Video Redundancy – A Best-Effort Solution to Network Data Loss

by

Yanlin Liu

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

in partial fulfillment of the requirements of the

Degree of Master of Science

In

Computer Science

by

____

May 1999

APPROVED

____

Prof. Mark Claypool

____

Prof. Micha Hofri, Head of Department

Abstract

With rapid progress in both computers and networks, real-time multimedia applications are now possible on the Internet. Since the Internet was designed to support traditional applications, multimedia applications on the Internet often suffer from unacceptable delay, jitter and data loss. Among these, data loss has the largest impact on quality. Current techniques that correct packet loss often result in unacceptable delays. In this thesis, we propose a new forward error correction technique for video that compensates for lost packets, while maintaining minimal delay. Our approach transmits a small, low-quality redundant frame after each full-quality primary frame. In the event the primary frame is lost, we display the low-quality frame, rather than display the previous frame or retransmit the primary frame. To evaluate our approach, we simulated the effect of data loss over network and repair the data loss by using the redundancy frame. We conducted user studies to experimentally measure users' opinions on the quality of video streams in the presence of data loss, both with and without our redundancy approach. In addition we analyzed the system overhead incurred by the redundancy. Result of the user study shows that video redundancy can greatly improve the perceptual quality of transmitted video stream in the presence of data loss. The system overhead that redundancy introduces is dependent on the quality of the redundant frames, but a typical redundancy overhead will be approximately 10% that of primary frames alone.

Acknowledgements

I would like to express my gratitude to my advisor Prof. Mark Claypool for his continuous support and help on all aspects from technical to material. His guidance and advice helped me overcome many technical obstacles that would take much more efforts.

My thanks are also to my reader Prof. Craig E. Wills for his valuable advice on my research.

Thanks also to all my friends and fellow graduate students particularly Mikhail Mikhailov, Kannan Gangadharan, Helgo Ohlenbusch for their kind support with my user study.

The thesis is dedicated to my parents. It is their love, encouragement, belief and understanding that made everything I have possible.

Contents

List of Figures

Chapter 1: Introduction

Emerging new technologies in real-time operating systems and network protocols along with the explosive growth of the Internet provide great opportunity for distributed multimedia applications, such as video conferencing, shared “whiteboards,” and media-on-demand services. Multimedia is engaging, entertaining, makes the computer friendlier and attracts more users. The introduction of multimedia to the Internet can also increase productivity since more information can be shown visually.

Since the Internet is packet routed, video frames go through different routes to reach the receiver. It is possible that some frames arrive at the receiver when the time they should be displayed has passed. In some cases, some frames are lost during network transmission. In order to recover from the data loss, retransmission can be used, but waiting for the retransmitted data can also incur added delay.

Traditional applications, such as FTP, which have no strict timing or end-to-end delay constraints, emphasize the accuracy of the transmitted contents and use retransmission to ensure quality. Multimedia applications have different requirements. With current technology, multimedia data transmission often suffers from three types of network problems: delay, data loss and jitter. Although today’s network and high-speed computers are increasingly fast, data loss is still common on the Internet. Unlike in traditional applications, a certain range of imperfection can often be tolerated in a multimedia stream. A small gap in a video stream may not impair the perceptual quality as much, and may not even noticeable to users.

Data loss is a common problem in today’s Internet. Network congestion and buffer overflow can all result in data loss, which results in a gap in the continuous data stream.

Data loss in multimedia data transmission can impact the continuity in the display. Data loss can occur involuntarily from network congestion or system buffer overflows, or voluntarily in order to avoid congestion at a client, server or network router. Audio conferences on the Mbone have reported data loss rates as high as 40% [Ha97]. Too much data loss can result in unacceptable media quality.

To compensate for data loss, much work has been done to find effective data-loss recovery techniques. There are two categories of data loss recovery techniques: sender-driven and receiver-based [PHH98]. Each of them has its own strength and weakness. These techniques have proven to be effective for audio stream data loss, but have yet to be applied to video. Most of the previous work in data loss recovery for video has focused on the media scalability, which proposes to transmit several versions of the same frame on different quality levels, and retransmission. However most existing media scaling techniques have special limitations, such as network requirements. Retransmission can serve for all types of networks, but it is not appropriate for some multimedia applications with which only short end-to-end delay can be accepted.

In this thesis, we apply an existing forward error correction technique used for audio and propose a means to piggy-back low bandwidth redundancy to the video stream at the sender. Unlike typical media scaling techniques, where the secondary frame is not useful unless the primary frame exists, the redundant frame we propose can be used alone. When the primary frame is received correctly, the redundancy is not useful and should be discarded. The redundancy needs to be retrieved and decoded only when the primary frame is lost.

Most video frames are compressed before being sent from sender to receiver. One popular standard used for video compression is MPEG[MP96]. MPEG uses lossy compression (some of the original image data is lost during encoding), by adjusting the quality and/or compression rate at encoding time. The higher the quality, the lower the compression rate, and vice versa. We use MPEG variable quality encoding to encode the original video frames into two versions, one with high quality and one with low quality. The high quality frames are sent as primary frames, while the low quality ones are considered secondary frames and piggy-backed with the next primary frame. If the primary frame is received correctly, the secondary frame is discarded without being decoded. When the primary frame is lost and the next packet arrives correctly, the secondary frame will be extracted and decoded to take the place of the lost one.

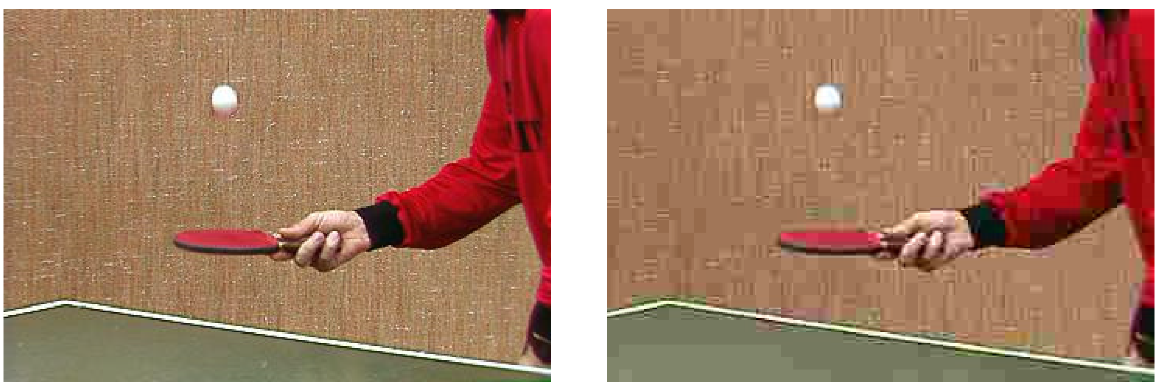

Figure 1.1: Two Frames with Different Compression Rates

Both of the two frames are compressed from the same original frame. The left one is compressed with high quality, but has a low compression rate. The size of this frame is 19K bytes. The right frame is compressed with low quality, but has a high compression rate. The size of this frame is 3K.

To evaluate our approach, we first examine the effects our technique has on Perceptual Quality (PQ). PQ is a measure of the performance of multimedia from the user's perspective. We simulated several different patterns of data losses and generated repaired video streams according to the loss and performed a user study with these streams. Since the redundancy added to the video stream needs extra processing time and network bandwidth, which may in turn affects the network transmission and end-to-end delay, we analyze the system overhead. In the following chapters, we describe the user study results, the system overhead and draw our conclusions.

The contributions of this thesis may be summarized as follows:

- A method for video data loss recovery by piggy-backing redundant frames to primary frames.
- User studies investigating the perceptual quality of this method.
- Analysis of the overhead redundancy adds to the system.
- A method for applying our redundancy technique to MPEG.
- A method for simulating loss and redundancy in MPEG video files
- A framework for conducting perceptual quality user studies.
- An analysis of typical loss percents and consecutive loss frequency in Internet multimedia transmission.

The remainder of the thesis is outlined as follows. In Chapter 2, we discuss related work. In Chapter 3, we propose our approach to the problem of packet loss, describe the simulation for testing the PQ, and discuss the user study results. In Chapter 4, we analyze the system overhead of the redundancy. In Chapter 5, we draw our conclusions and make suggestions of where to apply this method. In Chapter 6, we briefly discuss possible future work.

Chapter 2: Related Work

The goal of this chapter is to give the reader some fundamental concepts to better understand this work. Discussions in this chapter are directly related to this study and are dealt with in some detail. The topics include audio loss repair, video loss repair, data loss patterns, MPEG encoding, and multicast performance.

2.1. Audio Loss Repair

Most video frames are larger then audio frames, but since audio has similar real-time requirements as video, we build our work upon past research in audio over the Internet. There are two types of possible audio repair techniques: *sender-based* and *receiver-based*. Sender-based repair techniques require the addition of repair data from the sender to recover from the loss. Receiver-base repair techniques rely only on the information correctly received.

- **Sender Based Repair**

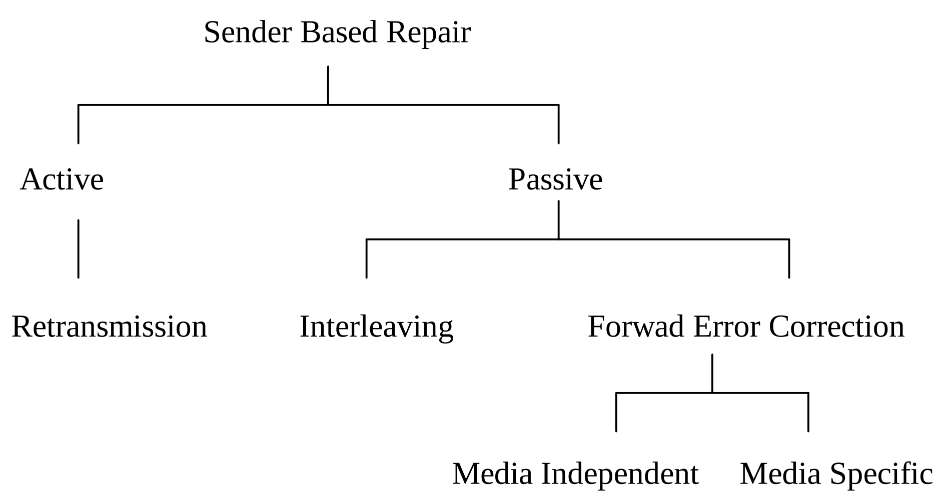

Figure 2.1 A Taxonomy of Sender-Based Repair Techniques

As indicated in Figure 2.1, sender based repair techniques can be split into two categories: *passive channel coding* and *active retransmission*. With passive channel coding, the sender sends the repair data. The sender is not informed whether or not the loss is repaired or not. If it is not, the sender will have no further intention to repair it. With active retransmission, if there is still time for repairing, the sender will be informed of the loss and required to assist in recovering from the loss.

Passive channel coding techniques include forward error correction (FEC) and interleaving-based schemes [PHH98].

1. **Forward Error Correction (FEC)**

Many forward error correction techniques have been developed to repair audio loss. These schemes rely on the addition of repair data (redundancy) to the data stream, from which the contents of the lost packets can be recovered. The repair data added to the stream can be either independent of the contents of that stream or those using the knowledge of the stream.

a) Media Independent FEC:

Most of the media independent FEC techniques use block, or algebraic, codes to produce additional packets for transmission to add the correction of losses. For the transmission of *n* packets, *k* additional packets will be generated for *n-k* original data packets.

One popular media independent FEC is *parity coding* [PHH98]. In this scheme, *1* parity packet is generated and transmitted after every *n-1* original data packets that are transmitted. The *i*'th bit in the parity packet is generated from the *i*'th bit of each of the associated data packets by applying the exclusive-or (XOR) operation across groups of packets. If only one the *n* packets is lost, the parity packet can be used to generate an exact replacement of the lost one. Figure 2.2 shows how parity coding works.

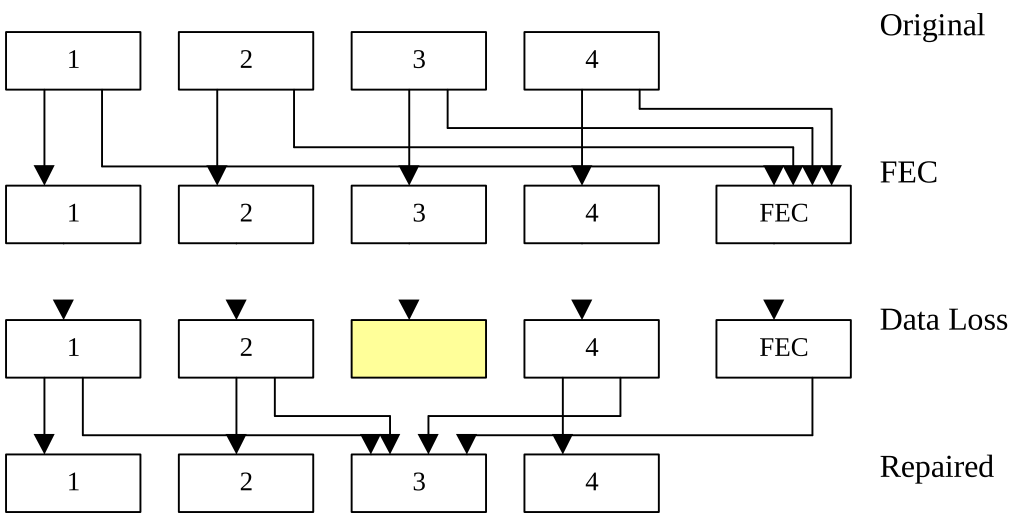

Figure 2.2 Repair Using Parity FEC

Media independent FEC does not need the knowledge of the media content and the repaired data is the exact replacement of the lost packet. Also the algorithm is simple and easy to implement. Unfortunately, it introduces additional delay and bandwidth.

b) Media Specific FEC:

A simple way to recover from data loss is just to transmit the same unit of audio in multiple packets [PHH98]. If a packet is lost, some other packet with the same unit can be used to recover the loss. The first transmitted copy is usually referred to as the primary encoding and subsequent transmission as the secondary encoding(s). The sender can decide whether the secondary encoding should be the same as the primary encoding or whether to use a lower-bandwidth, a lower quality encoding than the primary. Figure 2.3 illustrates this scheme.

