Abstract

Traditionally, the Internet had been dominated by text-based applications such as file transfer, electronic mail and recently the Web. With the rapid improvement in computer and network technologies, high-bandwidth, interactive streaming multimedia applications are now possible on the Internet. However, the Internet does not provide the necessary Quality of Service (QoS) guarantees needed to support high-quality, real-time multimedia transmission, causing Internet multimedia applications to suffer from delay, jitter and loss. Among these, loss, typically caused by network congestion, degrades the perceptual quality of multimedia streams the most. Interleaving is a media repair technology that ameliorates the effects of loss by spreading out bursty packet losses. It first resequences data before transmission to help distribute packet loss, and returns the data to their original order at the receiver. Interleaving has been applied successfully to audio, but, to the best of our knowledge, has not yet been applied to video. In this thesis, we propose an interleaving approach for Internet video. We apply our approach to MPEG and evaluate the benefits of interleaving to perceptual quality with a user study. We find that interleaving adds a small amount of delay and bandwidth overhead, while significantly improving the perceptual quality of Internet video.

Acknowledgements

I would like to thank my advisor, Mark L. Claypool for his wisdom, understanding, flexibility and encouraging spirit that shinning throughout this work. I would also like to thank my thesis reader, Robert E. Kinicki for his many valuable insights, error discoveries and great suggestions. I want to thank my friend Boyou Chen for his consistent active support. I would also like to extend my appreciation to all the people that participated in our user study, without whom this work could not be completed in success.

Table of Content

Abstract ii
Acknowledgements iii
Chapter 1 Introduction 1
Chapter 2 Related Work 7
2.1 Sender Based Error Control Techniques 7
2.1.1 Forward Error Correction (FEC) 8
2.1.1.1 Media Independent FEC 8
2.1.1.2 Media Specific FEC 9
2.1.2 Rate Adapting 10
2.1.3 Channel Coding 11
2.1.4 Interleaving 11
2.2 Receiver Based Error Concealment Techniques 13
2.2.1 Insertion Based Repair 14
2.2.2 Interpolation Based Repair 16
2.2.3 Regeneration Based Repair 17
2.3 Retransmission Based Technique 17
2.4 Hybrid Error-Control Schemes 19
2.5 MPEG Encoding 20
2.6 Network Properties 24
2.6.1 Services in Current Internet 24
2.6.2 Loss Characteristics in IP-based Networks 25
Chapter 3 Interleaving Approaches and Implementations 26
3.1 Effects of Packet Loss On a Video Stream 26
3.2 Whole-Interleaving 28
3.3 Partial-interleaving Approach 31
3.4 Bandwidth Overhead 33
3.5 Implementation 36
3.5.1 Simulation Tools 36
3.5.2 Whole-interleaving Implementation 38
3.5.3 Partial-interleaving Implementation 43

3.5.4 Simulation of Loss 44
Chapter 4 User Study on Perceptual Quality and Result Analysis 46
4.1 User Study on Perceptual Quality of Whole-Interleaving 46
4.2 Evaluation on Perceptual Quality of Partial-Interleaving 51
4.3 Result Analysis 52
4.3.1 User Background 52
4.3.2 MPEG Quality 54
4.3.3 Loss Rate 57
4.3.4 Movie Type 59
4.3.5 Interleaving Distance 61
Chapter 5 Future Work 63
Chapter 6 Chapter 6 Conclusions 65
Appendix A: Simulation Tools 68
References 75

Chapter 1 Introduction

Traditionally, the Internet had been dominated by text-based applications such as file transfer, electronic mail and recently the Web. With the rapid improvement in computer and network technologies, interactive multimedia applications have become possible on the Internet. Voice, video clips and animations are getting more and more popular. Multimedia products such as Internet telephony, Internet TV and videoconferencing have appeared on the market. In the future, people will enjoy other multimedia products in distance learning, distributed simulation, video-on-demand and more.

