21, 39] have proposed using retransmission of lost packets by delaying frame playout times to allow arrival of retransmitted packets *before* the display times of their video frames. Any packets received after their display times will be discarded. In these schemes, the display time of a frame is delayed by at least three one-way trip times after its initial transmission (two for frame transmission and one for a retransmission request). This latency can significantly impair the interactiveness of any video applications under the current Internet.

Forward error correction is also commonly proposed for error recovery of continuous media transmission [20, 1, 17, 6, 3]. However, conventional FEC schemes do not work well for interactive video. This is because unless the playout time of a frame is delayed, both the original packets and their parity packets must be transmitted within the same frame interval, rendering the schemes very susceptible to burst loss. Moreover, since FEC is applied to a block of packets, before FEC packets are computed and transmitted, large delay must transpire.

In our earlier work [32, 33], we proposed an entirely complementary approach to the above-mentioned approaches by focusing on eliminating error propagation when distortion on displayed images happens. Our point of departure from existing approaches was that packets do not have to arrive in time for them to be “useful” for display of that video frame. Of course, if packets can arrive before the display times of their frames, that is optimal. However, due to packet losses and high latency, repair packets inevitably arrive “late”, causing distortion in displayed images which starts to propagate to following frames. We believe that these late repair packets can be used to stop error propagation. In motion-compensated codecs, the correct display of a frame depends on the successful reception of all of its reference frames. If we buffer displayed frames and use late packets to restore errors in the buffered frames, error propagation

can be stopped. This is because the buffered frames will be used as reference frames for later frames. We named this approach *Recovery from Error Spread using Continuous Updates* (RESCU).

RESCU has been shown effective for interactive video transmission when retransmission is used to recover lost packets and round trip delays are small [32, 33]. Retransmission tends to prolong error propagation because of the delay involved in detecting and retransmitting lost packets. Moreover, in some networks such as wireless cable modems and direct satellites, feedback channels are highly contentious, and bandwidth-limited. Thus, in these networks, frequent transmission of feedback to the sender is too expensive.

Contribution The main contribution of this paper is the development of a new FEC technique for interactive video. By incorporating this FEC technique, RESCU can perform very effectively in an environment where little or no feedback is available, or transmission delay is too high for retransmission to be effective. Our FEC scheme clearly differs from the conventional ones in that FEC packets can be transmitted over a longer period than a single frame interval without introducing any delay in frame playout times. Since RESCU uses FEC packets to restore buffered reference frames (called periodic frames), FEC packets can be transmitted over a relatively longer period, interleaving with the packets of other (non-periodic) frames to help reduce the effect of bursty losses. Note that this interleaving is different from link-level symbol interleaving [41, 13] where symbols from multiple codewords are interleaved. The granularity of our proposed interleaving is much larger and thus, more effective. Since RESCU makes non-periodic frames temporally depend only on the immediately previous periodic frame and unlike retransmission, FEC involves no feedback delay, the proposed technique incurs shorter recovery delays and accordingly shorter error propagation. Thus, it can be effective for high frame rate transmission over lossy, high-latency networks.

Organization Section 2 briefly describes the RESCU scheme and presents the new FEC technique for RESCU, Section 3 presents our simulation and experimental results, and Section 4 contains the description of related work. Section 5 summarize our work in this paper and discusses the impact of our work and future directions.

2 Error Recovery Techniques

2.1 Recovery from Error Spread using Continuous Updates (RESCU)

In this section, we briefly summarize the concept of RESCU for the readers who are not familiar with RESCU. For more details, see [32, 33].

We base our discussion on H.261, an International Telecommunication Union (ITU) video standard. 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 displacement 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

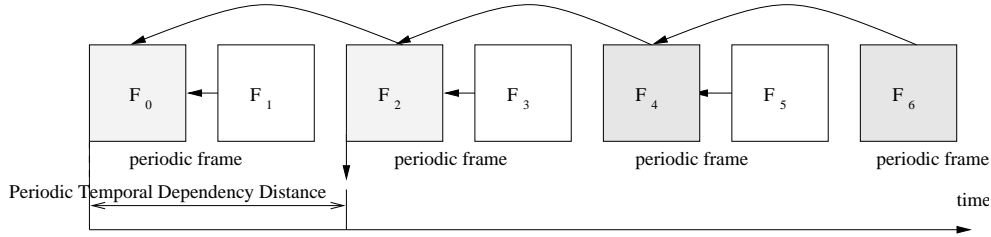


Figure 1: Periodic RESCU with PTDD 4 and NPF 1

obtained by simply cut-and-pasting the matching image blocks from its R-frame are encoded along with the motion vectors.

If you lose any packet(s) belonging to a video frame, not only is that frame shown with distortion but the error also propagates to the succeeding frames till the next synchronization point (an I-frame). However I-frames cannot be sent often since they have a large number of packets, which would increase the bandwidth too much. Note that in a high-frame rate real-time interactive video transmission using H.261, we cannot effectively use transport level recovery techniques without delaying playout times because of the network latencies involved in detecting and repairing lost packets. Such playout delay severely affects the interactivenss of video applications. Hence in H.261 an I-frame needs to be sent to stop error-propagation.

In RESCU, 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. By using the late packets to restore R-frames, we can stop errors due to lost packet from spreading to following frames.

The *deadline of a packet* is defined to be the time by which the packet must arrive at the receiver to be useful. RESCU allows this deadline to be arbitrarily adjusted through the *temporal dependency distance* (TDD) of a frame which is the minimum number of frame intervals between that frame and its temporally dependent one. By extending TDD, we can arrange a frame to be referenced much later than its display time. This adjustment effectively masks out the delay in repairing lost packets. For instance, we can make every p -th frame (which we call *periodic frame*) reference the one in p frame intervals away. This TDD of periodic frame is called *periodic TDD* (PTDD). Every non-periodic frame (frames between two consecutive periodic frames) depends only on its immediately preceding periodic frame. Thus the TDD of the non-periodic frames is between 1 and PTDD. Although a periodic frame may be displayed with errors because of some losses of its packets, if these losses can be recovered within a PTDD period through a transport-level recovery mechanism such as retransmission, the errors will stop propagating beyond the next periodic frame. Thus, the main benefit of the RESCU scheme is that it allows more time for a transport-level recovery mechanism to be successful. Also, errors in non-periodic frames do not propagate at all because all non-periodic frames temporally depend only on periodic frames. Note that extending TDD does not affect frame playout times because all frames are still displayed at their scheduled display times.

Figures 2 and 3 show video clips from a proof-of-concept experiment. The distortion in the second picture of Figure 2 is due to packet losses, which propagates even though the rest of frames are correctly received in time.



Figure 2: Error propagation



Figure 3: RESCU stops error propagation

However, in Figure 3, when RESCU is used, the quality of the third picture immediately bounces back when the packets for the second frame are recovered before the decoding of the third frame.

The encoder can determine PTDD based on the current traffic conditions. However, if network conditions change (e.g., latency becomes longer) after a periodic frame is sent, that frame on the way to the destination might have too short PTDD for the changed environment. This could cause the periodic frame to miss its deadline, resulting in error propagation. Since the frame has been already encoded and transmitted, there is nothing that the encoder can do to save the frame. *Cascaded recovery* alleviates this problem without involving the encoder. In RESCU, each periodic frame temporally depends on the previous periodic frames. Thus, by employing more reference frame buffers for periodic frames in the decoder, more late packets can be used to restore a sequence of erroneous periodic frames. Cascaded recovery allows packet deadlines to be extended *at the receiving times*, but not at the encoding times.