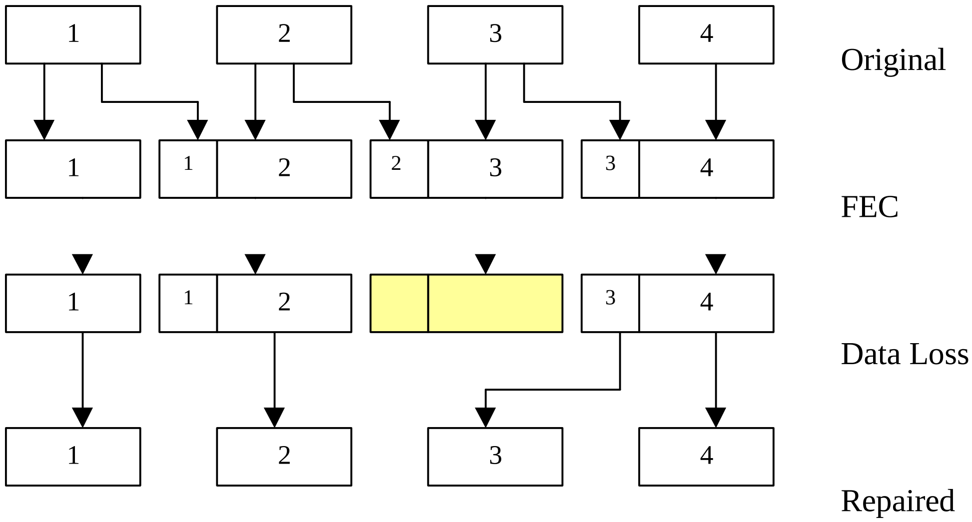

Figure 2.3 Repair Using Media Specific FEC

The use of media specific FEC incurs an overhead in terms of packet size. Like media independent FEC, the overhead is variable. It can be reduced without affecting the number of lost packets it can repair, but instead varies the quality of the repair with the size of the overhead.

2. **Interleaving**

Interleaving attempts to reduce the effect of the loss by spreading it out. Units are resequenced before transmission, so that originally adjacent units are separated into different packets. At the receiver side, units are returned to their original order. If one packet is lost during the transmission, instead of having a big hole in the stream, the loss is separated into several small holes which are meant to be easier to mentally ignore. Figure 2.4 illustrates this scheme.

The advantage of this scheme is that it does not introduce overhead to the data stream, but it increases latency. This limits the use of this technique for interactive applications which are delay sensitive. Interleaving-based repair can be used when the unit size is smaller than the packet size and the end-to-end delay is unimportant [PHH98].

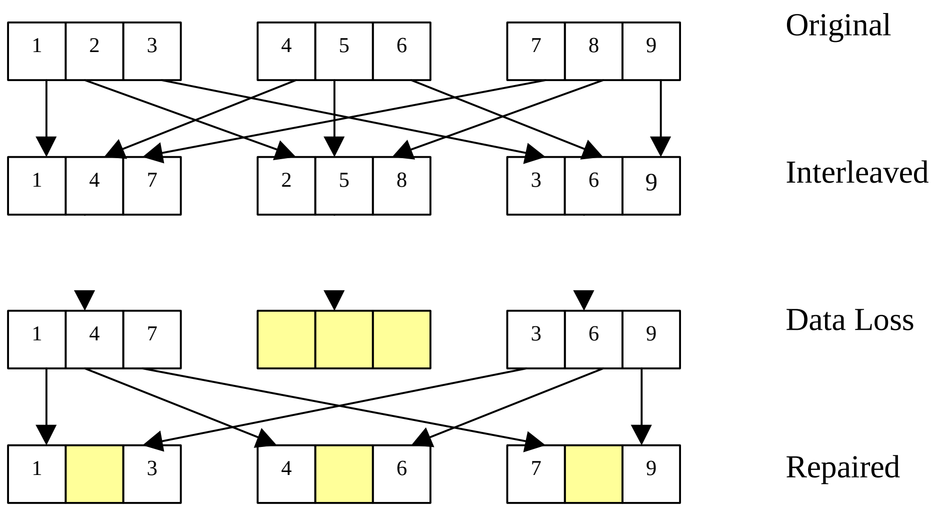

Figure 2.4 Interleaving Units Across Multiple Packets

Active Retransmission techniques can be used when larger end-to-end delay can be tolerated. A widely deployed reliable multicast scheme based on the retransmission of lost packets is Scalable Reliable Multicast (SRM) [PHH98]. When a receiver of a SRM session detects a loss, it will wait a random amount of time determined by its distance from the sender and then multicast a retransmission request. The timer is calculated such that, although a number of hosts may miss the same packet, the host that is closest to the failure will most likely timeout first and issue the request. Other hosts that miss the same packet but received the retransmission request will suppress their own requests to avoid message implosion. On receiving the retransmission request, any host with the requested data may reply. Once again, this host will wait for some time determined by its distance from the sender of the request to avoid reply implosion. With this scheme, typically only one request and one reply will occur for each loss.

- **Receiver Based Repair**

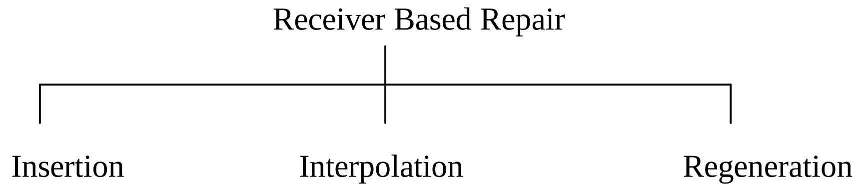

Figure 2.5 Taxonomy of Error Concealment Techniques

Receiver based repair techniques are also called Error Concealment. These techniques can be initiated by the receiver of an audio stream without the assistance of the sender. If the sender based repair schemes fails to recover all loss, or when the sender is unable to participate in the recovery, these techniques can be used. Error concealment techniques rely on making the loss of the packet less noticeable to the user. As shown in Figure 2.5, there are three kinds of receiver based data loss repair techniques: *insertion based, interpolation based,* and *regeneration based* schemes.

1. **Insertion-Based Repair**

   Insertion based repair schemes derive a replacement for a lost packet by inserting a simple fill-in [PHH98]. The characteristics of the signal are not used for generating the fill-ins.

   Splicing : Lost packets are ignored and the audio on either side of the loss is spliced together. No gap remains because of the missing packet, but the timing of the stream is impaired. Moreover, it is difficult to reorder the packets that arrived in a wrong sequence.

   Silence Substitution : Silence substitution fills the gap left by missing packets with silence in order to keep the timing relationship between surrounding packets.

Noise Substitution : Noise substitution fills the gap with background noise. Studies have shown that it is easier for humans to mentally patch-over gaps by filling it with noise rather than plain silence.

Repetition : Repetition replaces the lost units by repeating the unit received immediately before the lost one. It has low computational complexity and performs reasonably well.

2. **Interpolation-Based Repair**

   Some error concealment techniques exist try to interpolate from packets surrounding the loss to produce a replacement by using the changing characteristics of the signal. These techniques include waveform substitution, pitch waveform replication, and time scale modification. They are more complex compared to insertion based repair techniques.

3. **Regeneration-Based Repair**

   They use the knowledge of the audio compression algorithm to derive codec parameters, such that audio in a lost packet can be synthesized. Interpolation of transmitted state technique and model-based recovery techniques belong to this category. These techniques are even more complex then interpolation based repair.

Some of these techniques use the knowledge of audio compression characteristics, and are specific for audio use, while other techniques are more general, they can be applied to a broader area, such as video. Our approach combines media specific FEC and repetition repaired error concealment. A lost packet is replaced by the redundancy transmitted within the next packet. When the redundancy fails to repair the lost packet, a repetition based error concealment technique is used to fill the gap left. Figure 2.6 shows how our proposed scheme works.

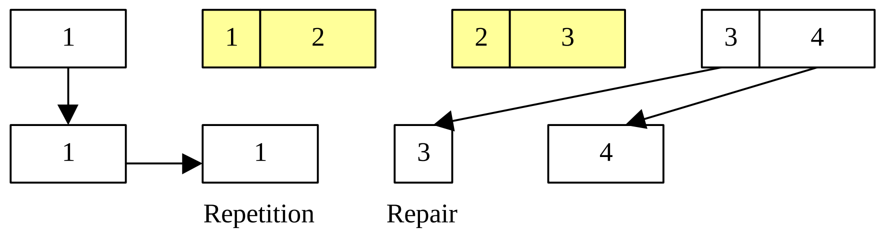

Figure 2.6 Our Approach – Combine Media Specific FEC and Packet Repetition

2.2. Video Loss Repair

Research in video data transmission over a network proposes to reduce the data loss by controlling the network congestion, or to provide a way to recover lost video frames.

Hemant Kanakia, et al. dynamically change the video quality level during network congestion [KMR93]. They propose a mechanism to study the performance of an overload control strategy that uses feedback from the network to modulate the source rate. During periods of congestion it can reduce the input rate from video sources substantially with a very graceful degradation in the image quality. Their mechanism does not focus on repairing the lost packet, rather it prevents future data loss by dealing with network congestion.

The research by Steven Gringeri et al is based on the ATM network which can provide higher speed and better services than traditional networks [GKL+98]. Since the ATM cells are fixed size (53 bytes) and allow multiplexing of various services such as voice, video and data with guaranteed cell rate, cell loss and cell delay variation parameters, it makes ATM cells suitable for real-time video applications. To deal with network data loss, a method is proposed to use hierarchical coding and scalable syntax.

Hierarchical coding allows reconstruction of useful video from pieces of the total bit stream. The MPEG standard specifies scalable syntax to support this process. Scalability is achieved by structuring the total bit stream into two or more layers starting with a stand-alone base layer and adding a number of enhancement layers. When video streams are transmitted through network, each layer has a different QoS. The base layer is transmitted with higher priority to ensure low cell loss, while the enhancement layers can be transmitted with lower priority. Within the ATM network, a channel with guaranteed QoS requirements is assigned to transmit the base layer to preserve its integrity. A less reliable channel can be used to transmit the enhancement layer(s). At the receiver side, the base layer data and enhancement layer data is combined to produce the original video stream. If errors occur in the enhancement layer, the video still can be reconstructed using only the base layer.

This technique can ensure the base quality level of video transmission, but it takes the advantage of ATM network. Many traditional, low-speed, low-bandwidth, best-effort networks are still in use throughout the world. Most of them cannot guarantee the quality of service, nor provide different channels with different priorities as ATM does. We seek to improve the quality of video streams with the existence of data loss on the widespread, traditional networks.

Some work has been done for distributing MPEG-encoded video over a best-effort heterogeneous network, such as the Internet, which does not have any support for QoS guarantees. A protocol called Layered Video Multicast with Retransmission is designed and developed by Xue Li *et al* to deal with data loss through error-prone networks [LPP+97]. The idea is to use a layered video coding approach. Layered multicasts provide a finer granularity of control compared to using a single video stream. A receiver can subscribe to one, two or more layers depending upon its capability. In [LPP+97], they propose to break the MPEG frames into three layers. The base layer includes only I-frames. The first enhancement layer includes P frames and the second enhancement layer includes B frames. The receivers will periodically generate an acknowledgement (ACK) which includes a sequence number and a bitmap to indicate what data packages it has correctly received. To prevent ACK implosion at sender's side, this scheme uses hierarchies of *Designated Receivers* (DRs) to assist the sender in processing ACKs and in retransmitting data. A DR is a special receiver, which caches received data, emits ACKs and processes ACKs [LPP+97].

Since there are strict end-to-end delay requirements for real-time video, it may be not useful to retransmit lost frames if they can not arrive at the receiver side before it has to be played. Xue Li *et al* propose a Smart Reliable Multicast Transport Protocol (SRMTP) to solve this problem. Before a retransmission is sent out, an algorithm is used to estimate whether there is enough time for this retransmission. If p_n denotes the time that the frame to be displayed to the user, t_n denotes the arrival time of the frame, Δ denotes the maximum jitter in the network, and T denotes the inter-frame time. Then $p_n = t_0 + \Delta + nT$. Here, $min\ (p_n - t_n) = 0$. In SRMTP, a *control time*, δ, is defined as the duration between the arrival instant and playback point of the first frame. The introduction of δ allows more time for retransmission. The equation now becomes $p_n = t_0 + \Delta + \delta + nT$. $min\ (p_n - t_n) = \delta$. A retransmission is effective when the retransmitted packet arrives before the playback point ($\delta > t_l + rtt + t_r$, t_l denotes loss detection time, rtt denotes the round trip time and t_r denotes retransmission processing time). When the application control multiplexes one or more substreams, the playback point can be adaptive. The *adaptive playback point* for frame n is defined to be $p_n' = p_n + kT$, where kT is the time interval between the current. For the frame pattern IBBPBBPBB, if a receiver subscribe to all three layers, $k = 1$, and $p_n' = p_n + T$. $min\ (p_n' - t_n) = \delta + T$. If the receiver drops the second enhancement layer, k becomes 3, then $min\ (p_n' - t_n) = \delta + 3T$. If the first enhancement is also dropped, $min\ (p_n' - t_n) = \delta + 9T$. During network congestion, playback points are transparently moved back and there is more time to recover from the lost packet by retransmission.

This technique uses active retransmission to recover from packet loss. It is suitable for applications with no critical end-to-end delay requirements. When only little delay is tolerable, most of the losses may not be recovered since there is not enough time for retransmission.

2.3. MPEG-1 Encoding

Since video data are usually too large for raw transmission or storage, most video streams are compressed. MPEG (Motion Picture Expert Group) is one of the popular standards used today [MP96]. MPEG strives for a data stream compression rate of about 1.2 Mbits/second. It delivers at a rate of at most 1.85 Mbits/second. MPEG is suitable for symmetric as well as asymmetric compression, where compression is carried out once, and decompression is performed many times.

MPEG compression method is lossy, which means to achieve a higher compression rate, some information in the original image may be lost during the compression and cannot be recovered when decoded. Thus, the compressed video streams may have lower quality than the original ones. The higher the compression rate, the lower the size of the frame, and vice versa.

To achieve a high compression rate, temporal redundancies of subsequent pictures must be exploited (inter-frame). MPEG distinguishes four frame types of image coding for processing: I-frame, P-frame, B-frame, and D-frame. Different coding types have different compression rates. To support fast random access, intra-frame coding is required. In the following, we discuss these four types of coding separately:

- I-frame (Intra-coded images). Frames of this kind are self-contained. They are compressed without any reference to other images. MPEG make use of JPEG [MP96] for the I-frames. I-frames can be treated as still images and are used for random access. The compression rate of the I-frames is the lowest within MPEG.
- P-frame (Predictive-coded frames). The encoding and decoding of P-frames requires the information of previous I frames and/or all previous P-frames. In many successive video images, the context does not change significantly. Rather, the view may be shifted when the camera pans. Based on this fact, the *temporal redundancy*, the block of the I- or P-frame that is most similar to the block under consideration, is determined. Compression rates for P frames are higher than I-frames.
- B-frame (Bi-directionally predictive-coded frames). The encoding and decoding of B-frames requires the information of the previous and following I- and/or P-frame. A B-frame is defined as the difference of a prediction of the past image and the following P- or I-frame. The highest compression rate can be attained by using these frames.

- D-frame (DC-coded frames). These frames are intra-frame encoded. They can be used for fast forward or fast rewind. D-frames consist only of the lowest frequency of an image.

Most MPEG video streams, contains only I-, P-, and B-frames. Their dependency relationship is illustrated in Figure 2.7. The encoding pattern of this stream is IBBPBBPBB, where the last two B-frames depend on both the second P-frame and the next I-frame.