However, the Internet does not provide the necessary Quality of Service (QoS) guarantees that are needed to support high-quality, real-time video transmission. For traditional applications like file transfer, reliable transfer is more important than low delay. Multimedia applications, on the other hand, are sensitive to delay but can tolerate some data loss. Transmission Control Protocol (TCP), the dominant transport protocol in the Internet, provides a reliable service, and the sending rate is controlled by a congestion window, which is halved for every window of data containing a packet drop. While TCP is feasible for text-based applications, halving the sending rate in response to a single packet loss is unnecessarily severe for multimedia applications, as it can noticeably reduce the user-perceived quality. User Datagram Protocol (UDP), on the other hand, offers a best effort service without abrupt changes in the sending rate. A number of applications have emerged which use Rate-based Transport Protocol (RTP) or UDP to deliver continuous media streams. However, due to the unreliable nature of UDP, the best-effort service it provides has no guarantees for loss or delay. In recent years, the fast growth of networks, especially the success of the World Wide Web has extraordinarily increased the amount of traffic on the Internet. However, this same success frequently leads to network congestion, thus packet loss that degrades multimedia quality.

Because of these problems, multimedia applications over the Internet often suffer from delay, jitter and loss. Of these, packet loss is often the biggest detriment to multimedia quality. 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 [5]. Recovery from losses in a multimedia application becomes a key problem to achieve better multimedia performance on the Internet.

A number of techniques exist by which the effects of packet loss may be repaired [2]. Among those, the most widely used technologies include retransmission, redundant transmission, forward error correction and interleaving. Retransmission can guarantee recovery of lost packets but has a cost of higher latency and potentially large bandwidth overhead, thus it is not suitable for interactive multimedia applications. Redundant transmission and forward error correction add redundant data to a media stream, so as to increase the probability that lost packets can be recovered. Interleaving resequences data before transmission to help distribute packet loss, and returns them to their original order at the receiver.

The idea of interleaving is simple yet powerful. Interleaving assumes that better perceptual quality can be achieved by spreading out bursty packet losses in a media flow. In other words, several small gaps are better than a big gap in a multimedia flow.

For example, assume there is a frame consisting of several characters of information:

WorcesterPolytechnicInstitute

Assume that during transmission several characters in the frame get lost:

terPolytechnicInstitute

The first word is then very hard to reconstruct. However, the original frame can be interleaved as:

otlhnuWsocItreynstcrtiteePeci

After applying the same loss in the interleaved frame, the frame then can be reconstructed as:

WrceserPoytecnicIstitute

It is much easier to "interpolate" the missing letters. The same idea has been applied to audio streams as a loss recovery technique [8]. However, it is known that the human visual system is less sensitive as compared to human auditory system, thus a small gap in a video stream is less noticable than a small gap in an audio stream, suggesting that interleaving may be even more effective for video than for audio. To the best of our knowledge, interleaving has not yet been applied to video.

In this thesis, we apply interleaving to video streams. The sender resequences the video stream before transmitting, so that original adjacent units are separated by a guaranteed distance in the transmitted stream, and returned to their original order at the receiver.

A challenge in applying interleaving to video is the great bandwidth and dependent encoding techniques required by video. Compared with traditional textual applications, multimedia traffic requires much higher bandwidth, and the typical end-to-end data rates on the Internet are not sufficient for uncompressed video communications. Video streams are usually compressed before transmission on the Internet.

MPEG-1 and its standard series (MPEG-2, MPEG-4, etc) were developed for compressing motion pictures and multimedia, in contrast to JPEG, which is used for still images. In this thesis, we use MPEG-1 to compress video streams. In MPEG-1, a video stream is a sequence of pictures, also called frames. Each video sequence is divided into one or more groups of pictures, and each group of picture is composed of one or more frames of three different types, I-, P- and B-. I-frames (intra-coded frames) are coded independently, entirely without reference to other frames. P- and B-frames are compressed by coding the differences between the frame and reference I- or P- frames.