When buffers are not available, PTDD is too short, or more data packets are lost than parity packets, periodic frames may not be recoverable. This leads to error propagation. To prevent this type of error propagation, the receiver can detect losses in periodic frames not recovered even after a PTDD period, and can notify the sender about those irrecoverable losses. The notification triggers the sender to code the next frame as an intra-frame. The intra-frame stops error propagation due to the earlier losses because the intra-frame does not have temporal signal dependency with any of frames transmitted earlier. This technique, called *replenishment*, can be adopted in any scheme (including H.261). However, it significantly increases bandwidth consumption. The main motivation for RESCU is to provide error-resilience without much impact on the bandwidth.

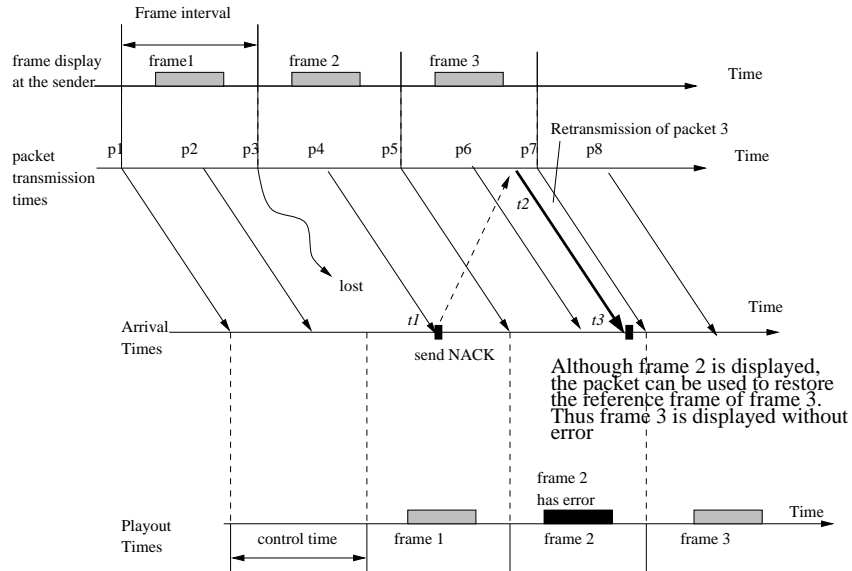


Figure 4: The recovery of frames using retransmission (from [32]).

2.2 Our Prior Work

In our previous publications [32, 33], we presented how retransmission and layered coding can be used along with RESCU. Retransmission is the most commonly used error recovery technique in reliable transport. The sender (or another receiver in the multicast case) simply retransmits the packets reported missing by a receiver. For interactive video transmission, conventional schemes require retransmitted packets to arrive within a single frame interval after the time that they are first lost. However, the associated delays in detecting and retransmitting the lost packets are often larger than one frame interval. In contrast, RESCU allows these retransmission delays to be masked out since retransmitted packets need to be received only within a PTDD period.

Figure 4 illustrates the error recovery using retransmission in a video stream containing two packets per frame and PTDD 2. Packet 3 is lost, and the receiver receives packet 4 at time t_1 and recognizing that packet 3 is not received, sends a retransmission request (NACK) to the sender. The sender gets the NACK at time t_2 and retransmits packet 3. The retransmitted packet arrives at time t_3 which is before frame 3 is displayed. Packet 3 is now used to restore the R-frame of frame 3 (frame 1), so frame 3 can be decoded and displayed without an error.

This retransmission technique is fundamentally different from other retransmission schemes [31, 28, 21, 39] in that it does not introduce any delay in frame playout times. In interactive video conferencing, introducing a delay in frame playout times severely impairs interactive communication.

2.3 New FEC-based Loss Recovery

One main disadvantage of retransmission-based error recovery is that its performance is too sensitive to transmission delay. Although RESCU can accommodate larger transmission delay than conventional retransmission schemes, a

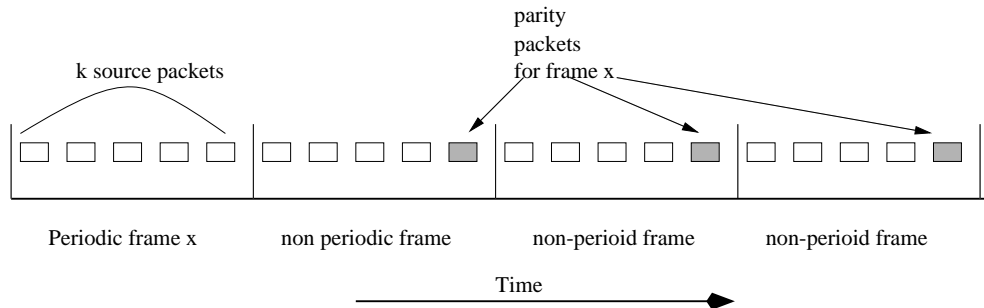


Figure 5: Interleaving of packets in RESCU of PTDD 4 with (5,7) RSE encoding

larger transmission delay requires a larger PTDD period. As PTDD increases, compression efficiency gets lower because two consecutive periodic frames may not have much temporal redundancy, and the TDD of non-periodic frames also increase. In addition, packet losses in periodic frames can be restored only after one round trip time. Thus, during the time, non-periodic frames can have errors propagated from the erroneous periodic frames.

Furthermore, over direct broadcast satellite links or cable modems, feedback channels are highly bandwidth limited and contention-based. Some mobile wireless hosts simply do not have extra capacity to frequently send feedback to the sender. Furthermore, in video multicast, it is not desirable to have direct feedback from each receiver to the sender because of the known ramification of the acknowledgment implosion problem. In all of these circumstances, using feedback is very limiting. Even when we could use as much feedback as we may wish, retransmission-based techniques which have very high sensitivity to network latencies become very ineffective.

FEC is a very compelling alternative for all these environments. A Reed Solomon Erasure correcting code (RSE code), such as the one described by McAuley [25], is a commonly used FEC encoder where k source packets of P bits, d_1, d_2, \dots, d_k , are encoded into $n > k$ packets of P bits (i.e., $n - k$ parity packets, p_1, p_2, \dots, p_{n-k}). This n packets is called as a *FEC block*. The RSE decoder at the receiver side can reconstruct the source data packets using any k packets out of its FEC block. Efficient (n, k) RSE encoding and decoding algorithms have been developed and implemented to achieve real-time performance [2, 15, 34, 27]. For instance, the throughput of the software coder by Rizzo [34] can achieve 11 MB/s on a 133 MHz Pentium.

In RESCU, the original data of a periodic frame packetized into k source packets are transmitted at the frame interval of the periodic frame and then its $n - k$ parity packets are transmitted over the PTDD period. The transmission time of each parity packet is evenly spaced over the period, interleaving with the packets of other frames. Figure 5 shows a transmission sequence of data and parity packets in RESCU. When several data packets are lost, the corresponding periodic frame and accordingly, its dependent non-periodic frames will be displayed with errors. However, as following parity packets are received to recover the original data packets, the periodic frame can be restored. This will cause the remaining non-periodic frames within the PTDD period and the next periodic frame to be displayed without errors if these frames are correctly received.

Conventional FEC schemes can be categorized into two kinds. One type is to transmit both data and their parity packets within the same frame interval. The other scheme is to transmit the parity packets in later frame intervals than

the interval in which data packets are sent. The former scheme is susceptible to burst packet losses and since FEC is applied to a block of packets, before FEC packets are computed and transmitted, large delay must transpire. The latter scheme has to introduce additional delays in frame playout times to allow enough time for the receiver to receive parity packets and restore the currently displayed images. Although these schemes can be effective for a one-way, near-real-time video transmission, it seriously impairs interactive video communication.