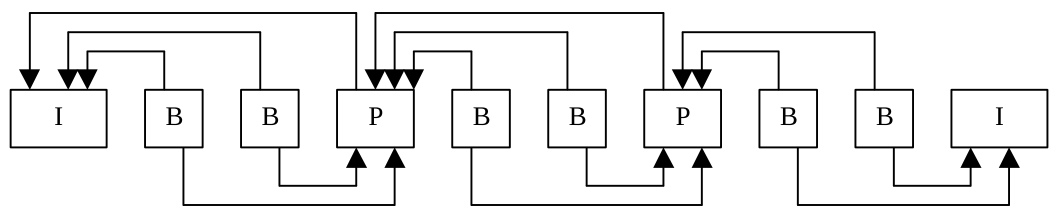

Figure 2.7.a. MPEG Frame Dependency Relationship

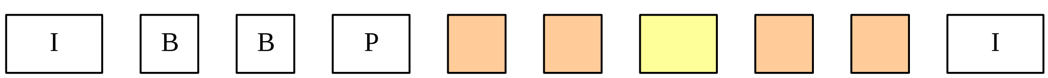

Figure 2.7.b. The Loss of second P-Frame.

Shown in Figure 2.7, a P-frame depends on the previous I- or P-frame. A B-frames depends on the previous and following I- or P-frame. The loss of one P-frame can make some other P- and B-frames useless, while the loss of one I-frame can result in the loss a sequence of frames. In MPEG encoded video streams, I-frames and P-frames are more important than B-frames.

2.4. Multicast Performance

In many applications, such as videoconferencing, multimedia data are multicast to more than one receiver. Before addressing our approach, we need to have a clearer idea of multicast performance.

A thorough examination of Mbone multicast performance is presented in [Ha97]. Mark Handley examined the routing tables to monitor route stability, and observed traffic as it arrived at sites to which they could have access to look at individual packet losses. The loss rate was calculated by dividing the packets received by the packets expected in that interval. It is possible that the loss reported may occur in the end-system rather than the network. However since the traffic measured constitutes a relatively low frame-rate video stream, it is unlikely that this is a significant source of loss. The research shows that 50% of receivers have a mean loss rate of about 10% or lower, while 80% reported loss rate less then 20%. Around 80% of receivers have some interval during the day when no loss was observed. On the other hand, 80% of sites reported some interval during the way when the loss rate was greater than 20%, which is generally regarded as being the threshold above which audio without redundancy becomes unintelligible. About 30% of sites reported at least one interval where the loss rate was above 95% at some time during the day. Research also shows that packet losses are not independent, but occur in long, bursts than would be the case if they were independent. Yet, the excess of bursts of 2-5 packet losses compared with what could be expected from random loss, although statistically significant, is not significant to greatly influent the design of most applications. Single packet losses still dominate. They concluded that for a large session with many receivers, it is most probably that each packet will be lost by at least one receiver. To rely on retransmission for data loss repair, the majority of the packets will be NACKed and retransmitted at least once. If the retransmitted data is sent to all receivers, there will be a retransmission implosion and more network bandwidth will be consumed making the existed congestion even worse. Even when there is no high loss rate receivers in the multicast group. The evaluation results indicate that packet-level or ADU-level FEC techniques should be considered by the designers of any reliable multicast protocol. The additional traffic for FECs serves to fix many *different* small losses at each different site.

In our research, we build our redundancy approach upon the existing audio loss repair techniques and try to repair video data loss with lower delay compared to retransmission. We use the MPEG encoding features and propose to compress original images into two versions with different compression rate (quality). High quality is transmitted as primary frames and low quality version as secondary frames. With the knowledge of multicast data loss patterns, we simulate the effect of our repair method and conduct a user study to experimentally evaluate how effective can redundancy improve the perceptual quality in the presence of data loss. In the next Chapter, we present our approach and discuss the user study result in detail.

2.5 Perceptual Quality

The strict study in data loss and end-to-end delay measures and assesses the quality of multimedia services at the network level. Perceptual Quality (PQ) is the subjective quality of multimedia perceived by the user [WS98].

The users' expectation from a multimedia data transmission is that the Quality of Service (QoS) with which they are shown can enable the users to assimilate and understand the informational contents of such clips. Therefore, Perceptual Quality is the end-user measurement for determining whether a multimedia transmission is successful.

In investigating a user's perception of a video transmission, the influence of many variables needs to be considered, such as color, brightness, clearness, background stability, frame rates, delay and speed in image reassembling. With current technologies, it is often the case that the trade-off for improving the quality in one respect is to decrease the quality in other respect. For instance, in order to ensure the image clearness of video, retransmission can be used, which will potentially affect the delay in the display. Within our method, we seek to ensure a short end-to-end delay in the presence of data loss with the trade-off to be the degradation of the clearness of some images.

Many methods have been proposed to measure Perceptual Quality. One of them is the standard recommended by the International Telecommunications Union (ITU) [WS98]. They propose a five-scale measurement to assess the quality of video. Figure 2.8 shows the standard of recommended by ITU.

Image Quality	*Score*
Excellent	5
Good	4
Fair	3
Poor	2
Bad	1

Figure 2.8 Image Quality Scale

However, this standard provides no international interval, nor does it have international ordinal. It is not a strictly legitimate assessment. New approaches must be found to effectively measure the perceptual quality. A slider mechanism labeled with the

Dutch quality scales term was proposed by de Ridder & Hamberg [RH97]. The observers manipulated this slider as they watched video sequences, and the results showed that they were able to monitor video quality variations as they occurred.

In our research, we evaluate our redundancy method by measuring users' Perceptual Quality by building upon the work of past researches.

Chapter 3: Perceptual Quality

In this chapter, we explain the redundancy based repair technique in detail. We simulate the effects of our technique on MPEG video streams in the presence of packet loss by building movies that repeat frames if there is no redundancy and use a low quality frame when using redundancy. We use these streams in a user study. In which we gather the opinion of the users, and draw conclusion on whether this technique can practically improve the perceptual quality of the video streams with loss.

3.1 Our approach

In the presence of data loss, without the redundancy, lost frames cannot be repaired. We use a repetition technique to compensate for the loss by playing the frame that is received immediately before the lost one again. If the lost frame is an important frame, such as an I frame or a P frame, the subsequent frames may be lost as well since they are dependent upon the lost one. By playing the previous frame again and again, the perceptual quality of the video may decrease. The end users may notice some sudden stop during the display, as screen seems momentarily frozen and followed by a big jump from one scene to a totally different one.

To solve this problem, we propose a method to include redundancy for video repair in the presence of packet loss into the video stream during the network transmission. As indicated in the discussion of MPEG in Chapter 2, the compression rate and the quality of the compressed video stream can be controlled by the encoder. The quality of these videos can scale from sharp and clear to fussy and undistinguishable, resulting in large and small frame sizes, respectively.

Before transmission, the encoder generates the two versions of compressed frames, one with high quality and a low compression rate, the other with low quality and a high compression rate. The high quality frames will be considered *primary frames*. In this paper, we refer to them as H_i. The low quality frames will be considered *secondary frames*. We refer to them as L_i. For each frame *i*, H_i will be transmitted first. L_i will be piggy-backed with H_{i+1}. At the receiver side, if H_i is received successfully, it will be played to the end user directly and L_i will be discarded upon its arrival. If, unfortunately, H_i is lost or totally corrupted during the transmission, the decoder will wait for the next packet. L_i will be extracted and take the place of the lost (or corrupted) H_i. Figure 3.1 shows how our redundancy scheme may be incorporated into a video server.

With redundancy, in a network where bursty loss exists, the secondary frame might also not be able to reach the receiver. In such a case, not all the losses can be repaired. If neither H_i nor L_i managed to survive the network transmission, we use repetition. Although the redundancy can make the video look better, sudden stops and abrupt jumps may still exist in the presence of heavy loss. Part of our user study examines to what extent consecutive frame loss has the effect on repaired video streams.

3.2 Simulation

In this section, we describe in detail the methodology we used to build movies that simulate lost frames.

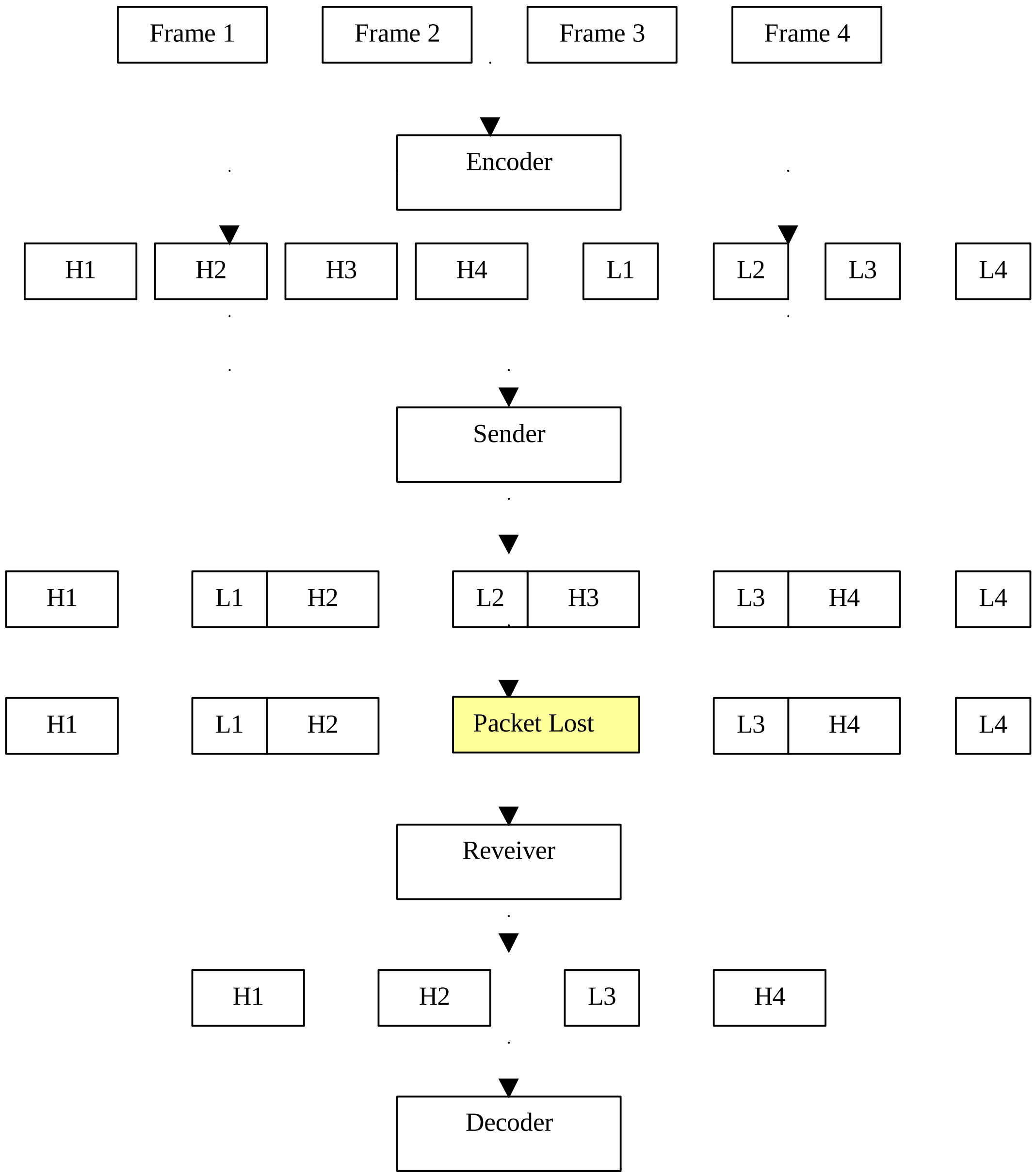

Figure 3.1. Video Redundancy Architecture

In this Figure, each box represents a frame. The ones with H*i* represent high quality frames and the ones with L*i* represent low quality frames. Each low quality frame is piggy-backed with a high quality frame during the transmission.

There are two approaches to measuring user Perceptual Quality – in real field trials, and in controlled experimental conditions which mimic aspects of the real world situation. Although field trails are more desirable in that they actually *are* what a user would expecting, they are costly and time-consuming, can be frustrating for the user and do not always provide the means for acquiring the information that is required by the human factors investigator [WS97]. In our research, we chose the second approach. We simulated the network data loss and tried to repair the loss by using redundancy or repetition.

Original high quality MPEG files are broken into images and compressed into high quality frames and low quality frames. If redundancy is not used, lost frames are repaired by repeating the previous frame. If redundancy is used, lost frames are replaced by the low quality ones.

The encoding tool we used is Berkeley MPEG-1 Video Encoder. It contains the following tools that we used for this simulation: `mpeg_encode`, and `ppmtoeyuv`. The decoding tools we used are Berkeley MPEG-2 player [BM2] and the Microsoft Media Player [MMP].

We wrote a Perl script to automate building the streams.

- First we break the original .mpg file into separate .ppm files, one file for each frame in the video stream. Since images with EYUV format can be accepted by the MPEG encoder as original files and the size of EYUV file is much smaller than the .ppm file, we convert the each .ppm file into a .yuv file (EYUV format).
- Then we adjust the frame rate from 30 fps to 5 fps. Since the encoder can accept frame rate no less then 24 fps, and the normal frame rate through a WAN is at most 5 fps, we simulate the 5 fps by duplicating the frames in the video stream and dropping others. Thus in our simulation, the frame rate was set to be 30fps with the duplicate rate 6, which means each frame in the frame is played 6 times and only 5 different

frames are played within one second. For example, in an original 30fps MPEG file, the first 12 frames are:

```
F0  F1  F2  F3  F4  F5  F6  F7  F8  F9  F10 F11
```

In our simulated stream, the frames become:

```
F0  F0  F0  F0  F0  F0  F6  F6  F6  F6  F6  F6
```

Although the real stream is still 30 fps, the effect to the user is the same as 5 fps.

- Next we adjust the IPB pattern. In this simulation we used the common IPB pattern for mpeg files: IBBPBBPBB.
- Then, we adjust the loss rate.   In order to realistically simulate packet loss, we relied upon work by Gerek and Buchanan [GBC98]. They gathered the data of 102 network data transmissions over the Internet across the USA and New Zealand [GBC98]. UDP was the protocol used for the experiment. Each of these transmissions was a 200-second trace. The contents transmitted included MPEG video data with different IPB pattern (only I-frames, or only I- and P-frames, or I-, P-, and B-frames) and audio (CBR voice or VBR voice). Figure 3.2 shows the loss rate distribution and Figure 3.3 shows the distribution of consecutive loss numbers.

From Figure 3.2 we can see that 50 of these transmissions got a loss rate greater than 20%. Of those who got a loss rate less than or equals 20%, most of them are within the range between 0% and 5%. For a transmission with a loss rate greater than 20%, the quality is bound to suffer with all kinds of repair techniques and most users will simply give up. Also with a very high loss rate, users tend to have difficulty distinguishing really bad quality from poor quality. So we focused our attention to the part where repair techniques can efficiently improve the video quality.

From these results we concluded that for low loss rates (0% to 10%), most loss is of single consecutive packet. As you can see from Figure 3.3 that the total number of consecutive loss is much less than that of single loss.