A frame consists of macroblocks, as sketched in Figure 1.1. Each macroblock consists of a 16x16 sample array of luminance (grayscale) samples together with one 8x8 block of samples for each of two chrominance (color) components. The 16x16 sample array of luminance samples is actually composed of four 8x8 blocks of samples, and these 8x8 blocks are the units of data that are fed to the compression models.

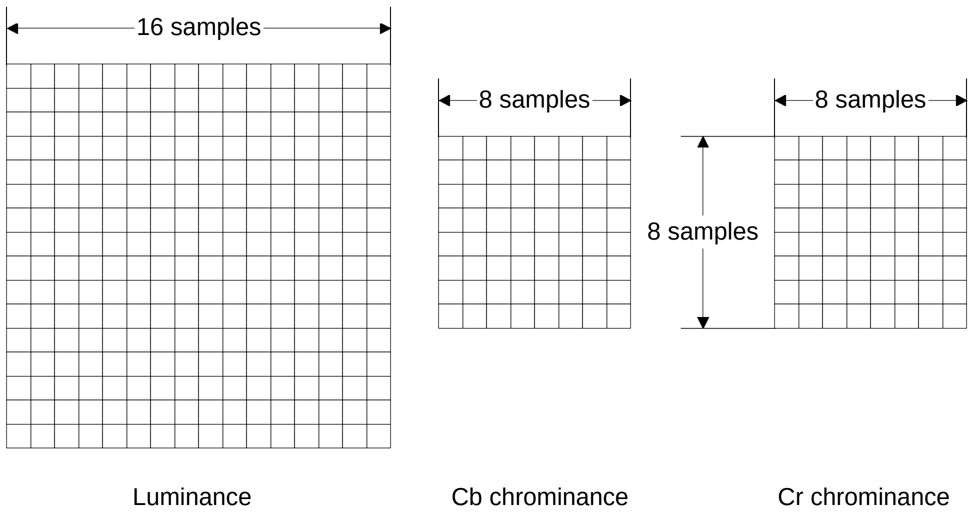


Figure 1.1 Structure of MPEG Macroblock

In our video stream interleaving algorithm, the size of the basic unit is a very important parameter. One choice is to use the whole frame as an interleaving unit. We call this *whole-interleaving*. Whole-interleaving is efficient since details within each picture are ignored. However, some size overhead is introduced into the video stream since interleaving changes the order of the frames, thus altering dependency relationships used to compress inter-coded P- or B- frames. We address the problem by encoding the interleaved video stream with selected lower quality scale factors in the P- and B- frames, so that with frame loss, the interleaved video stream can still gain higher perceptual quality than those non-interleaved video streams, although the latter are encoded with higher quality scale factors.

We conduct a user study to measure the performance of the interleaving algorithm on perceptual quality. Several loss rates are applied to the video streams, and the interleaved video streams and non-interleaved streams under the same loss rates are then evaluated by users. Results of the user study show that in the presence of packet loss, interleaving can significantly improve the perceptual quality of a video stream. The details of the user study and the results are discussed in Chapter 4.

The remaining chapters of the thesis are organized as follows. In Chapter 2, we discusses the related work. Chapter 3 contains an in-depth discussion of our interleaving approaches. Chapter 4 gives a detailed description about how our user study is performed, including the results and analysis for applying interleaving on video streams. Chapter 5 provides a brief discussion for possible future work. In Chapter 6, we draw conclusions to our work.

Chapter 2 Related Work

This chapter describes the work related to loss repair of multimedia applications over a network. Topics discussed include sender based error control techniques, receiver based error concealment techniques, retransmission based techniques, hybrid error control schemes, MPEG encoding and network properties.

2.1 Sender Based Error Control Techniques

All the error control techniques discussed in this section require the participation of the sender of a multimedia stream to achieve recovery from packet loss. These techniques can be further split into two major classes: active and passive. The taxonomy is illustrated in Figure 2.1.