In contrast, our scheme does not have these drawbacks. In RESCU, we can interleave FEC packets of a periodic frame and data packets of non-periodic frame thus greatly minimizing chances of (more than one) FEC packets getting lost in a burst. As soon as we receive enough parity packets (equal to the number of lost data packets of the periodic frame), we can restore the periodic frame and the succeeding frames see absolutely no traces of any distortion, thus stopping error-propagation quickly.

When more data packets are lost than parity packets received, periodic frames may not be recoverable. This leads to error propagation. To prevent this type of error propagation, we can apply intra-frame replenishment. However replenishments are costly in terms of bandwidth usage and hence we should try to avoid it as much as possible. Long loss periods are relatively rare and lost packets are generally dispersed throughout the stream of packets received at a destination. Hence we believe that a small portion of the total bandwidth is enough for FEC packets. In our experimental results we show that about 10 percent of the bandwidth is enough to recover from losses in most of the cases and greatly reducing the need for replenishments.

3 Experimental Result

We showed the efficacy of RESCU with retransmission in [33, 32]. However retransmission scheme can be effective only when RTT is small enough.

The main objective of the experiments is to show that FEC integrated with RESCU is a very effective error recovery technique for real-time interactive video transmission over the Internet. We show this through video transmission experiments over transpacific Internet connection from the East coast of the US. We implement the RESCU codec by modifying an implementation of H.261. We use this codec to compare the performance of our FEC and ARQ scheme in terms of the end video quality and bandwidth overhead. For convenience, we call the FEC scheme integrated with RESCU as *RESCU-FEC* and the retransmission scheme as *RESCU-REC*.

Our experiments are conducted with the following goals in mind.

1. We show the final video quality sustained by RESCU-REC and associated bit efficiency. We also show the high sensitivity of RESCU-REC to varying network delays.
2. We show the final video quality sustained by RESCU-FEC and associated bit efficiency. We also show the insensitivity of RESCU-FEC to varying network delays and its ability to quickly recover lost packets.
3. We show the bit efficiency advantage of RESCU-FEC over H.261 in achieving the same level of error resilience. H.261 can improve error resilience by transmitting intra-frames more frequently. We find out how much bandwidth is required for H.261 to achieve the same video quality as RESCU-FEC.

4. We show the performance advantage of RESCU-FEC over a conventional FEC scheme which transmit the FEC redundant packets of each frame at the same frame interval as the original frame. The advantage can be shown by finding out how much bandwidth is required for the conventional scheme to achieve the same video quality as RESCU-FEC.

3.1 Testing Methodology

RESCU is implemented based on a H.261 codec. We use the full-search motion estimation for all experiments and PTDD is varied for different experiments. RESCU with one additional buffer is used in the retransmission scheme for cascaded recovery. We use the test image sequences that are obtained from the MPEG-4 test sequences encoded by Telenor H.263 encoder. In this paper, we show the results for a MPEG-4 class A test video sequence called *container*. For every experiment, the frame rate is set to 10 frames per second. The image size of CIF (352×288 color) is used for experiments. Both the codecs (RESCU and H.261) use a default quantization step size 8. The test video sequence is first compressed using each codec and the encoded video frame is packetized into approximately 256-byte packets such that the individual packets contain an integral number of Macro Blocks. In RESCU-FEC we also embed a specified number of parity packets evenly spaced in the PTDD. We then generate a packetized sequence corresponding to 190 frames. This sequence is replayed several times for about 2 minutes (1200 frames). The replay does not affect the integrity of the experiment because the first frame is always intra-coded in all the tested schemes.

Simulation Method We model burst packet losses using a two state continuous Markov chain $\{X_t\}$ where $X_t \in \{0, 1\}$. A packet transferred at time t is lost if $X_t = 1$ and not lost if $X_t = 0$. The infinitesimal generator of this Markov chain is

$$Q = \begin{pmatrix} -\mu_0 & \mu_0 \\ \mu_1 & -\mu_1 \end{pmatrix}$$

The stationary distribution associated with this chain is $\pi = (\pi_0, \pi_1)$ where $\pi_0 = \mu_1 / (\mu_0 + \mu_1)$ and $\pi_1 = \mu_0 / (\mu_0 + \mu_1)$. Let $p_{i,j}(t)$ be the probability that the process is in state j at time $t + \tau$ given that it was in state i at time τ . Let λ be the packet transmission rate, b the expected number of consecutively lost packets, and p the packet loss probability. Then $\mu_0 = -\pi_1 \lambda \log(1 - 1/b)$ and $\mu_1 = \mu_0(1 - p)/p$. The network delay is modeled by the exponential distribution with the mean delay D . Network conditions are characterized by the loss probability p , the mean burst loss length b , and the mean network delay D .

Given a packetized sequence, we obtain transmission traces of the sequence which contain information about the delivery time of each packet, and the number of retransmission attempts. Using the above network model, we calculate the delivery times. When retransmission is used for recovery, for each lost packet that belongs to a periodic frame, we find out whether the packet is received by retransmission before its deadline. The deadline is determined by the time period between the initial deadline of the packet and the time when the retransmission is made. Each retransmission attempt costs one round trip time which is calculated from the network model. A packet can be retransmitted as many

times as it is allowed by its deadline. When the packet is received by retransmission, we record the time that the packet is received.

After obtaining a transmission trace of a video sequence, we run the decoder on the trace to measure the image distortion due to packet losses. The image distortion is computed using the peak signal-to-noise ratio (PSNR) of decoded images over the original images.

Internet-Transmission Test We conduct actual video transmission tests over the Internet from Korea to US. These testing sites are chosen because transmission delays between two sites are frequently over 300 ms. The experiments conducted are intentionally biased against retransmission to show the effectiveness of RESCU-FEC over RESCU-REC over the given environment. The transmission tests were conducted every 45 minutes between Oct.10 and Oct. 13 to obtain traces. Each packet of a frame is transmitted at a regular interval by the given frame rate (10 fps) and the number of packets within that frame.

The automatic repeat request (ARQ) scheme of RESCU-REC 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 two periodic frames. The sender estimates the current RTT based on the sending time of the acknowledgment and its reception time. The sender maintains a record of most recent timestamp of the retransmission of each packet and does not retransmit the packet unless the difference between that timestamp and the current time is at least as large as the RTT estimate. This mechanism reduces unnecessary retransmissions. Also, whenever the sender comes to know that a periodic frame has not been recovered even after 2 PTDDs, it sends a replenishment (I-frame) to stop error-propagation.

Each packetized sequence of RESCU-FEC includes one FEC-redundant packet per frame. Within one PTDD period, the number of FEC packets is equal to PTDD. Thus, as PTDD increases, although compression efficiency gets lower, the rate of FEC redundant data over the original data does not change.

For fair comparison between FEC and REC, in the actual transmission tests, we transmit only the packetized sequences of RESCU-REC (not FEC). For each transmission test, we obtain a 2-minute trace that records the packet sequence numbers, the arrival times of all received packets and the number of retransmission attempts. Then, each of the obtained traces T are mapped to the packetized sequences of H.261 and RESCU-FEC. We first obtain a 2 minute length of a packetized sequence S of H.261 and RESCU-FEC as if they are transmitted in a real test. Each packet p in trace T is mapped to a packet q that has 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 q as lost.

This mapping technique provides a very accurate comparison of various transmission schemes because the sequence of all the schemes are mapped to the same trace. The mapping is possible because RESCU-FEC and H.261 have very little or no feedback to be given to the sender. RESCU-FEC uses average RTTs seen by the RESCU-REC traces and an exponential distribution to calculate the time of arrival of a replenishment request when it needs one.