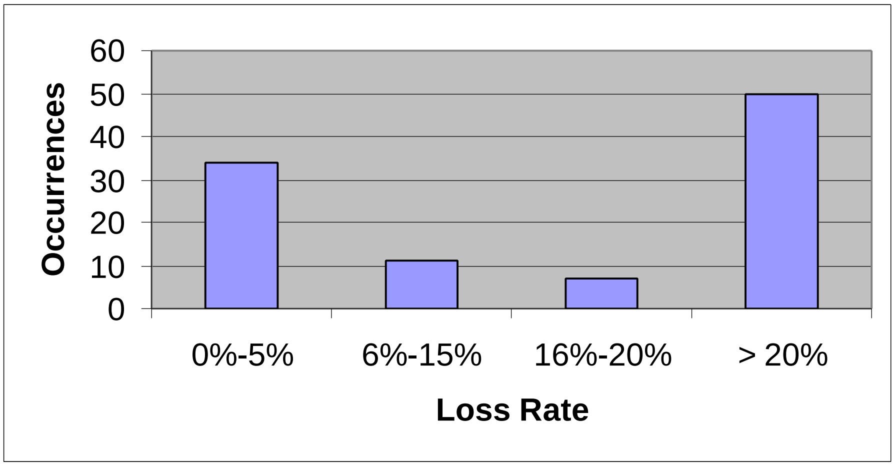

Figure 3.2 Loss Rate Distribution

In this Figure, x-axis represents the loss rate. Four ranges are examined. The y-axis represents the number of occurrences within these 102 network transmissions.

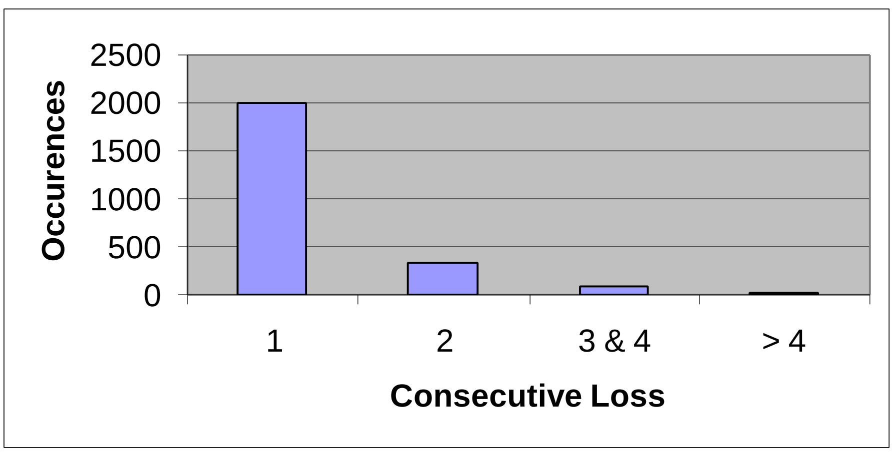

Figure 3.3 Consecutive Loss Distribution

In this Figure, x-axis represents the consecutive loss pattern. Four cases are examined. The y-axis represents the number of occurrences within these 102 network transmissions.

Thus, in our experiment, we choose 3 loss rates for examination: 1%, 10%, and 20%, which we call the *raw* loss rate. For example, if 10 out of 100 frames are lost through the network, the *raw* loss rate is 10%. Some of the lost frames may be I frames or P frames. The loss of this kind of frame can leave the frames that are dependant on it useless, which results in a even higher loss rate to the end user.

- Lastly, we adjust the consecutive loss parameter. In some circumstances, the network can introduce bursty loss to the video stream, with 2 or more consecutive lost frames. Most of the consecutive losses are from the transmission with loss rates greater than 10% (not shown in these graphs). However, Figure 3.3 shows that 4+ packet consecutive loss do occur. In this case, both the primary and redundancy frames will be lost. Therefore, some frame loss can be repaired while some others cannot. We include this parameter to study how much the bursty feature of packet loss can affect the repair result. Three different numbers are used for this study: 1, 2 and 4.

Therefore, the combinations of loss rate and loss pattern we used are:

Loss Rate:	1	10	20	20	20
Loss Pattern:	1	1	1	2	4

- Our next step is to simulate packet loss. Since B frames rely on the I and/or P frame both before it and after it, it is impossible to play a B frame without first transmitting all the necessary frames. Thus the actually compression sequence and transmission sequence for the frames are different from the IPB pattern we specified. For the pattern IBBPBBPBB, the transmission sequence will be IPBBPBBIBB. So even if the two frames are lost in a sequence during the transmission, when playback, they are not necessarily played adjacent to each other. Please refer to Appendix B for more details on how we simulated the lost frames.

3.3 User Study

Using the above techniques for simulating the loss in video streams, we generated MPEG files for our user study. Twenty-two unique video clips were chosen for the study. Two are perfect frames without any loss, ten are redundancy repaired with the five combinations of loss rate and loss pattern, and ten are of the same five combinations that simulate the effect of normal packet loss with repetition.

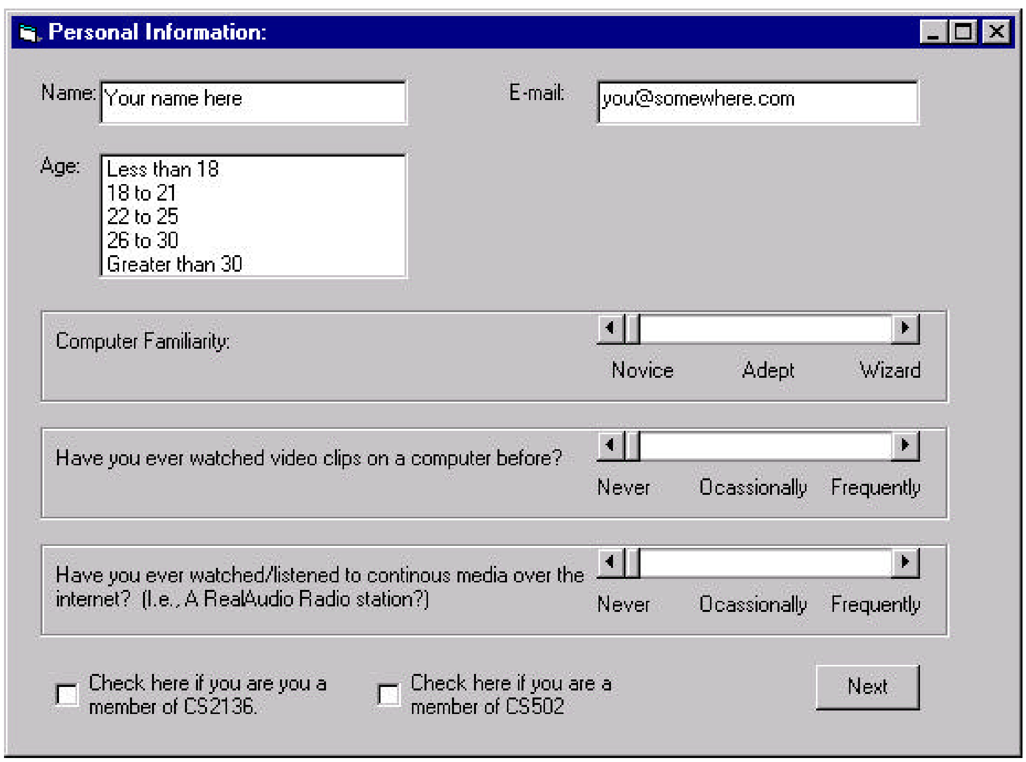

Figure 3.4 Screen Shot of the Page Where Users Enter Profile Information

The study was done on two Alpha machines running Windows NT version 4.0. The CPUs of these two machines are 600MHz. The player used was Microsoft Media Player

6.0. The average frame rate achieved was 30 fps, which matched the frame rate specified during the generation of video clips.

We designed and developed a Visual Basic program to assist the user study. A separate directory with two files is created for each new user. One of the files records the user information, such as the computer familiarity and video watching frequency. The other file records the scores that the user gives to each video clip. Figure 3.4 shows the screen shot where users are required to enter profile information. After the information is entered, we show a perfect video clip to “prepare” all users equally. The 22 clips were ordered such that the video clips with relatively low quality were not clustered together.

In order to effectively measure the perceptual quality of videos, we accepted the method proposed by de Ridder and Hamberg and provided a slider for the users to enter Perceptual Quality scores [RH97]. Figure 3.5 shows the message box displayed to the users after a video clip was displayed. The text box in the bottom of this message box shows the user’s average score they have given for all the video clips that have been displayed. The initial value of the slider is also set to the average, so that the user can easily move it up if they find the current video has a quality above average, and down if they find the current video quality below average.

Figure 3.6 lists the information of all the video clips used in the user study. The first column shows the names of the original files. The second column shows the order in which videos were displayed. The third column shows the percentage of loss. The fourth column shows the numbers of consecutive loss in the video clip. The last column shows whether the particular clip simulates the effect of normal packet loss or redundancy repaired packet loss.

Figure 3.5 Screen Shot for the Message Box for Entering Perceptual Quality Scores

File name	No.	Loss Rate	Consecutive	Redundancy
simp7	1	1	1	n
game1	2	20	2	y
married2	3	20	2	n
simp2	4	0	0	n
cnn6	5	20	2	y
soccer1	6	10	1	y
simp1	7	20	4	y
ski2	8	20	1	n
married1	9	20	4	y
news1	10	1	1	y
simp6	11	20	1	n
simp4	12	1	1	y
soccer2	13	20	2	n
ski1	14	10	1	y
cnn7	15	20	4	n
cnn8	16	10	1	n
simp5	17	20	4	n
simp3	18	20	1	y
ski3	19	0	0	n
hockey1	20	20	1	y
married3	21	1	1	n
third	22	10	1	n

Figure 3.6 Information of the Video Clips for User Study

The first column shows the name of the files. The second column shows the sequence number that the video clip to be displayed. The third column shows the *raw* loss rate of that video. The fourth column shows the consecutive loss number. The fifth column shows whether the video clip is redundancy repaired or not, “y” represents it is a redundancy repaired video.

The user study lasted for two weeks. Forty-two users took part in it. For each video the user judged the quality of it and gave a score between 0 and 100. Users ranked the quality of the video as to its clearness as well as continuity.

After gathering all the scores from the users, we examine the data to compare the average scores for redundancy repaired video clips and normal ones. Figures 3.7 and 3.8 are derived from the user study data. Figure 3.7 plots the average quality scores for the videos that have no consecutive loss. Figure 3.8 plots the average quality scores versus packet loss pattern. To get more accurate information, we calculated the confidence intervals for these data with the probability confidence to be 95%. Each point within the figure is accompanied with an error bar.

We can see that redundancy repair technique improves the quality of the video by 20% in the presence of low loss (1% *raw* loss rate). With high *raw* loss rate (20%), this technique can improve the quality of the video by 65%. As shown in Figure 3.7, the average score for 0% loss, which is considered as perfect video, is 71.80. It is the highest score in the figure. With the increase of the percent loss, the quality for both redundancy repaired videos and normal videos decreases exponentially. However, the perceptual quality with redundancy repair decreases much less than without. For a 1% frame loss, the average score for redundancy repaired videos is 69.40, which is very close to the perfect. Figures 3.7 shows that the average point for 1% loss with redundancy repair falls within the range of the confidence interval of the average quality for perfect videos. The difference between the qualities of these two kinds of videos is small and cannot be noticed in some cases. With the same percent loss, there is no overlap between the confidence intervals of those with redundancy and those without. Without the repair technique, the quality of the frame decreases dramatically to 57.65, which shows a big difference between 0% and 1% loss. Apparently, users can easily notice the seemingly small degradation of the quality.

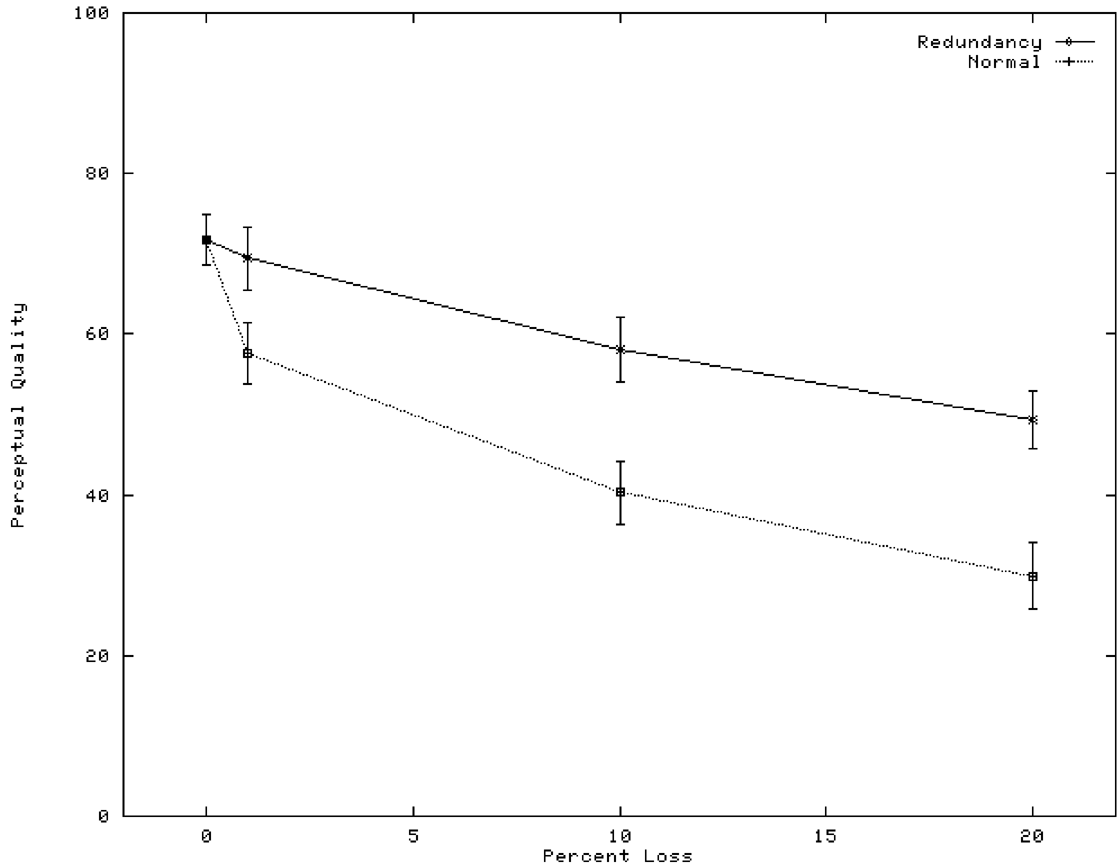

Figure 3.7 Effects of Loss Rate to the Perceptual Quality

The x-axis represents percent loss, ranging from 0% to 20%. The y-axis represents average score of the perceptual quality we gathered from the user study. The error bars represent the confidence intervals with the probability confidence to be 95%.

With the increase of the percent loss, the difference between redundancy repaired videos and normal videos becomes larger. Without redundancy, the average quality score is 29.93, while the average quality score with redundancy is 49.30. While this is far from the perfect perceptual quality scores, it is still far better than the 10% loss without redundancy.