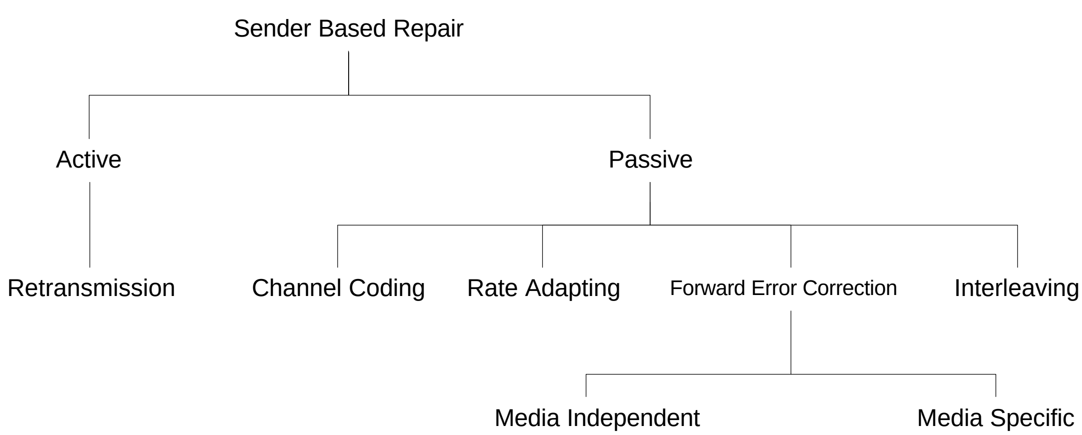


Figure 2.1 A Taxonomy of Sender Based Repair Techniques

2.1.1 Forward Error Correction (FEC)

The idea of FEC is to avoid retransmission by adding data [1]. For this purpose, FEC transmits some redundant data, called parities, together with original data to allow reconstruction of lost packets at the receiver. Two kinds of parities may be added to a stream: those that are independent of the contents of that stream and those that use knowledge of the stream to improve the repair process.

2.1.1.1 Media Independent FEC

There has been much interest in the provision of media independent FEC using block, or algebraic, codes to produce additional packets for transmission to aid the correction of losses [8]. A large number of block coding schemes exist. They were originally designed for the detection and correction of errors within a stream of transmitted bits and so a set of check bits were generated from a stream of data bits. Packet streams, such as video streams over Internet, are concerned about the loss of entire packets, so those block coding schemes are applied across the corresponding bits in blocks of packets.

One popular block coding scheme is parity coding. It allows one parity packet to be computed for a given set of original packets using the exclusive OR (XOR) operation. In one scheme that has been implemented by Rosenberg [15], 1 parity packet is transmitted after every $n - 1$ data packets. Provided there is just one loss in every n packets, then that loss is recoverable. This is illustrated in Figure 2.2.

The advantage of media independent FEC is that its operation does not depend on the contents of the packets, and the repair is an exact replacement for a lost packet. Also the scheme is relatively simple to implement. The disadvantages are that it causes additional delay and increased bandwidth, and it is difficult to implement the decoder.