We obtained 66 traces for PTDD 3 and 68 each for PTDDs 6 and 9 through actual transmission . We then mapped these traces to the RESCU-FEC and H.261. Using these transmission traces of the video sequence, we run the off-

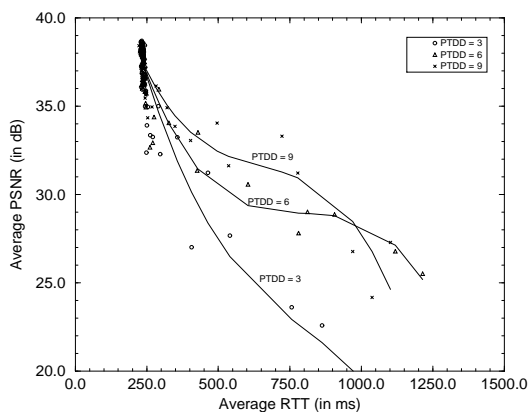


Figure 6: Average PSNR of RESCU-REC with different round trip network delays

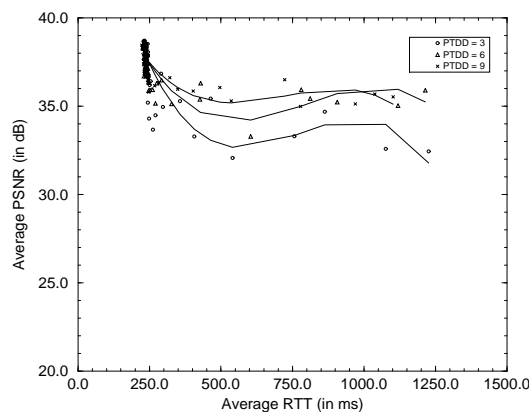


Figure 7: Average PSNR of RESCU-FEC with different round trip network delays

line decoder on the traces to measure the distortion in the video frame due to packet losses. The image distortion is computed using the peak signal-to-noise ratio (PSNR) of decoded images over the original images.

3.2 Experimental Results

3.3 Comparison of RESCU-FEC with RESCU-REC

The disadvantage of RESCU-REC is its sensitivity to transmission delay. When transmission is too high, retransmitted packets may not arrive before their deadlines, causing error propagation. This problem is completely eliminated in RESCU-FEC.

Figures 6 and 7 clearly show the advantage of RESCU-FEC over RESCU-REC when the round trip time (RTT) is large. These figures are obtained from the Internet transmission tests. The lines are the results from cubic-order regression based on experimental data points.

In Figure 6, RESCU-REC shows good video quality under low network latency (less than 250 ms) even with a short PTDD of 3. However in all cases, RESCU-REC is seen to be highly sensitive to the network latency. When network delays are long and PTDD is not sufficiently large, most retransmitted packets are not received before their deadlines and video quality degrades. RESCU-REC shows total ineffectiveness under high RTTs. As PTDD becomes larger, the video quality generally improves, but this causes low compression efficiency. In Figure 7, RESCU-FEC is clearly much less sensitive to RTTs. As RTTs increases, its performance degrades a little because high latency usually occurs at the time of congestion. However, its sustained performance is very high compared to that of RESCU-REC.

Figures 8 and 9 show the video quality of RESCU-FEC and RESCU-REC over various loss rates. When the loss rate becomes larger than 12%, both techniques suffer their quality. Particularly, the performance of REC becomes

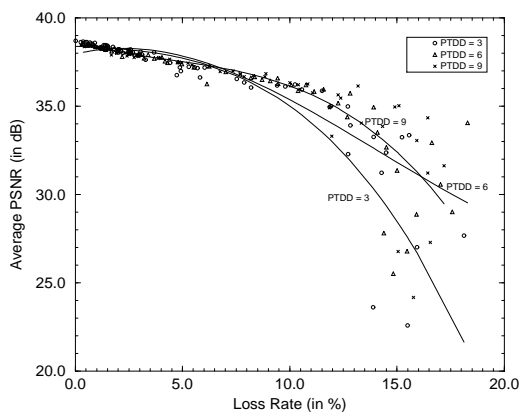


Figure 8: Average PSNR of RESCU-REC with different loss rates (Internet experiment)

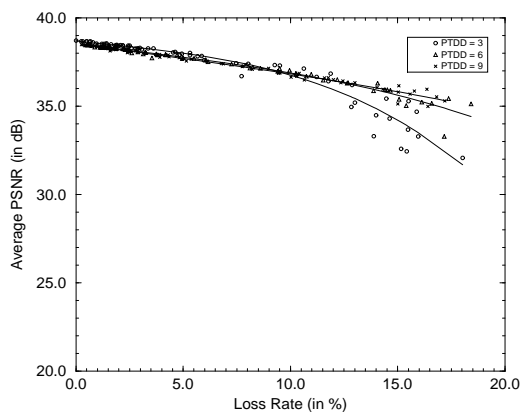


Figure 9: Average PSNR of RESCU-FEC with different loss rates (Internet experiment)

highly unpredictable, and sometimes it gives very low video quality (less than 30dB). This is even if REC uses one additional buffer for cascaded recovery (explained in Section 2.1) (i.e., the effective deadline of cascaded REC is twice as much as given PTDD).

The better performance of FEC, however, comes at the expense of higher bit rates. Figures 10 and 11 The bit rate of FEC is generally 5 to 8% higher than that of REC. The fluctuation in the bit rate of FEC for the transmission tests with the same PTDD is due to the coarseness of timers in controlling transmission rates. There are two reasons why REC gives lower bit rates. First, retransmission occurs only when packet losses occur while FEC redundant packets are continually sent regardless of packet losses. Second, REC is less sensitive to the loss burst length. Since retransmitted packets take more than one RTT to arrive, if loss burst starts at the time of the first loss that triggers retransmission, the burst is most likely to be ended by the time when retransmitted packets arrive to the receiver. However, this is not the case for FEC-redundant packets. Although the effect of loss burst is much less critical for our FEC technique than for conventional FEC techniques, the performance of our FEC is still affected by loss burst. For instance, as the loss rate gets larger than about 10%, the total number of losses from a periodic frame is frequently more than 3. This means that that periodic frame cannot be recovered with only three FEC packets. This causes the receiver to send a replenishment request to the sender. The large occurrence of replenishment is the reason for the surge in the bandwidth usage for PTDD 3 under a loss rate higher than 13% in Figure 11.

Simply increasing PTDD without affecting the rate of bit redundancy over the original bit rate significantly improves the quality under high loss rates. If we increase the PTDD to 6 and have 6 FEC packets, we still have approximately the same percentage of total bandwidth allocated to FEC redundancy. The results in Figure 9 shows that PTDD 6 provide an adequate amount of protection for all loss rates we observed with only a small number of replenishments. At high loss rates, when RESCU-FEC with PTDD 3 is not able to sustain good video quality in spite of very high bandwidth, RESCU-FEC with PTDD 6 consistently gives better PSNR (more than 35dB) without any increase in the

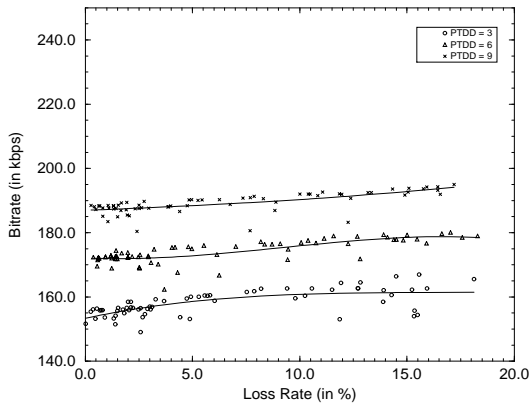


Figure 10: Average bit rate of RESCU-REC with different loss rates (Internet experiment)

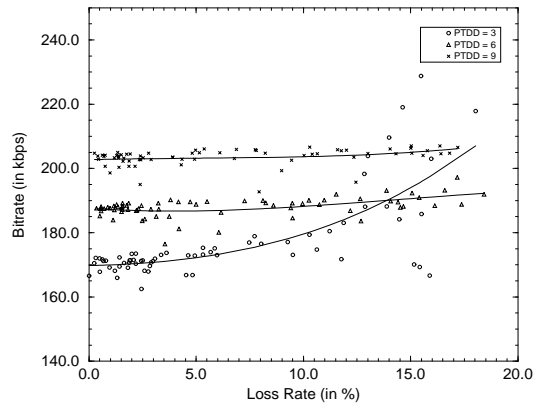


Figure 11: Average bit rate of RESCU-FEC with different loss rates (Internet experiment)

bandwidth usage. The results obtained for RESCU-FEC and PTDD 9 show that although the PSNR is always high, the gains are not significant enough for the added bandwidth usages.