From Figure 3.7, we can safely conclude that for single losses our approach can practically improve the perceptual quality of the transmitted video streams to that of perfect video. Videos that have some low quality image can be more easily accepted than those that have stalled frames and abrupt changed in scenes from normal packet loss.

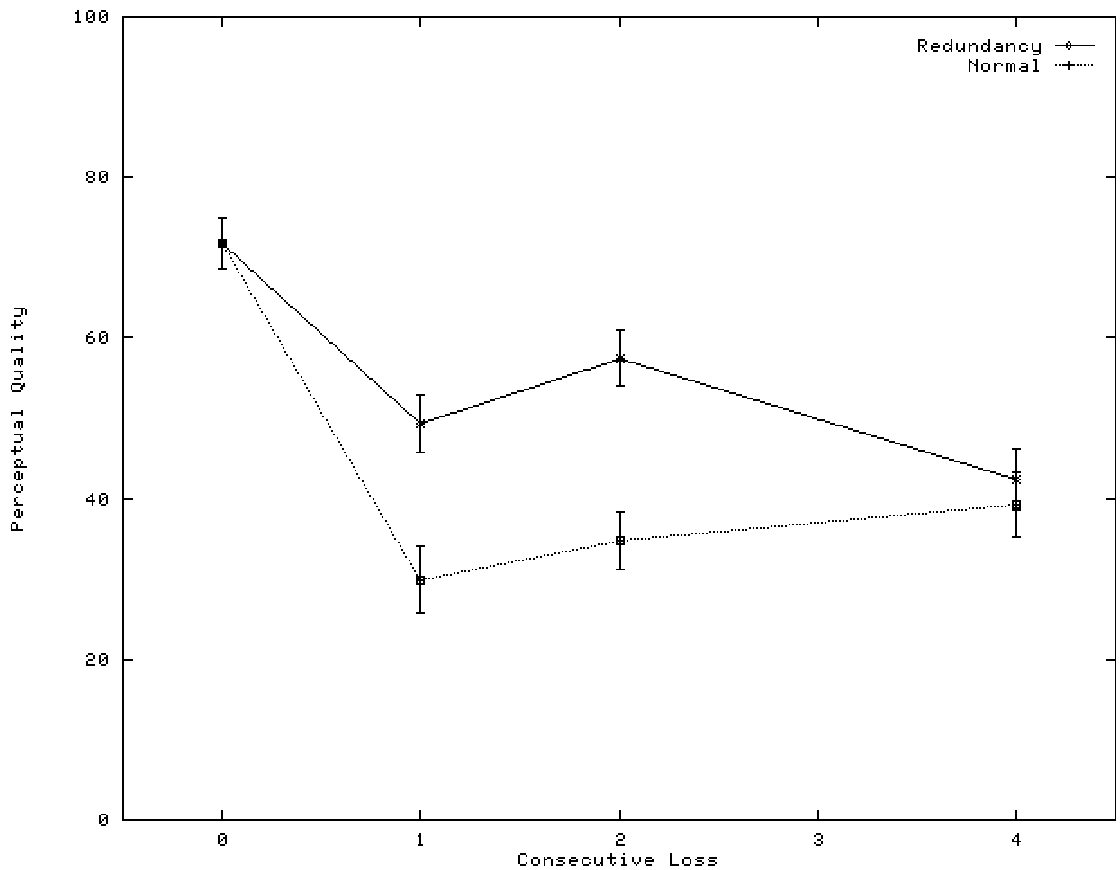

Figure 3.8 Effects of Loss Pattern to the Perceptual Quality

The x-axis represents the number of losses in a sequence. The y-axis represents the average score of perceptual quality. The error bars represent the confidence intervals with the probability confidence to be 95%.

Figure 3.8 shows the average perceptual quality of the video clips with the same loss rate, 20%, but with different loss patterns. Some of them are single losses, while others are consecutive losses. The x-axis represents the number of the lost frames in a sequence. Three numbers were chosen for the user study, 1, 2, and 4. The y-axis represents the average perceptual quality.

Note that the average quality increases as the number of consecutive losses increases. We believe it is because with higher consecutive number and same loss rate, there are less gaps within the stream than within the single losses. Thus, fewer dependent frames are lost because of the loss of other frames.

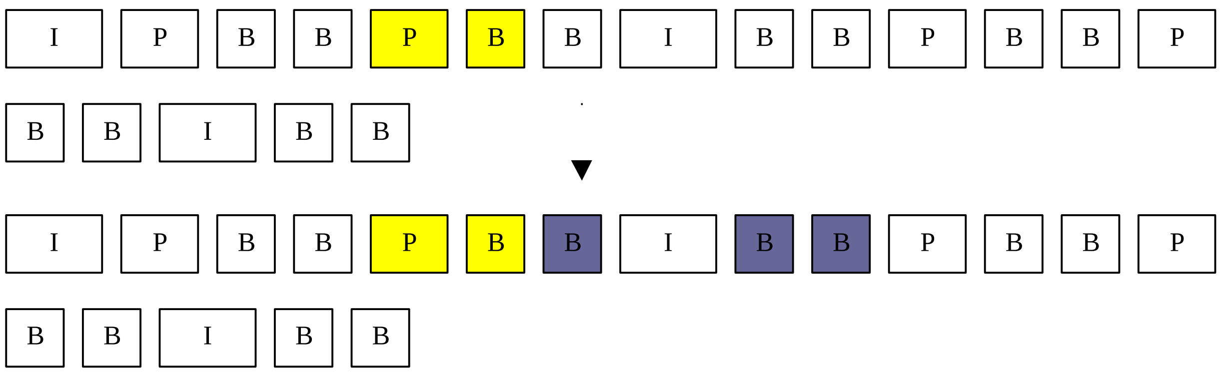

Figure 3.9.a Two Frames Lost in a Sequence

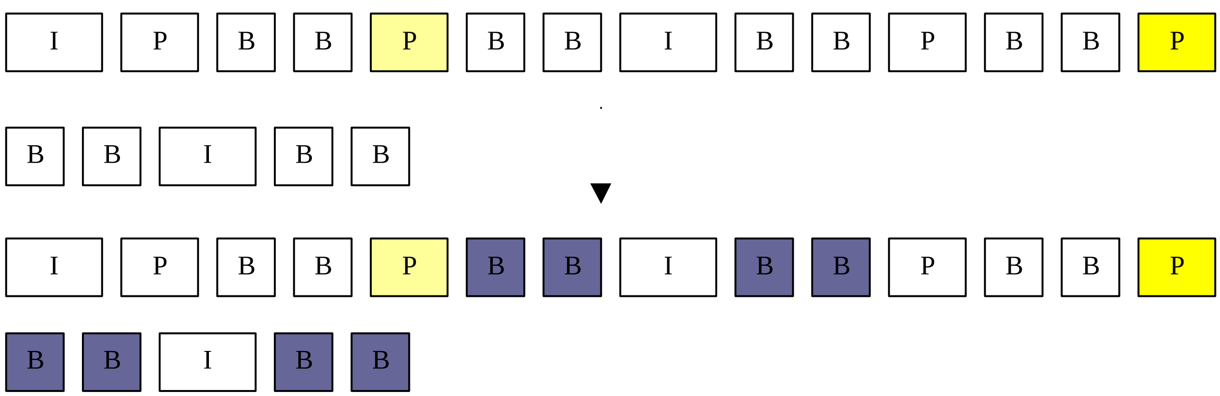

Figure 3.9.b Two single losses

Each box represents one frame. The boxes that have the color of light gray are considered to be lost while those have the color of dark gray are considered useless due to the loss of other frames.

As shown in Figure 3.9.a, a P-frame and a B-frame are lost in a sequence. Another three B-frames all depend on the P-frame, and cannot be reconstructed if the P-frame is lost. Even if the B-frame is not lost, it will become useless as well. The loss of the B-frame does not affect other frames. The real total loss of the stream is 5 frames. In Figure 3.9.b, two P-frames are lost. Each one has some other frames dependent on it. The first lost P-frame left four B-frames useless, while the second lost P-frame left another four B-frames. The real total loss of the stream is 10 frames. Although these two streams have the same *raw* loss rate, the loss pattern makes a big difference to the final streams visible to the user. Usually with the same frame rate, single losses may result in a greater number of perceptually lost frames.

However, consecutive loss does make redundancy less useful. As shown in Figure 3.8, the average perceptual quality for redundancy repaired video clips increases when the consecutive loss number changes from 1 to 2. But it goes down when the number increases further. For single losses, the redundancy can always be received and thus the loss can always be repaired. With the existence of consecutive loss, the redundancy can be lost with the primary frame in a sequence. With a consecutive loss number to be 2, it is likely that no important frames, such as the I- and P-frame, are lost. However, in a sequence of 4-frame loss, there will always be one I- or P-frame within the lost frames and the chance of this important frame can be repaired is small. We calculated the confidence interval for probability confidence of 95%. The average quality scores for videos with and without redundancy with 4 consecutive losses are very close. In fact, their confidence intervals overlap. Figure 3.8 shows that with the consecutive loss number to be 4, there is hardly any difference between the average perceptual qualities of redundancy repaired and normal video clips and there is no obvious advantage of using the redundancy repair technique when a large amount of bursty losses exist.

3.4. Summary

In this chapter, we presented our approach of how to repair network packet loss. We simulated the repair effect of this technique and conducted a user study to get the possible result. The data gathered from the user study shows that video redundancy can greatly improve the perceptual in the presence of high loss, and can result in near perfect video quality under low loss. Although delay and jitter introduced by redundancy can also affect the perceptual quality, we leave those issues to future works.

Chapter 4: System Analysis

Although our user study indicated that redundancy can improve the perceptual quality of video in the existence of packet loss, the secondary frames require extra buffer and processing time. In this chapter, we analyze the overhead that the low quality redundancy adds to the system.

In section 4.1, we discuss MPEG quality. Both file size and decoding time are examined for different MPEG quality numbers. In section 4.2, we analyze the high quality frames vs. the low quality frames. Examinations are based on the types of the videos and the average sizes of the I-, P-, and B-frames. Sizes are compared to gain a fundamental idea of how large will the overhead be. In this part, the high quality frames have the quality number 1, which is the highest quality that can get from this encoder. The low quality frames have the quality number 25 out of maximum of 31. With this number, a very low frame size can be reached, while at the same time, it can still serve the purpose as redundancy. In section 4.3, a final remark is given a summary.

4.1 MPEG Quality

In 4.1.1, we analyze the relationship between MPEG quality and file sizes. In 4.1.2, we analyze the relationship between MPEG quality and the decoding time needed.

The platform we ran our experiments on was a Digital Alpha running Digital UNIX V4.0D. The software we used for encoding and decoding are the Berkeley MPEG-2 Encoder and MPEG player. Seven MPEG quality numbers are selected for this examination: 1, 5, 10, 15, 20, 25, and 30. The contents of the video was a sports show.

4.1.1 File Size

MPEG quality numbers are peak signal-to-noise ratio, defined as (20 $\log_{10}$) * 255 / SQR(MSE), where MSE is the mean squared error. The higher the quality number, the higher the compression rate, yet the lower the frame quality. Figure 4.1 shows the relationship between the quality number and MPEG file size.

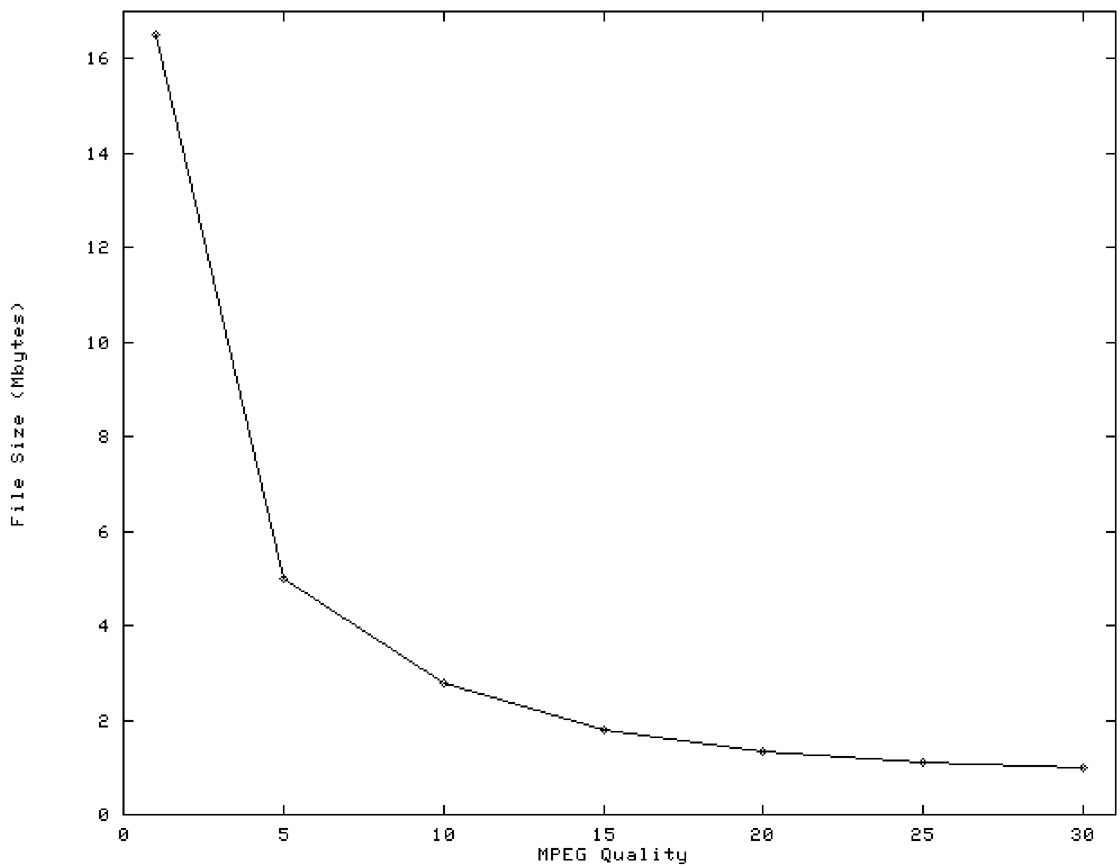

Figure 4.1.1 MPEG File Size vs. MPEG Quality

The x-axis represents the quality numbers ranging from 1 to 30. The y-axis shows the sizes of MPEG files. Each unit represents 1 Mbyte. The lower the quality number, the larger the file size. The size of the files range from 1 to 17 Mbytes.

The MPEG file size decreases exponentially with an increase in the quality number. The encoding of quality number 1 has the highest frame quality as well as the largest file size. It is about 16.5 Mbytes. With the quality number 5, the file size becomes 5 Mbytes, which is only 30.3% the size of the one with quality number 1. MPEG file size further decreases when the quality number increases to 10. The size now is only 16.9% of the one with quality number 1. The largest quality number we examined is 30. The file size with a quality number 30 is only 1 Mbyte, but the frames appear very coarse.

With these results, we can conclude that the size of the overhead relates directly to the quality number of the low quality frame. Depending on the network bandwidth availability, the size of the overhead can vary from relatively large to very small. The trade-off is not in the number of losses that can be repaired, but rather the quality of the repair data.