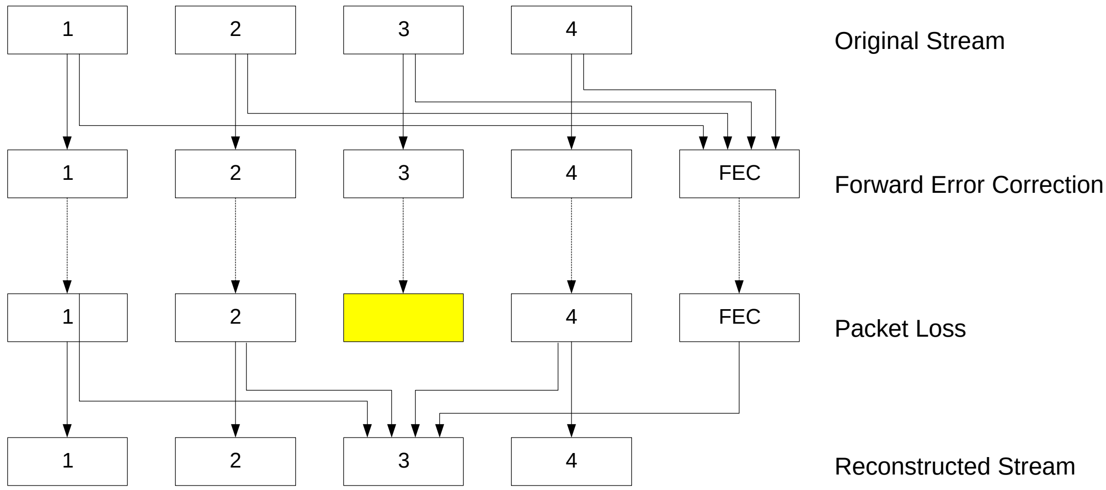


Figure 2.2 Repair Using Parity FEC

2.1.1.2 Media Specific FEC

A simple means to protect against packet loss is to transmit each unit of audio in multiple packets [8]. If a packet is lost, another packet that contains the same unit will be able to recover the loss, as illustrated in Figure 2.3.

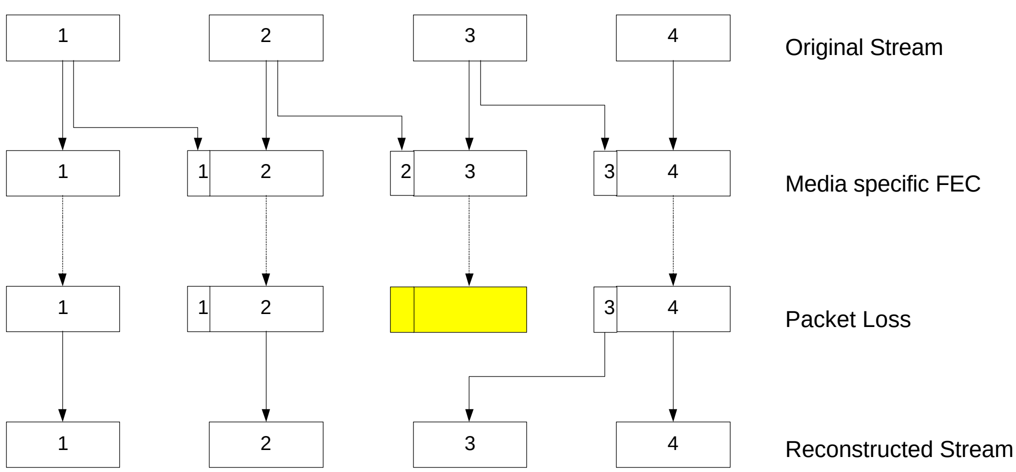


Figure 2.3 Repair Using Media Specific FEC

Two copies of each packet are transmitted, in which the first copy is referred to as the primary encoding and subsequent transmissions as secondary encodings. The sender decides on whether or not to use the same coding scheme in both primary and secondary encodings. However, compared to the primary encoding, the secondary is encoded using a lower-quality, thus lower-bandwidth encoding. Note that it is not necessary to transmit media specific FEC for every packet, a decision which could be made based on the policy specified in the codec.

Like media independent FEC, media specific FEC also incurs an overhead in terms of packet size, since redundant data is added to the media stream. Unlike most of the other sender-based techniques, this scheme has the advantage of low-latency, with only a single-packet delay added. For interactive applications, where large end-to-end delays are not tolerated, media specific FEC is a good candidate to be used as the loss recovery technique.