To study the impact of burst losses to the image quality of RESCU-FEC, we look at the PSNRs of RESCU-FEC over the average burst lengths (AVL) of obtained traces (Figure 12). Most of long loss burst traces we obtained are very low loss rates. They gave long loss burst lengths because they include one or two occurrences of very long burst lengths (larger than 100 packets). When loss rates are low, these traces result in long average burst lengths. We have very few occurrences of long burst for high loss rates. Most traces grouped around 1 to 3 AVLs. For these AVLs, RESCU-FEC with PTDD 6 or 9 shows very good performance. Note that consecutive FEC packets within a PTDD period are very unlikely to lose in a loss burst since they are spaced by one frame interval (about 10 packets). Thus, burst losses happen mostly for the original periodic data packets. Since high loss rate traces have AVLs around 1 to 3, 6 FEC packets in PTDD 6 can effectively recover these losses. The short AVLs we obtained for most traces are aligned with Bolot's experiments [4] which indicate that long sustained loss bursts are rare in the Internet.

In order to further study the behavior of RESCU-FEC under high burst loss environments, we ran simulation experiments. Figures 13 and 14 illustrate the impact of burst losses in the performance of RESCU-FEC. The loss rate of 10% is applied to all simulation experiments. While the PSNR remains relatively the same over different burst lengths, the replenishment count shows the effect of bursts. Replenishment with an intra-frame occurs when RESCU-FEC fails. Thus, the count represents how effective RESCU-FEC with a given amount of redundancy can be. For each additional redundant packet with a PTDD period, the bandwidth increases by a factor of about 1.6% since each frame consists of about 10 packets. Clearly from Figure 14, we can see that two parity packets within PTDD 6 are not enough in all loss burst lengths tested – more than 48 replenishments within the 2 minute playout time were made in all cases. The count reduces as more redundant packets are added. It is also shown that a long loss burst length causes more replenishments. Under burst length 1, four parity packets are enough while under burst lengths 1.5 and 2,

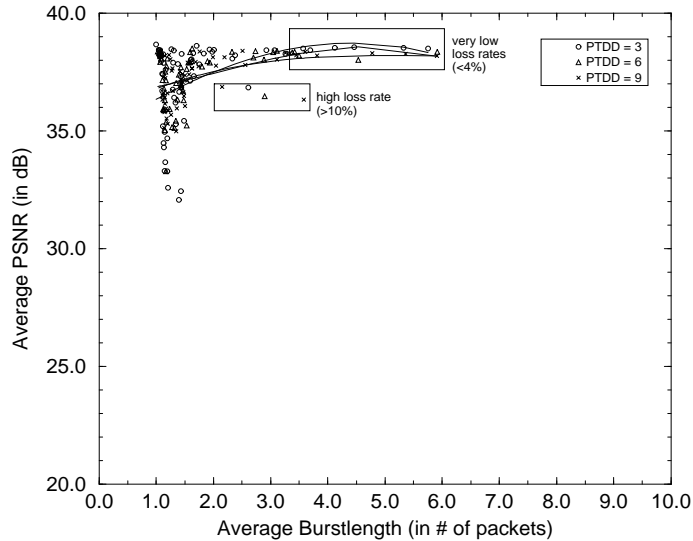


Figure 12: Average PSNR of RESCU-FEC over average burst length (experiment)

five and six parity packets are needed which is only 6-8% bit overhead. This indicates that RESCU-FEC can perform very well with only a small amount of redundancy even under high burst losses. Since replenishment confines error propagation very well, the video quality of RESCU-FEC does not show many variations under different burst lengths. However, when only two parity packet are used, the quality degrades quite severely.

In summary, results obtained for RESCU-REC suggest a clear correlation between round trip time and the performance of RESCU-REC and this is one of the main motivations for the design of RESCU-FEC. The advantage of FEC over retransmission is that FEC repairs lost packets more quickly than retransmission since no loss detection and feedback delays are incurred in FEC. In RESCU-REC, lost packets are detected only by a gap in received packet sequence numbers, and furthermore feedback has to travel to the sender to trigger retransmission. RESCU-FEC does not have these problems and is the reason we observe the insensitivity of RESCU-FEC to network latencies. The experiment result indicates that with a small amount of bit overhead (6-8%), RESCU-FEC can achieve good error resilience under significantly high loss rate and loss burst length.

3.4 Comparison of RESCU-FEC with H.261

We compare the bandwidth overhead of H.261 over RESCU-FEC in achieving comparable video quality. H.261 can improve its error resilience by transmitting intra-frames more frequently. We ran a series of experiments and observed that by having one I-frame every 5 frames H.261 can give video quality similar to that of RESCU-FEC with PTDD 6. The results are presented in Figures 15 and 16.

For loss rates up to 5%, H.261 gives a slightly better video quality than RESCU-FEC. This is because at low loss

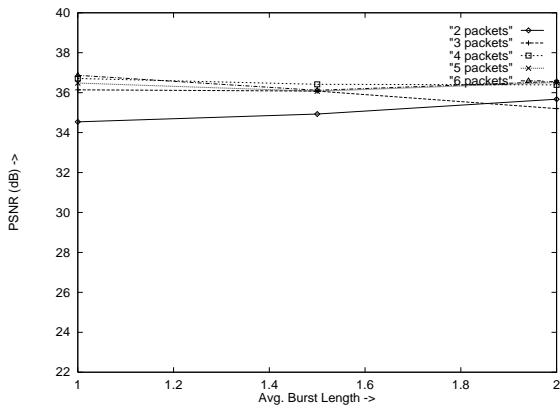


Figure 13: Average PSNR of RESCU-FEC with various numbers of parity packets within PTDD 6 over different loss burst lengths (simulation)

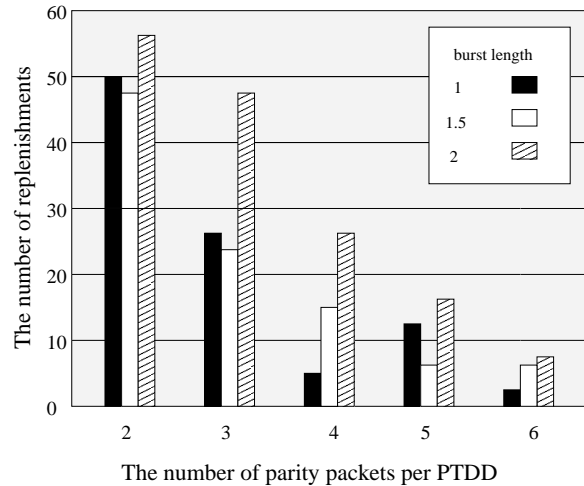


Figure 14: The number of replenishments for RESCU-FEC with various numbers of parity packets within PTDD 6 over different loss burst lengths (simulation)

rates, very few frames are likely to lose packets. Thus the effect of error propagation is not so pronounced. Also, the I-frame has much better quality than other frames. Since I-frames are sent every 5 frames we see a slight improvement in the overall video quality compared to RESCU-FEC. However, at all other loss rates, video quality sustained by RESCU-FEC is better than H.261. At high loss rates we see that RESCU-FEC can give about 1dB higher PSNR than H.261. Note that this is inspite the fact that H.261 is using 25 percent more bandwidth than RESCU-FEC. This is clearly seen in Figure 15.