4.1.2 Decoding Time

Figure 4.1.2 shows the relationship between the decoding time and the MPEG quality. We used the Berkeley MPEG player to decode the files. With the primary decoding scheme, the display time includes the time used for I/O and the idle time to preserve the frame timing; not all the time is used for processing. Thus, there is not necessarily much difference between the time used for processing high quality frames and that used for processing the low quality frames. In order to measure the effects of just the decode time, we set the parameters to `mpeg_play` to be "no display" and "frame rate 0". With no display, the player will decode the MPEG file without trying to write the decoded frames onto screen. With the frame rate set to be 0, the player will decode it as fast as possible without preserving the timing constraints.

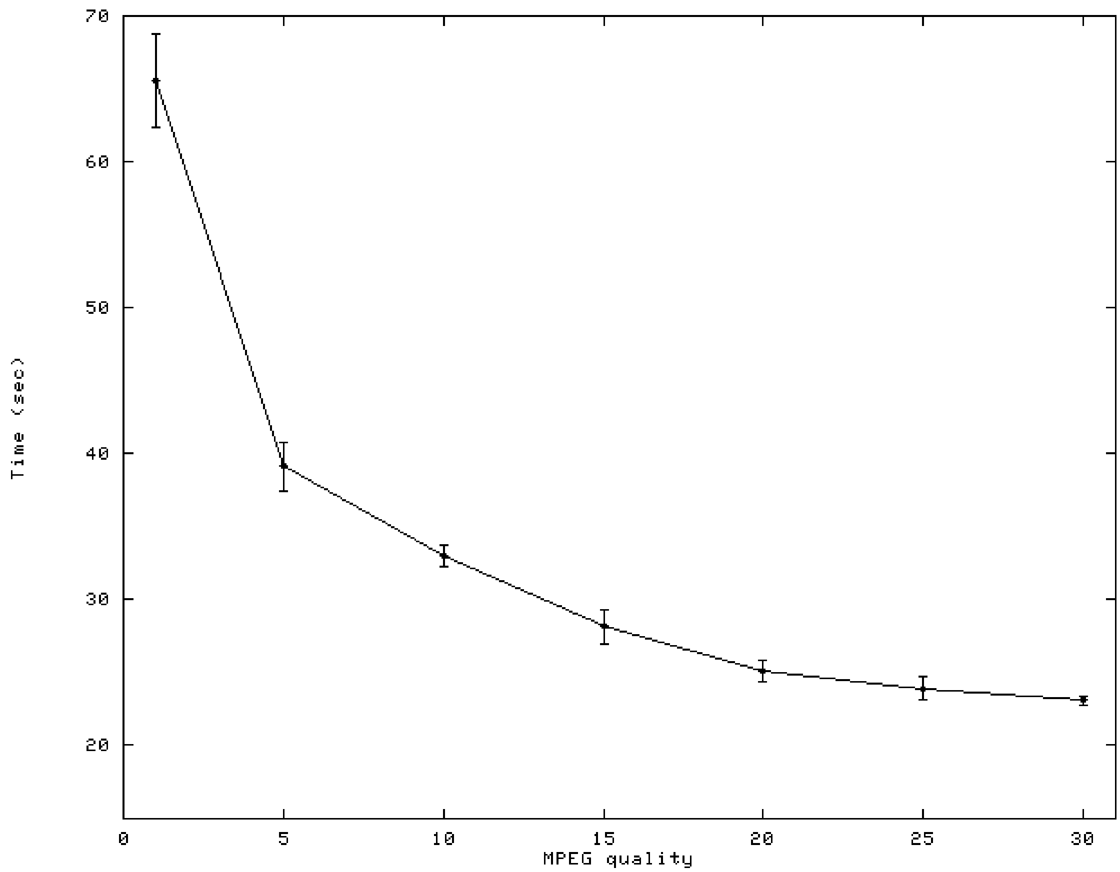

Figure 4.1.2 Encoding Quality Number vs. Decoding Time

The x-axis represents the quality numbers ranging from 1 to 30. The y-axis shows the decoding time of MPEG files. Each unit represents 1 second. The lower the quality number, the more the time used in decoding.

We decoded each of these MPEG files for 5 times when the system was lightly loaded. The movie was 30 seconds long. The average decoding time for each file as well as the confidence interval (probability confidence 95%) is shown in Figure 4.1.2. Generally, the time to decode the MPEG file is exponentially inversely related to the quality number. The lower the quality number is, the more the time used in decoding.

Comparing Figure 4.1.1 and Figure 4.1.2, we can see that these two curves are very similar to each other. We conclude that the lower the quality number is, the greater the time that is spent decoding.

During the time we prepared for user study, we also noticed that the time for encoding is also affected by the quality of the encoding. Less time was spent in encoding low quality frames (these frames are with higher quality number). However, since for many real-time video applications, such as video-on-demand, the frames are pre-encoded before transmission, there is usually no encoding time involved when the transmission is happening. Thus, we omitted the discussion of encoding quality verses encoding time. We assume that encoding of frames has a similar relationship to MPEG quality as decoding does.

4.2 High Quality vs. Low Quality

In the user study, we encoded the primary frames with quality 1, which is the best quality. For the secondary frames, we encoded it with quality 25. With this quality level, the encoded frames have a much higher compression rate. Although the frame quality of such encoding is quite coarse, under typical Internet loss, the low quality frames have fewer chances to be displayed. In the redundancy repaired frames, only a small percent of frames are low quality. We chose a quality level of 25 for the secondary frames based on tests that indicated users could notice the degradation of the clearness, but the frames still conveyed the basic information of the contents. Figures 4.2.1 – 4.2.7 all capture the system effects of quality 1 versus quality 25 frames. Figure 4.2.1 compares the sizes of primary and secondary frames for one particular video stream. Figures 4.2.2 and 4.2.3

gather the information of overhead ratio to the primary frames and Figures 4.2.4 - 4.2.7 compare the size differences for different kinds of videos.

4.2.1. Frame Size Differences

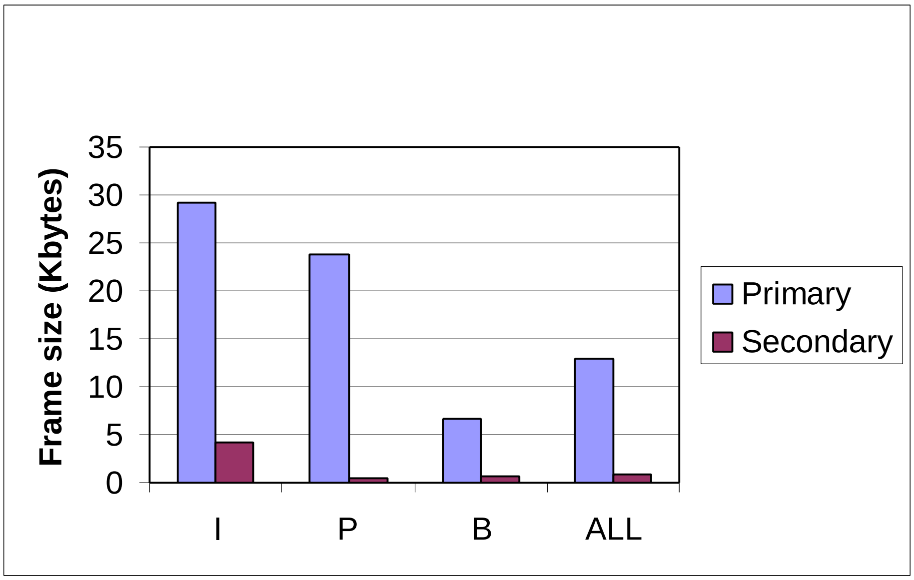

Figure 4.2.1 Frame Size Difference for Primary Frames and Secondary Frames

The x-axis represents the type of frames to be examined. Information of three kinds of MPEG frames, I-, P-, B-, as well as the average of all the frames is shown in the figure. The y-axis represents the average size of each type of frame. This experiment was done on a news show movie.

Figure 4.2.1 compares the encoded sizes of primary frames vs. secondary frames with the same news type movie. The GOP we used is IBBPBBPBB. Information about 51 I-frames, 100 P-frames and 300 B-frames for each of these two quality levels are gathered to derive the result. From Figure 4.2.1, we can see that the average size of primary (quality number 1) I-frames is 29.2 Kbytes, while the average size of secondary (quality number 25) I-frames is 4.1 Kbytes. The average ratio of secondary I-frame size over primary I-frame size is about 14.25%. The confidence interval is between 14.01% and 14.49% with the probability confidence to be 95%.

Similarly, we got the data for P-frames and B-frames. The average size of primary P-frames is 23.8 Kbytes, which is slightly smaller than that of primary I-frames. However, the average size of secondary P-frames is 0.46 Kbytes, almost 10% of average size of secondary I-frames. The average ratio of secondary P-frame size over primary P-frame size is only about 1.8%, which is extremely low, with the probability confidence to be 95%, the confidence interval is between 1.5% and 2.0%.

Average size of primary B-frames is 6.6 Kbytes, which is about ¼ the size of primary I-frames. The average size of secondary B-frames is 0.53 Kbytes. The average ratio of secondary B-frame size over primary B-frame size is 12.8%, similar to that of I-frame, with the probability confidence to be 95%, the confidence interval is between 11.6% and 14.1%. Thus the size of secondary B-frames is about 13% of the primary counterparts.

After gathering the information of all the frames, we have the following result: average primary frame size is 13 Kbytes; average secondary frame size is 10 Kbytes; average ratio of secondary frame size over primary frame size is 10.5%; with the probability confidence to be 95%, the confidence interval is between 9.6% and 11.4%. Therefore, for this video clip, the average overhead that the redundancy (secondary frames) added to the video stream is about 10% of the primary frames.

4.2.2. I-Frame, P-Frame, and B-Frame Size Differences

In the above section, we examine the frame size differences of different quality numbers for one particular video (news type video). It is also possible that the contents in the video affect the encoding and the frame sizes. Figures 4.2.2 - 4.2.5 compare the I-frame, P-frame, and B-frame sizes for four kinds of videos: animations, sports, sitcoms, and news.

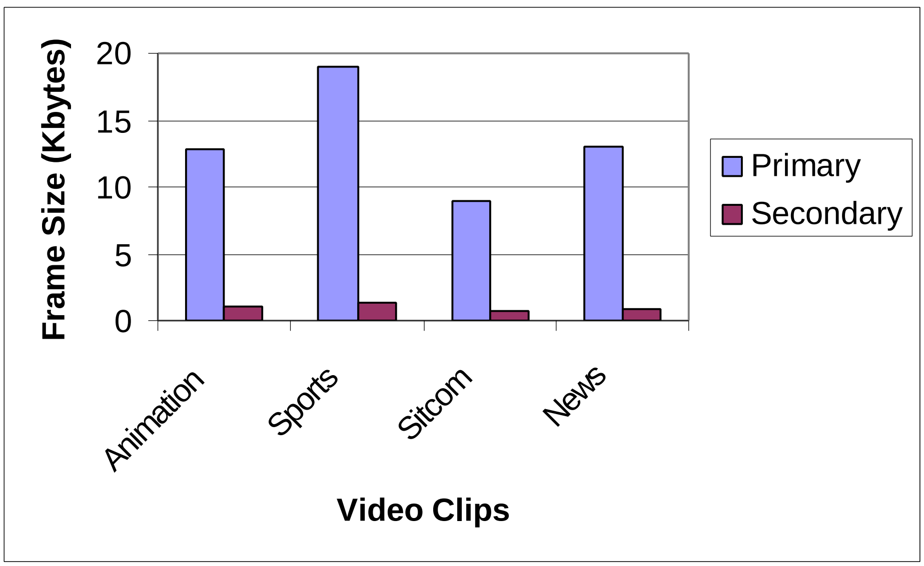

Figure 4.2.2 Frames Size Differences for Four Videos

The x-axis represents different types of videos. Four basic types are examined here, animation, sports, sitcom, and news. The y-axis represents the average size of the all frames.

Figure 4.2.2 shows the frame size differences for all the I-, P-, and B-frames. The primary frames range from 9.0 Kbytes to 19.0 Kbytes, and the secondary frames range from 0.72 Kbytes to 1.4 Kbytes. The four secondary frame size vs. primary frame size ratios are 22%, 11%, 9%, and 11%. Most of them are about 10%, which we consider primary. The 22% one results from the high ratio of I-frames and B-frames. P-frames for the animation video are quite similar to other videos.

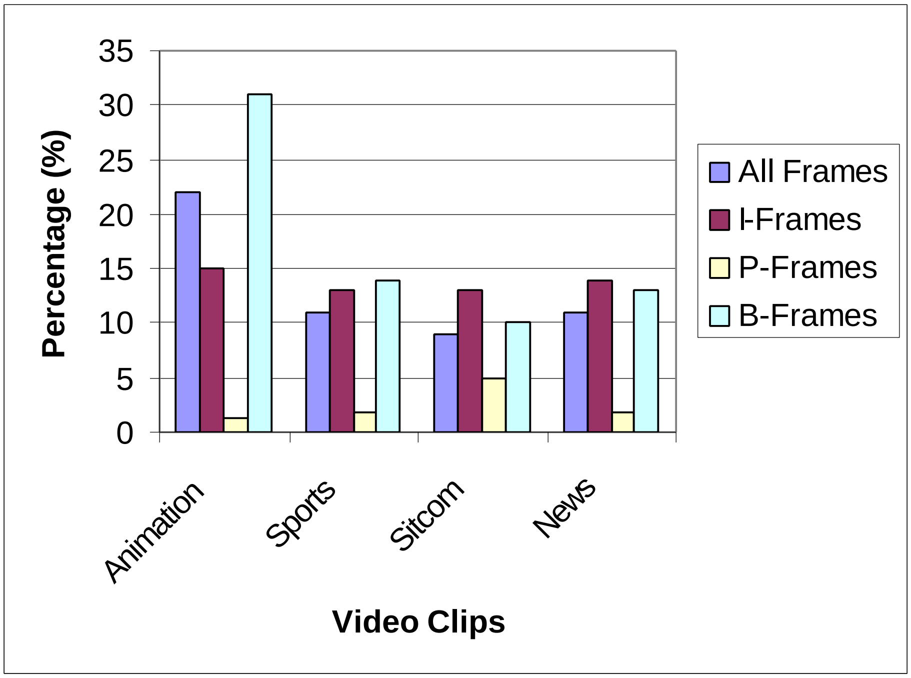

Figure 4.2.3 Ratios of the Overhead Size vs. Primary Frames Size

The x-axis represents different types of videos. Four basic types are examined here: animation, sports, sitcom, and news. The y-axis represents the average percentage of the secondary frame size over primary frame size.

In Figure 4.2.3 we can clearly see that ratios for sports, sitcom, and news are quite similar to each other. The overall average ratio, I-frame ratio and B-frame ratio are all between 9% and 14%. The ratios for all the P-frames are very low. All of them are below 5% with most of them below 2%. P-frames are relatively important within the MPEG frames and the overhead for secondary P-frames is very low suggesting that they are excellent candidates for redundancy.

Figure 4.2.4 compares the I-frame sizes for four different videos. For primary frames, the average size varies from 38.8 Kbytes to 16.9 Kbytes. Despite of the differences, all of them are more than 15k bytes. For secondary frames, the average sizes are in the range of 2.1 Kbytes and 5.9 Kbytes. Most of them are less than 5k bytes. The four secondary frame size vs. primary frame size ratios are 15%, 13%, 13%, and 14%. All of them are about 14%.

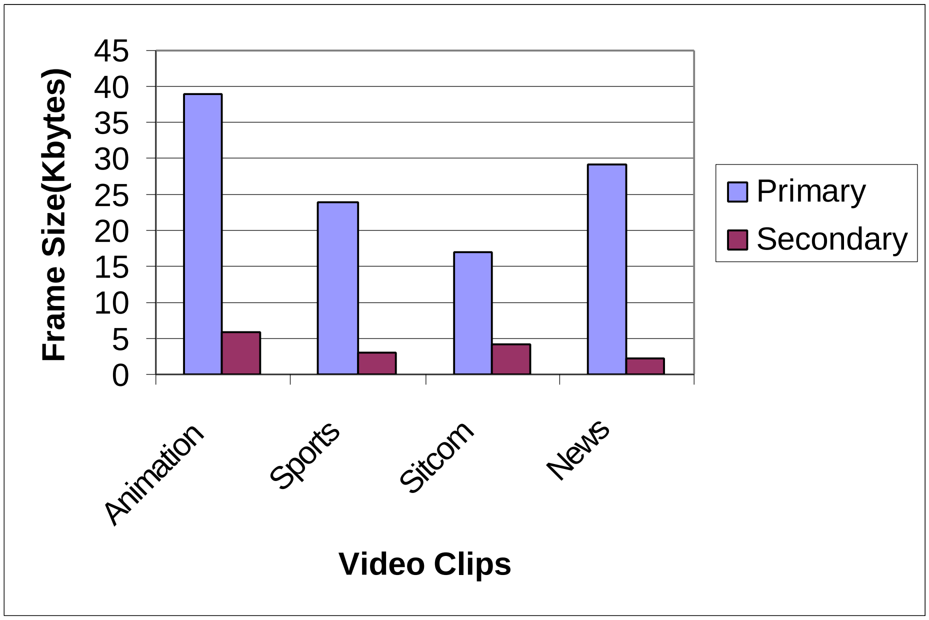

Figure 4.2.4 I-frame Size Differences for Different Videos

The x-axis represents different types of videos. Four basic types are examined here: animations, sports, sitcoms, and news. The y-axis represents the average size of the I-frames.

Similarly, in Figure 4.2.5 the P-frame sizes for different videos are compared. For primary frames, the average size varies from 28.1 Kbytes to 15.3 Kbytes. Most of them are greater than 15 Kbytes. However for secondary frames, the average sizes are in the range of 0.3 to 1.4 Kbytes. Most of them are much less them 1k. The four secondary frame size vs. primary frame size ratios are 1.3%, 1.9%, 4.9%, and 1.8%. All of them are less than 5% and most of them are less than 2%

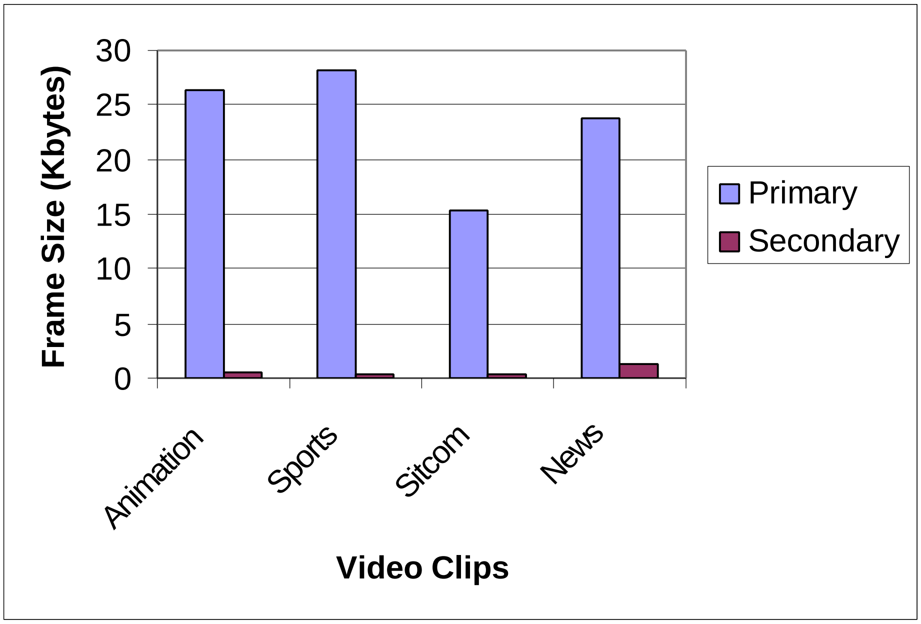

Figure 4.2.5 P-frame Size Differences for Different Videos

The x-axis represents different types of videos. Four basic types are examined here: animations, sports, sitcoms, and news. The y-axis represents the average size of the P-frames.

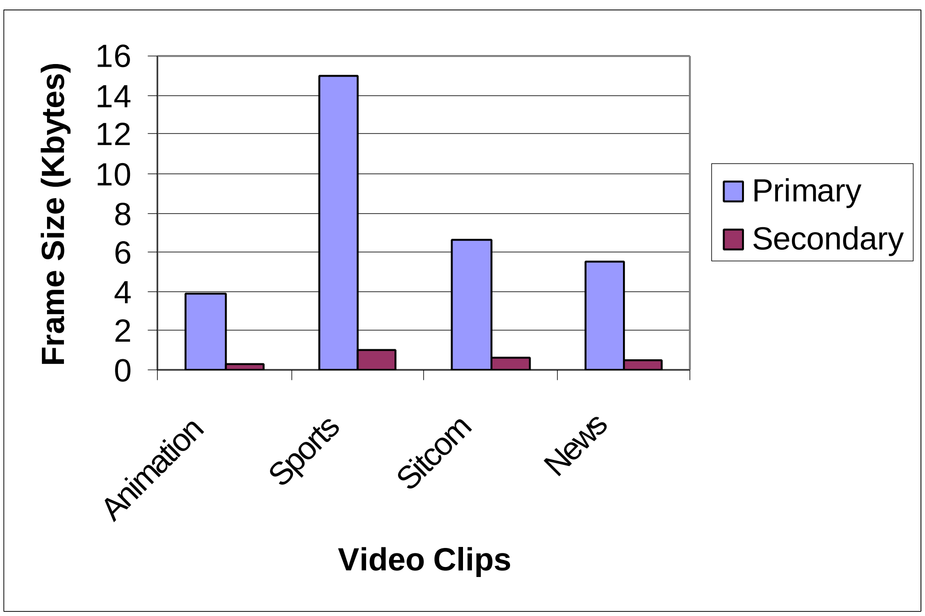

Figure 4.2.6 B-frame Size Differences for Different Videos

The x-axis represents different types of videos. Four basic types are examined here: animations, sports, sitcoms, and news. The y-axis represents the average size of the B-frames.

Similar analysis for B-frames is shown in Figure 4.2.6. The size of B-frames is much smaller than the size of I-frames and P-frames. For primary frames, the average size varies from 14.9 Kbytes to 3.9 Kbytes. Most of them are between 10k and 3k. For secondary frames, the average sizes are in the range of 0.3 to 1.0 Kbytes. Most of them are much less them 1k. The four secondary frame size vs. primary frame size ratios are 31%, 14%, 10%, and 13%. Three of them are about 13%, which is similar to the ratio of I-frames. The animation has a very high ratio (31%). Some properties of the animation video may have properties that are different much from the real world videos. The highest average size is from the sports video. We believe that it is because of the frequently changing scenes that make the adjacent frames vary much from each other more than the other videos. B-frames have the highest compression rate of MPEG frames. Even when the ratio is very high (31%), the average secondary frame size is 0.3 Kbytes, which is very low compared to the I-frames and P-frames, which have averages of 5k and 1k respectively.

Figure 4.2.7 shows the average ratios of the size of the overhead over the size of the primary frames. With the probability confidence to be 95%, we computed the confidence interval for I-frames, P-frames, B-frames, and all I-, P-, B-frames.

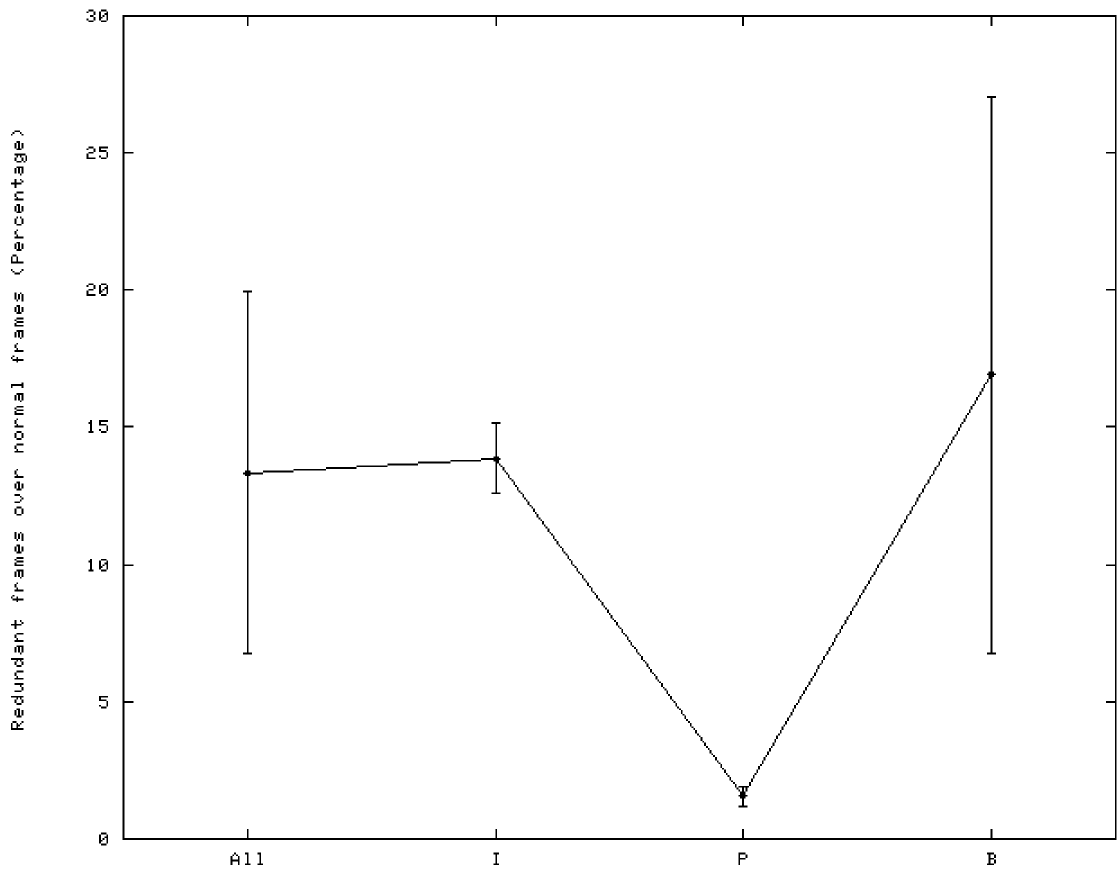

Figure 4.2.7 Ratios of Overhead Over Frame Size

In this Figure, All in x-axis represents all the I-, P-, B-frames. I represents I-frame, P-represents P-frame and B-represents B-frames.

From Figure 4.2.7, we can conclude that for I- and P-frames, there is little variance in the ratios. The ratios for different video clips tend to be close to each other. While there can be a lot of variance with the ratio of B-frames, which makes the overall ratio vary dramatically. However, since B-frames are the least important frames with MPEG encoding scheme, the need to repair lost B-frame is not as beneficial as for I-frames and P-frames. One option for us to reduce the overhead the secondary frames added to the system and network transmission is to transmit secondary frames only for I-frames and P-frames. Since the overhead sizes of these two kinds of frames are more predictable and the main purpose of this repair technique can be fulfilled by the modified version of the our approach, the large variance in B-frame overhead size is not of much importance. Even if we do transmit the secondary frames for B-frames, the absolute overhead size for each frame is about 343 bytes, which is very low compared to I- and P-frame's overhead.

4.3 Summary

In this chapter we analyzed the overhead the secondary frames added to the system and network.

Chapter 5: Conclusions

In this thesis, we present a solution to ameliorate the effects of network data loss for video data transmission. Our approach piggy-backs redundant video frames within the transmitted video stream in order to repair lost frames. At the sender, images are compressed into two versions, one with high quality and a large frame size, the other with low quality and a small frame size. High quality frames are sent to the receiver as the primary frame, while low quality frames are piggy-backed with the next primary frame as the redundant frames. In the case the primary frame is lost, the corresponding low redundant quality frame is used to replace it. We investigated the effect of this approach on users, as well as on the system.

With single frame loss, most loss can be repaired by the redundancy added to the video stream. When the loss rate is 20%, the quality of repaired video is rated about 65% greater than that of normal video. Even when the loss rate is low, such as 1%, without the existence of redundancy the degradation of video quality can be easily noticed. With 1% single loss, perceptual quality can drop as much as 20% from video with no loss, while the average quality with 1% single loss redundancy repaired video streams is very close to that of perfect videos. We conclude that video redundancy competely repairs the perceptual quality in the presence of single consequent packet loss, the major loss pattern.

We also examined how the loss pattern reduces the effectiveness of this technique. With the same loss rate, higher consecutive loss results in a better perceptual quality for normal video streams. However, for the redundancy repaired videos, this is not the case. When four consecutive frames are lost, at least one important frame (I- or P-frame) is lost and the chance of this particular frame can be repaired is low. It is unlikely that the redundancy scheme will make significant differences in the presence of bursty loss. However, a pattern of more than two frames lost in a sequence rarely happens when the loss rate is relatively low. Video redundancy is a reasonable repair method when the loss rate is under 20%. Moreover, at a very high loss rate, perceptual quality is bound to suffer under all repair schemes.

The advantage of video redundancy is that it improves the video quality in the presence of most packet loss, while the disadvantage is that it adds overhead to the system and the network. Low quality frames need time and space to be read, multiplexed, transmitted, and extracted. The extra time and space depends mainly on the size of low quality frames. The analysis shows that size of low quality frames is only between 9% and 10% that of high quality frames. If we choose to only repair I- and P-frames, the ratio can be further reduced and will add an even smaller overhead.

In summary, we proposed a method to encode two different versions of frames for the same video at the sender. We designed a means to simulate the effect of data loss and repair the lost high quality data by low quality redundancy. We conducted a user study to measure the effects of this repair technique. Results of the user study indicate that with the addition of about 10% overhead, video redundancy can greatly improve the perceptual quality of video streams in the presence packet loss.