In H.261, distortion in a video frame propagates till the next I-frame. This is because in H.261, every frame depends on the immediately previous frame and error propagation between two I-frames could be caused by loss of packets belonging to any frame. For moderate to high loss rates, packet losses could occur in any frame and hence the deteriorating quality as a result of error propagation. Recall that in RESCU, non-periodic frames uses only periodic frames as reference frames. Thus, distortion in non-periodic frames do not propagate at all. Also, FEC packets help recover the loss of packets of the periodic frame quickly and this allows some of non-periodic frames within a PTDD to be displayed without error propagation. Thus, continuous updates in a periodic frame subsequently improves the quality of non-periodic frames dependent on that periodic frame and also stops error propagation beyond the next periodic frame.

3.5 Comparison of RESCU-FEC with a conventional FEC scheme

In H.261, every frame temporally depends on its immediately preceding frame. Although this can give very good compression efficiency, error propagation can happen because of losses in any frame. Thus FEC schemes integrated with H.261 have to provide protection against losses in every frame. This means having to send original data packets

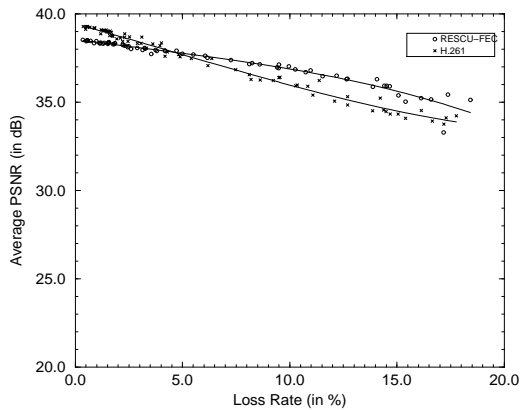


Figure 15: Average PSNR of H.261 with an Intra-frame every 5 frame

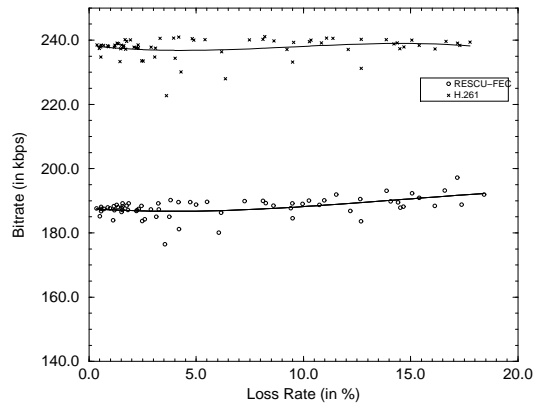


Figure 16: Average bitrate of H.261 with an Intra-frame every 5 frame

and FEC packets in that frame's interval.

We ran experiments to measure the bandwidth required to give performance comparable to that of RESCU-FEC. Traces for these tests were obtained by mapping the actual traces as done in generating traces for RESCU-FEC. When the receiver does not receive enough FEC packets to recover the lost data packets, error propagation would start. In order to stop error propagation, the receiver asks for replenishment. The timing of this replenishment is estimated based on the average RTT seen by the corresponding actual transmission trace and an exponential distribution. The results for experiments with 4 and 6 FEC packets per frame are shown in Figures 17 and 18.

The results show that at low loss rates, 4 FEC packets are most of the time enough to recover the losses in a frame. However, at moderate to high loss rates, we notice that often the receiver did not receive enough FEC packets to recover the frame and hence asked for replenishments. This is the reason we see increasing bandwidth. The time of arrival of the replenishment depends on the network delay and there can be error propagation between the time when the replenishment is requested and its arrival at the receiver. We see the effects of this error propagation on the video quality at high loss rates (which also tend to have longer network delays) which falls even as the bandwidth increases because of replenishments. In contrast to this the bandwidth is somewhat steady when there are 6 FEC packets per frame. This is because most of the times losses in the frame are recoverable. Thus we see that at low loss rates the conventional scheme with 4 FEC packets gives good video quality and at moderate to high loss rates, 6 FEC packets are needed for adequate protection.

In RESCU-FEC, non-periodic frames are not protected at all. Thus if they lose any packets, distortion will be seen in those frames. And periodic frames are recovered relatively later than in the conventional FEC since the FEC packets are dispersed in the PTDD. This is the reason why we see that the conventional FEC scheme gives a PSNR about 1dB higher than RESCU-FEC. However to embed a significant number of parity packets per frame incurs a large bandwidth overhead. But in H.261 based FEC schemes, this is absolutely essential since error propagation can happen because

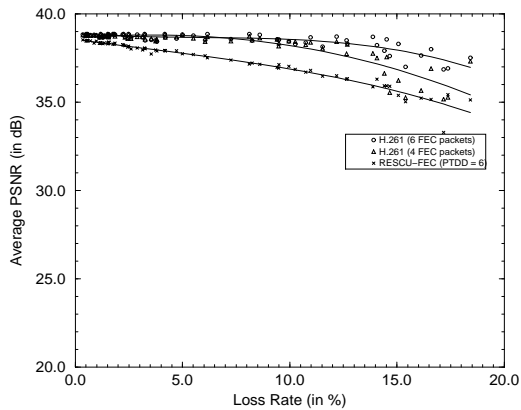


Figure 17: Average PSNR of RESCU-FEC and conventional FEC schemes over different loss rates lengths

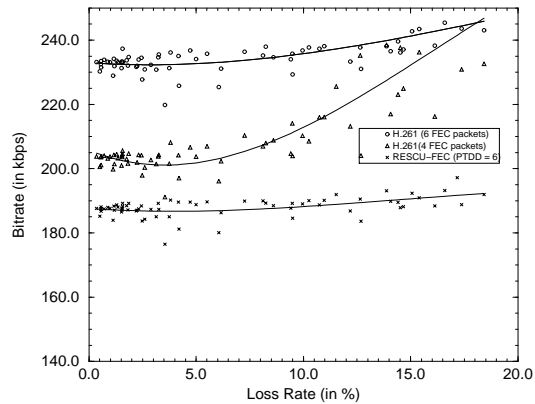


Figure 18: Average bit rate of RESCU-FEC and conventional FEC schemes over different loss rates lengths

of losses in any frame. However, in the RESCU-FEC scheme, distortion in non-periodic frame does not propagate at all. Hence we can sustain reasonably good video quality even when these frames are not protected. Longer PTDD means a somewhat reduced compression efficiency compared to H.261, but the trade off in FEC packets weighs in favor of RESCU-FEC. Also, in high frame rate interactive video, there is still considerable temporal redundancy between frames at a short distance away. All these factors contribute to considerably lower bandwidth needed by RESCU-FEC. Moreover, in the conventional FEC scheme, if there are data losses in the frame, we have to reconstruct those packets using FEC packets. This may happen at almost every frame since packets are frequently lost in the Internet. This may conflict with stringent real-time requirements of applications like videoconferencing. In RESCU-FEC we need to reconstruct the periodic frame only once in the PTDD. Also, packet losses in the Internet have been known to have a bursty nature with short average burst lengths. We believe that dispersing the FEC packets in the PTDD we minimize the chances of more than one of the FEC packets being lost in the burst and thus reduce the number of FEC packets needed to protect data packets which are likely to undergo burst losses.

4 Related Work

Error concealment is one of widely proposed error control techniques (e.g., [11, 16, 40, 24]). Although error concealment techniques can be combined with error recovery techniques, they are not strictly error recovery techniques [7], so we do not discuss them. We also focus on recovery techniques for video transmission.