Chapter 6: Future Work

Our redundancy repair technique may introduce some delay and jitter, which we did not measure. If each frame is to be decoded and played as soon as it arrives, there will be a halt during the display when one packet is lost. The decoder has nothing to process, but must wait for the arrival of next packet. The secondary frame will be extracted, decoded and played after the one frame halt. To keep the right timing, next frame will be played right after it, which makes an abrupt jump to the next scene. Analysis on how much jitter video redundancy introduces to the display and how to solve this problem can be an interesting issue for future research.

Some packet loss is due to network congestion. With the addition of overhead to video streams, network traffic could become more congested resulting in more packet loss. To what degree will the overhead affect network congestion could be another area for further study.

Another topic for future work is to compare video redundancy with other repair techniques and measure the effects each technique has on Perceptual Quality. For example, Interleaving introduces no system overhead, but the effect of small losses on Perceptual Quality is uncertain. Retransmission can improve the quality of each image, with a trade-off of longer delay. Media independent FEC may have lower system overhead, but the number of losses it can repair may be much less, resulting in lower Perceptual Quality.

Lastly, there are a variety of possible modifications to video redundancy. Depending on the transmission requirements and resource availability, video redundancy can be modified to serve special purposes. When a larger end-to-end delay is tolerable, it is possible to piggy-back the transmission of the secondary frame 3 or 4 or even more frames back in order to repair even in presence of bursty loss. When the buffer or network bandwidth is scarce, the omission of redundancy for B-frame can be helpful to provide better service. It is also possible to combine video redundancy with other repair techniques, such as interleaving. In the case that only part of the primary frame, such as one or more macro blocks, is lost or corrupted, the counterpart in the low quality frame can be used to repair the specific part. Application of this technique to a compression scheme other than MPEG-1 can also be studied.

Appendix A: Tools Used in the Simulation

- mpeg_encode This command takes in a sequence of input files and generate the MPEG file. The encoder is invoked in the following manner:

  mpeg_encode <options> parameter_file

  In the parameter file we need to specify the following the parameters for the encoding

  Q-Scale: The quantization scale values give a trade-off between quality and compression. Usage:

  IQ-Scale num

  PQ-Scale num

  BQ-Scale num

  Larger numbers give better compression, but worse quality. The quality numbers are peak signal-to-noise ratio, defined as $(20 \log_{10}) * 255 / SQR(MSE)$, where MSE is the mean squared error.

  IPB Pattern: The sequence of I, P, and B-frames. Usage:

  PATTERN <IPB pattern>

  Specifying Input and output files: The encoder can accept five base types of input file: PPM, PNM, JMOVIE, JPEG, and YUV. Usage:

  BASE_FILE_FORMAT format

  YUV_SIZE widthxheight

  YUV_FORMAT yuv_format

  INPUT_DIR directory

  INPUT

  filename.*.yuv [start-end]

END_INPUT

Frame rate

With this encoder, we can specify the frame rate to be one of the eight legal values.

Usage:

FRAME_RATE float

Float is one of {23.976, 24, 25, 29.97, 30, 50, 59.94, 60}.

We specify the frame rate for our video streams to be 30fps.

In the command line we could specify:

1. Encoding frames at a time: Instead of encoding an entire sequence, we can encode individual frames. These frames can later be joined together to form an MPEG file. Usage:

   -frames first_frame last_frame

   Output will be placed in separate files, one per frame, with the filenames being the normal output file with the suffix ".frame.<frame num>".

   The frames can be combined as: -combine_frames. But in the parameter file, it need to use FRAME_INPUT_DIR, FRAME_INPUT, FRAME_END_INPUT instead.

- ppmtoeyuv This command takes in a file in PPM format and convert it into EYUV format. Usage:

  ppmtoeyuv filename.ppm filename.yuv

Decoding tools: Berkeley MPEG-2 player and Microsoft Media Player.

- mpeg_play This command takes in a mpeg file and display it to screen. It works in UNIX system. Usage:

mpeg_play filename.mpg

It has an option of breaking the mpeg file into each individual frame rather than display it to the screen. It takes in a mpeg file and output frames with the name filename.mpeg.<frame num>.ppm. Usage:

mpeg_play –dither ppm filename.mpg

- Microsoft Media Player   This player works on Windows NT platform. Usage:

  mplayer2 filename.mpg

Appendix B: How to Simulate the Lost Frames

We generate a loss table based on the loss rate and loss pattern. For each frame, we use a function to decide whether it is lost or not: `rand(50/lossrate)`. Function rand(num) can generate a number between 0 and num. If this number is between 0 and 1, we would consider the frame lost. This function is for single loss only. The loss rate can be any number between 1 and 50. The possibility of the number falls into the range of 0 and 1 is (2 * loss rate) %. This is because to force it to be single loss, we examined only half of the frames. If there were 100 frames in the video stream, only those with odd sequence number are examined. Consider loss rate to be 10%, rand() would generate a number between 0 and 5. For 50 frames, approximately 10 of them will get the number less than 1. Overall, about 10% of the frames will be "lost".

For the consecutive loss, we simplified it to narrow down the study focus. In one video, all the loss would have the same consecutive loss number. If we specify it to be 2, all the lost frames would be 2 adjacent frames. The function to determine this kind of loss is modified to: `rand((50*num)/lossrate)`, where the num is the consecutive loss number.

The relationship between the actual frame sequence number and the transmission sequence number for this pattern is:

For I frame: $n = tn(n) + 2$, where n != 1. $n = tn(n)$, where $n = 1$.

For P frame: $n = tn(n) + 2$.

For B frame: $n = tn(n) - 1$.

In these equations, n is the actual sequence of the frame, where $tn(n)$ is the sequence number of the frame during the transmission.

With the above functions, we could decide for each particular frame in the video stream, whether to "lose" it or not. A loss table can be generated to map the information of all the frames in one video stream. Figure A.1 shows an example of the loss table. In this table, the I, P, and B defines the type of the frame. Each of them is followed by a number of 0, 1, and 2. These numbers indicate the status of the frames.

0: This frame is not lost.

1: This frame is lost and there is a secondary frame available for repair.

2: This frame is lost and the secondary frame is not available for repair.

```
I 0   B 0   B 0   P 0   B 0   B 0   P 0   B 0   B 0
I 0   B 0   B 0   P 0   B 0   B 0   P 0   B 0   B 0
I 0   B 0   B 0   P 0   B 0   B 0   P 0   B 0   B 0
I 0   B 0   B 2   P 0   B 0   B 0   P 1   B 0   B 0
I 0   B 0   B 0   P 0   B 0   B 0   P 0   B 0   B 0
I 0   B 0   B 0   P 0   B 0   B 0   P 0   B 0   B 2
I 0   B 0   B 0   P 1   B 0   B 0   P 0   B 0   B 0
I 0   B 0   B 0   P 0   B 0   B 0   P 0   B 0   B 0
I 0   B 0   B 0   P 0   B 0   B 2   P 0   B 0   B 0
I 1   B 0   B 0   P 0   B 0   B 0   P 0   B 0   B 0
I 0   B 0   B 0   P 0   B 0   B 0   P 0   B 2   B 1
I 0   B 0   B 0   P 0   B 0   B 0   P 0   B 1   B 0
I 2   B 0   B 0   P 0   B 0   B 0   P 0   B 0   B 0
I 0   B 0   B 0   P 0   B 0   B 0   P 0   B 0   B 2
I 0   B 0   B 0   P 1   B 0   B 0   P 0   B 0   B 0
I 0   B 0   B 0   P 0   B 0   B 0   P 0   B 0   B 0
I 0
```

Figure A.1 Example of Loss Table

From the loss table, we could derive the repair table, which contains the information of whether the frame is lost, whether it is affected by some other frame(s), or whether it can be repaired. Figure A.2 shows an example of the repair table.

I 0	B 0	B 0	P 0	B 0	B 0	P 0	B 0	B 0
I 0	B 0	B 0	P 0	B 0	B 0	P 0	B 0	B 0
I 0	B 0	B 0	P 0	B 0	B 0	P 0	B 0	B 0
I 0	B 0	B 2	P 0	B 3	B 3	P 1	B 3	B 3
I 0	B 0	B 0	P 0	B 0	B 0	P 0	B 0	B 0
I 0	B 0	B 0	P 0	B 0	B 0	P 0	B 0	B 2
I 0	B 3	B 3	P 1	B 3	B 3	P 3	B 3	B 3
I 0	B 0	B 0	P 0	B 0	B 0	P 0	B 0	B 0
I 0	B 0	B 0	P 0	B 0	B 2	P 0	B 3	B 3
I 1	B 3	B 3	P 3	B 3	B 3	P 3	B 3	B 3
I 0	B 0	B 0	P 0	B 0	B 0	P 0	B 2	B 1
I 0	B 0	B 0	P 0	B 0	B 0	P 0	B 7	B 4
I 2	B 4	B 4	P 4	B 4	B 4	P 4	B 4	B 4
I 0	B 0	B 0	P 0	B 0	B 0	P 0	B 0	B 2
I 0	B 3	B 3	P 1	B 3	B 3	P 3	B 3	B 3
I 0	B 0	B 0	P 0	B 0	B 0	P 0	B 0	B 0
I 0								

Figure A.2 Example of the Repair Table

This table is derived from the table in Figure 3.6. The status of the frames is:

0: The frame is intact. No action required.

1: The frame is lost. Not affected by other frames. Redundancy for this frame exists, so that it can be repaired.

2: The frame is lost. Not affected by other frames. Redundancy for this frame does not exist, so it cannot be repaired.

3: The frame is intact. Affected by other frames. The dependent frame can be repaired.

4: The frame is intact. Affected by other frames. The dependent frame cannot be repaired.

5: The frame is lost. Affected by other frames. The dependent frame can be repaired. The examined frame can be repaired.

6: The frame is lost. Affected by other frames. The dependent frame can be repaired. The examined frame cannot be repaired.

7: The frame is lost. Affected by other frames. The dependent frame cannot be repaired.

After we got the repair table we finally “lose” the frames. Without the redundancy technique, all the frames with the status other than 0 will be considered lost. Therefore the original frame is removed, with the previous frame takes its place and played again from the final MPEG files.

Still using the repair table, we “repair” the corrupted video streams we just created. Both high quality and low quality frames will be generated at this stage. For each frame, we need to make the following action according to it status.

0 – No action

1 - Generate low quality frame and copy it

2 - Repeat the previous frame

3 – No Action

4 - Repeat the previous frame

5 - Generate low quality frame and copy it

6 - Repeat the previous frame

7 - Repeat the previous frame

Before the two groups of the frames are created, the frames that cannot be repaired by the redundancy should be “repaired” by the repetition technique the way just as we discussed in the previous section. Two parameter files are used for generating the frames. The primary frames have the quality level 1 while the secondary frames have the quality level 25.

References

[bm2] *Berkeley MPEG-2 player*, http://bmrc.berkeley.edu/frame/research/mpeg/.

[GBC98] Jason Gerek, William Buchanan, Mark Claypool, *mmlib – A Library for End-to-End Simulation of Multimedia over a WAN*, MQP CSMLL-ML98, May 1998.

[GKL+98] Steven Gringeri, Bhumip Khasnabish, Arianne Lewis, Khaled Shuaib, Roman Egorov, Bert Basch, *Transmission of MPEG-2 Video Streams over ATM*, *IEEE Multimedia*, Vol. 5, No. 1, Jan-Mar, p.58-71, 1998.

[GT98] G.Ghinea, J.P. Thomas, *QoS Impact on User Perception and Understanding of Multimedia Video Clips*, ACM Multimedia 1998.

[Ha97] Mark Handley, *An Examination of Mbone Performance*, USC Information of Science Institute Research Report: ISI/RR-97-450, 1997.

[HSH+95] Vicky Hardman, Martina Angela Sasse, Mark Handley, Anna Watson, *Reliable Audio for Use over the Internet*, in Proceedings INET95, Ohahu, Hawaii, 1995.

[LPP+97] Xue Li, Sanjoy Paul, Pramod Pancha, Mostafa Ammar, *Layered Video Multicast with Retransmission (LVMR) : Evaluation of Error Recovery Schemes*, Proceedings of NOSSDAV 97, St. Louis. MO, May 1997.

[LV97] L. Rizzo, L. Vicisano, *A Reliable Multicast Data Distribution Protocol Based on Software FEC Technique*s, Proc. of The Fourth IEEE Workshop on the Architecture and Implementation of High Performance Communication Systems (HPCS'97), Sani Beach, Chalkidiki, Greece June 23-25, 1997.

[KMR93] Hemant Kanakia, Partho P. Mishra, Amy Reibman, *An Adaptive Congestion Control Scheme for Real-Time Packet Video Transport*, ACM SIGCOMM’93 – Ithaca, pp. 20-31, N.Y., USA September 1993.

[KT97] Marwan Krunz, Satish K. Tripathi, *On the Characterization of VBR MPEG Streams*, Performance Evaluation Review (Proceedings of ACM SIGMETRICS'97 Conference), Vol. 25, No. 1, pp. 192-202, June 1997.

[MM96] Matthew Mathis, Jamshid Mahdavi, *Forward acknowledgement: refining TCP congestion control*, ACM SIGCOMM Computer Communication Review,Vol. 26, No. 4, Oct, Pages 281-291, 1996.

[Ma94] Bruce A. Mah, *Measurements and Observations of IP Multicast Traffic*, Technical Report UCB/CSD-94-858, University of California, Berkeley, CA, December 1994.

[mmp] *Microsoft Media Player6.0*, http://www.freeware.com.

[MP96] Joan L. Mitchell, William B. Pennebaker, *Mpeg Video: Compression Standard*, Chapman & Hall; ISBN: 0412087715, 1996.

[PHH98] Colin Perkins, Orion Hodson, Vicky Hardman, *A Survey of Packet-Loss Recovery Techniques for Streaming Audio*, IEEE Network Magazine, September/October 1998.

[PKH+97] C. Perkins, I. Kouvelas, O. Hodson, V. Hardman, M. Handley, J. C. Bolot, A. Vega-Garcia, S. Fosse-Parisis, *RTP Payload for Redundant Audio Data*, IETF Audio/Video Transport Working Group, September 1997, RFC2198.

[RH97] H. de Ridder and R. Hamberg, *Continuous Assessment of Image Quality*, SMPTE Journal, February Pages 123-128, 1997

[SCF+96] H. Schulzrinne, S. Casner, GMD Fokus, R. Frederick, V. Jacobson, *RTP: A Transport Protocol for Real-Time Applications*, IETF Network Working Group, January 1996, RFC1889.

[SN95] Ralf Steinmetz, Klara Nahrstedt, *Multimedia: Computing, Communications & Applications*, Prentice-Hall Inc; ISBN 0-13-324435-0, 1995.

[WS97] A. Watson & M. A. Sasse, *Multimedia Conferencing via Multicast: Determining the Quality of Service Required by the End-User*, Proceedings of the 1997 International Workshop on Audio-Visual Services Over Packet Networks 15-16 September 1997.

[WS98] A. Watson & M. A. Sasse, *Measuring perceived quality of speech and video in multimedia conferencing applications*, In Proceedings of ACM Multimedia'98, Bristol, UK, September 1998.