4.1 Feedback-based recovery

Recently H.263+ incorporated two feedback-based techniques: error tracking and reference picture selection. Error tracking (ER) utilizes the intracoding of blocks to stop error propagation, but limits its use to severely impaired image regions only. ER requires the encoder to know the location and extent of erroneous image regions in displayed images. This can be achieved by feedback from the receiver. The receiver sends information about missing packets, and the encoder estimates the region of error propagation in the displayed images, and intracodes those blocks contained the region. The reference picture selection (RPS) mode allows the encoder to select one of several previously decoded frames as a reference picture for motion estimation. It is designed To support a coding technique called NEWPRED [19]. In NEWPRED, using feedback from the receiver, the encoder uses for prediction only those pictures that are reported to be received (in the *ACK mode*), or not reported missing (in the *NACK mode*). Since motion prediction is always based on the frames that are received by the receiver, error propagation is effectively eliminated.

Retransmission has recently attracted much attention for packet loss recovery in video transmission. Ramamurthy and Raychaudhuri [31] 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. Padopoulos and Parulkar [28] 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.

Retransmission is also used for video multicast. Li *et al.* [21] proposed an elegant scheme for multicasting MPEG-coded video. 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. Retransmission is used for high priority streams. The scheme is shown to be effective for non-interactive real-time video applications. In a video conferencing 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.* [39] contended 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.

Remarks A drawback of feedback-based techniques is that when the networks do not support feedback channels, they are useless. If feedback channels are supported, but limited in bandwidth, then those techniques requiring frequent feedback may not be effective. Another drawback of ER and NEWPRED is that they are not suitable for multicast. Since they compensate for packet losses by altering the coding pattern based on information about missing packets, if there is more than two receivers with widely different loss patterns, then their coding efficiency will drastically decrease. All of the mentioned retransmission techniques use frame playout delays to compensate for retransmission delays in high-latency networks.

4.2 Proactive recovery

Error propagation can be alleviated by intracoding more image blocks, but at the expense of compression efficiency. Several popular video conferencing tools, such as `nv`[10], `vic`[26] and `CU-SeeMe`[9], adopt this 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 that are contained in lost packets.

FEC has been successfully applied to audio transmission [5, 3, 29, 30]. There are only a few studies on applying FEC to video transmission. *Priority encoding transmission* (PET) [1, 20, 36], encodes different segments of video data with different priority. Each packet contains relatively more redundant information about the higher priority segments of the data, so the information with a higher priority can have a higher chance of correct reception. The PET scheme is also incorporated into `vic` [26] and is reported a good performance [38]. This good performance is partially due to `vic`'s intracoding which limits error propagation caused by loss of lower priority segments.

Bolot and Turetli [6] proposed an interesting FEC technique for packet video where a packet contains the redundant information of some of previous packets. The redundant information is created by encoding the image blocks contained in the previous packets with a large quantization step. They claimed that if the video source is not bursty, long burst losses are rare, and the proposed scheme would work well for video.

H.263+ also includes a similar technique called *independent segment decoding* (ISD) [18]. In the ISD mode, each video slice is encoded as an individual picture (or subvideo) independent of other slices. In particular, each slice boundary is treated just like picture boundary. ISD does not eliminate error propagation, but limits the extent of error propagation to a slice.

MPEG-4 adopted several error resilient techniques for wireless video transmission [37]. These include resynchronizations strategies, data partitioning, reversible VLCs, and header extension codes. Most of these techniques focus on preventing lost data from affecting the decoding of received data. For instance, loss of some header information affects all the data that tagged on the header although they are received correctly. The techniques minimize this effect.

Remarks A drawback of proactive techniques is that it wastes bandwidth under error-free transmission. Furthermore, in wireless mobile networks, the assumption made in [6] does not hold as long burst losses can be common due to fading and channel interference. Thus, the FEC schemes mentioned above are susceptible to burst errors because both data and FEC-encoded packets have to be transmitted at the same frame interval. Also if FEC-encoded packets are transmitted over a longer period, additional playout delays may be incurred. MPEG-4's techniques can be used along with RESCU to further reduce the effect of data loss.

4.3 Hybrid recovery

Hybrid techniques combine ARQ (retransmission) and FEC for better error resilience. There are two types of hybrid techniques: *type-I hybrid ARQ* [8], and *type-II hybrid ARQ* [22]. Type-I hybrid ARQ transmits both error detection and

correction data at the initial transmission of data. If the receiver cannot recover lost packets using the transmitted parity data, it requests retransmission of the same data from the sender. Type-II hybrid ARQ does not send any redundant data with the first transmission, but sends only parity data when retransmission is requested.

Liu and El Zarki [23] applied a hybrid ARQ technique for video transmission. They proposed a hybrid ARQ technique that combines the benefit of type-I and type-II techniques, and showed its efficacy for video transmission over wireless networks. They showed that using rate-compatible punctured convolutional (RCPC) codes [12, 14], one or two retransmission attempts achieve a low packet error rate under a perfectly interleaved Rayleigh fading channel.

Hybrid ARQ techniques were also studied in reliable multicast [35, 27]. While FEC helps reduce occurrences of independent losses, retransmission repairs correlated packet losses. They showed that hybrid ARQ reduces the bandwidth overhead of repairing packet losses in reliable multicast involving many receivers.

Remarks Although hybrid ARQ schemes work better than FEC or retransmission alone, they do not overcome the limitations of traditional recovery techniques. Since retransmission always incurs delays and FEC is still susceptible to burst losses, hybrid ARQ scheme still fall short of handling the retransmission' delays and burst losses without introducing delays in frame playout delays. Delaying frame playout reduces interactiveness and introduces annoying jitters.

5 Summary

In our earlier work we had shown that retransmission could be made a feasible alternative in error-recovery schemes for interactive video applications without introducing any artificial extension of frame playout times. The central idea in our earlier work was that correcting errors in a reference frame due to packet losses could be used to prevent error spread. However, performance of retransmission based schemes is very sensitive to network latencies.

In this paper, we presented a FEC technique based on the RESCU scheme. We showed through experiments that with little extra bandwidth we could quickly recover from errors and maintain a good overall video quality. Our result indicates that the FEC technique is very insensitive to network latencies. Under low to moderate loss rates (less than 10%), about 3 FEC packets are sufficient to recover the losses of periodic frames in most cases. But at higher loss rates we need more FEC packets for each periodic frame. By increasing PTDD to 6 frame distance, we have the same amount of total bandwidth allocated to FEC as in PTDD of 3 frames and yet show very good error resilience under high loss rates with the overall video quality of more than 35 dB. Bandwidth usage increases slightly as PTDD increases. The increased bandwidth was due to reduced compression efficiency associated with longer PTDD. This could be controlled by increasing quantization step-size a little bit without noticeable loss in quality. However this is more of an optimization problem and is considered out of the scope for this paper.

The result also showed that for packet loss rates encountered in actual experiments, RESCU-FEC with PTDD of 9 frames was an overkill from the bandwidth point with only insignificant gains over the performance of RESCU-FEC with PTDD 6. Then we showed the bandwidth advantage of RESCU-FEC over H.261 in achieving comparable error

resilience to that of RESCU-FEC. Our result clearly indicates that H.261 needs about more than 25% bandwidth to get the same video quality.

The main implication of our work is that for most of the practical situations on the current Internet, proactive techniques such as RESCU-FEC can very effectively alleviate the problem of error spread with only a small extra bandwidth. They also have the advantage in making a minimal use of the feedback channel. Hence this scheme has the potential to be very useful in multicast scenarios and wireless and satellite based communications.

Our future goal is to develop an adaptive technique which can adjust the amount of redundant data and the length of PTDD according to the network conditions.

References

- [1] A. Albanese, J. Blomer, J. Edmonds, M. Luby, and M. Sudan. Priority encoding transmission. *IEEE Transactions on Information Theory*, 42(6), November 1996.
- [2] N. Alon and M. Luby. A linear time erasure-resilient code with nearly optimal recovery. *IEEE Transactions on Information Theory*, 42(6):1732–1736, Nov 1996.
- [3] E. W. Biersack. Performance evaluation of FEC in ATM networks. In *Proceedings of the ACM SIGCOMM*, Baltimore, MD, August 1992.
- [4] J. Bolot. End-to-end packet delay and loss behavior in the internet. In *Proceedings of the ACM SIGCOMM*, pages 289–298, San Francisco, CA, September 93.
- [5] J. Bolot and A. Vega-Garcia. The case for FEC-based error control for packet audio in the internet. *ACM Multimedia Systems Journal (to appear)*.
- [6] J-C. Bolot and T. Turetletti. Adaptive error control for packet video in the internet. In *Proceedings of International Conference on Internet Protocols*, Lausanne, September 1996.
- [7] G. Carle and E. Biersack. Survey of error recovery techniques for ip-based audio-visual multicast applications. *IEEE Network*, December 1997.
- [8] H. Deng and M. Lin. A type I hybrid ARQ system with adaptive code rates. *IEEE Transactions on Communications*, COM-46(2):733–737, Feb. 1995.
- [9] T. Dorsey. Cu-seeme desktop videoconferencing software. *ConneXions*, 9(4), March 1995.
- [10] R. Fredrick. Network video(nv). Technical report, Xerox Palo Alto Research Center.
- [11] M. Ghanbari and V. Seferidis. Cell-loss concealment in ATM video codecs. *IEEE Transactions on Circuits and Sys. for Video Tech.*, 3(3):238–47, June 1993.
- [12] J. Hagenauer. Rate-compatible punctured convolutional codes (rpc codes) and their applications. *IEEE Transactions on Communications*, 36(4):389–400, April 1988.

- [13] H. Hessenmuller. Video signal transmission in ATM-based broadband network-treatment of cell losses. In *3rd Int. Workshop on Packet Video*, March 1990.
- [14] S. Kallel and D. Haccoun. Generalized type-II hybrid ARQ scheme using punctured convolutional coding. *IEEE Transactions on Communications*, 38(11):1938–1946, November 1990.
- [15] S.K. Kaseta, J. Kurose, and D. Towsley. Scalable, reliable multicast using multiple multicast groups. In *Proceedings of ACM SIGMETRICS*, pages 64–74, Seattle WA, 1997.
- [16] L. Kieu and K. Ngan. Cell-loss concealment techniques for layered video codecs in an ATM network. *IEEE Transactions on Image Processing*, 3(5):666–77, September 1994.
- [17] T. Kinoshita, Nakahashi, and M. Takizawa. Variable bit-rate hdtv coding algorithm for ATM environments in B-ISDN. In *Proceedings of SPIE Conference on Visual Communications and Image Processing: Visual Communication*, pages 604–612, November 1991.
- [18] LBC. Document, LBC-95-309 (ITU-T SG 15, WP 15/1), Sub-videos with retransmission and intra-refreshing in mobile/wireless environments, 1995.
- [19] LBC. Document, LBC-96-033 (ITU-T SG 15, WP 15/1), An error-resilience method based on back channel signalling and FEC, 1996.
- [20] C. Leicher. Hierarchical encoding of MPEG sequences using priority encoding transmission (pet). Technical Report 94-058, ICSI, November 1994.
- [21] X. Li, S. Paul, P. Pancha, and M. Ammar. Layered video multicast with retransmission (lvmr):evaluation of error recovery schemes. In *Proceedings of the Sixth International Workshop on Network and Operating System Support for Digital Audio and Video, St Louis*, May 1997.
- [22] S. Lin and D. Costello. *Error control coding: fundamentals and applications*. Prentice-Hall Inc., Englewood Cliffs, NJ, 1983.
- [23] H. Liu and M. El Zarki. Performance of video transport over wireless networks using hybrid ARQ. In *Proceedings of Proceedings of IEEE International Conference on Universal Personal Communications (ICUPC)*, pages 567–571, Boston, MA, October 1996.
- [24] W. Luo and M. El Zarki. Analysis of error concealment schemes for MPEG-2 video transmission over ATM networks. In *Proceedings of the SPIE/IEEE Visual Communications and Image Processing*, Taiwan, May, 1995.
- [25] J. McAuley. Reliable broadband communications using a burst erasure correcting code. In *Proceedings of the ACM SIGCOMM*, Philadelphia, PA, September 1990.
- [26] S. McCanne and V. Jacobson. vic: a flexible framework for packet video. In *Proceedings of ACM Multimedia'95, San Francisco, CA*, pages 511–522, November 1995.
- [27] Jorg. Nonnenmacher, Ernst Biersack, and Don Towsley. Parity-based loss recovery for reliable multicast transmission. *ACM Transactions on Networking (to appear)*.

- [28] C. Papadopoulos and G. Parulkar. Retransmission-based error control for continuous media applications. In *Proceedings of the Sixth International Workshop on Network and Operating System Support for Digital Audio and Video*, pages 5–12, 1996.
- [29] M. Podolsky. A study of speech/audio coding on packet switched networks. Technical report, MS Thesis, Dept. of Electrical Eng. and Computer Sciences, Univ of California, Berkeley, December 1996.
- [30] M. Podolsky, C. Romer, and S. McCanne. Simulation of FEC-based error control for packet audio on the internet. In *Proceedings of the ACM SIGMETRIC/PERFORMANCE*, June 1998.
- [31] G. Ramamurthy and D. Raychaudhuri. Performance of packet video with combined error recovery and concealment. In *Proceedings of the IEEE INFOCOM*, pages 753–761, April 1995.
- [32] I. Rhee. Error control techniques for interactive low-bit rate video transmission over the internet. In *Proceeding of the ACM SIGCOMM'98 (to appear)*, Vancouver, Canada, Sept. 1998.
- [33] I. Rhee. Retransmission-based error control for interactive video applications over the internet. In *Proceedings of International Conference on Multimedia Computing and Systems*, pages 118–127, Texas, Austin, June 1998.
- [34] L. Rizzo. Effective erasure codes for reliable computer communication protocols. *Computer Communication Review*, 27(2):24–36, Apr 1997a.
- [35] Dan Rubenstein, Jim Kurose, and Don Towsley. Real-time reliable multicast using proactive forward error correction. *NOSSDAV 98 (to appear)*.
- [36] R. Storn. Modeling and optimization of pet-redundancy assignment for MPEG sequences. Technical Report TR-95-018, ICSI, Berkeley, CA, May 1995.
- [37] R. Talluri. Error-resilient video coding in ISO MPEG-4 standard. *IEEE Communication Mag.*, 36(6):112–119, June 1998.
- [38] Priority Encoding Transmission. Web page: <http://www.icsi.berkeley.edu/pet/icsi-pet.html>.
- [39] R. Xu, C. Myers, H. Zhang, and R. Yavatkar. Resilient multicast support for continuous-media applications. In *Proceedings of the Sixth International Workshop on Network and Operating System Support for Digital Audio and Video*, St. Louis, May 1997.
- [40] J. Zdepski and H. Sun. Error concealment strategy for picture-header loss in MPEG compressed video. In *Proceedings of High-Speed Networking and Multimedia Computing*, pages 145–152, San Jose, CA, Feb 1994.
- [41] M. Zorzi and R. Rao. Capture and retransmission control in mobile radio. *IEEE Journal on Selected Areas in Communications*, 12(8):1289–1298, 1994.