2.1.2 Rate Adapting

Rate adapting is a technique where the sender dynamically adapts the transmission rate to network conditions [4]. A maximum threshold and minimum threshold for the packet loss rate are set at the sender, typically with the value of 15% and 5%, respectively. If the sender learns that the packet loss rate exceeds the maximum threshold, it responds to this situation by slowing down the transmission. On the other hand, if the packet loss rate is below the minimum threshold, the sender increases the transmission rate. By increasing or decreasing the bandwidth according to the packet loss rate, this technique is able to react to congestion in the network in a proper way. Although rate adapting is sender controlled, it requires feedback from the receiver about the loss rate, and probably cannot respect most applications' loss tolerances alone. Therefore it is often used in combination with other error-control schemes, typically FEC or retransmission [4].

2.1.3 Channel Coding

In channel coding, multiple channels are used to transmit the components of a single encoded stream [4]. For example, in a voice application, for each voice sample, the odd-numbered bits are stuffed into one network channel while the even-numbered bits are inserted into another channel. If, for example, the even packet of a sample is lost, it can be recovered through interpolation using the odd packet of the sample. Channel coding imposes extra processing overhead and synchronization delay, and it only works with non-recursive encodings.

2.1.4 Interleaving

When the chosen unit size is smaller than the packet size and end-to-end delay is unimportant, interleaving is a useful technique for reducing the effects of loss [21]. Units are resequenced before transmission, so that originally adjacent units are separated by a guaranteed distance in the transmitted stream and returned to their original order at the receiver. Interleaving disperses the effect of packet losses [8]. As illustrated in Figure 2.4, if, for example, we have 4 units per packet, then after interleaving at the sender, the first packet would contain units 1, 5, 9, 13; the second packet would contain units 2, 6, 10, 14 and so on. In this example the basic unit fed into the interleaving algorithm at the sender is one whole frame in the data stream, therefore we give it the name "*whole-interleaving*".

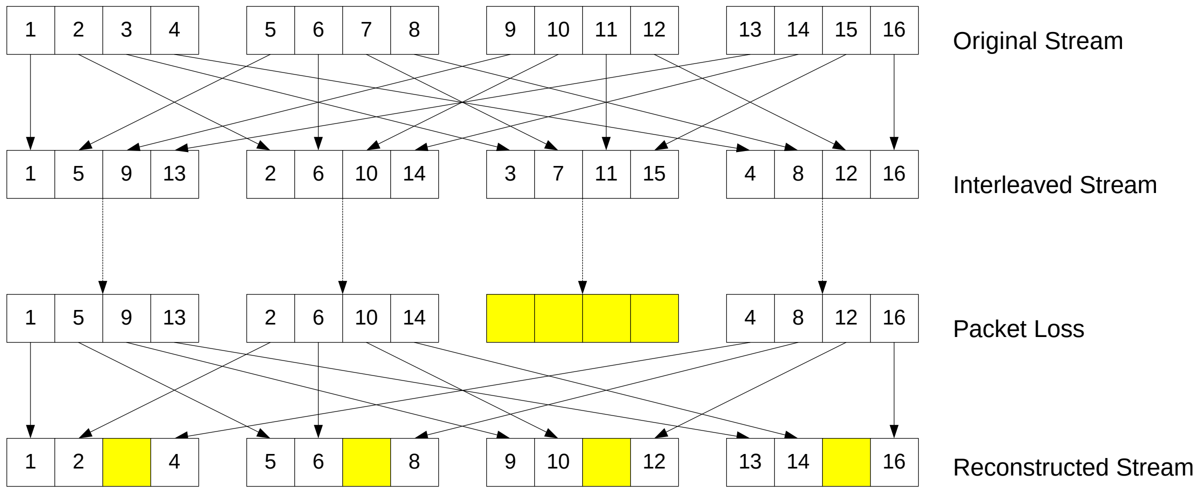


Figure 2.4 Interleaving units across multiple packets

From Figure 2.4 we can see that a single packet loss results in multiple small gaps in the reconstructed stream, as opposed to one big gap if no interleaving is applied are the sender. Interleaving has been employed in some Mbone audio tools which typically transmit packets that are similar in length to phonemes in human speech, and loss of a single packet will therefore have a large effect on the intelligibility of speech. In those audio applications, if the loss is spread out so that small parts of several phonemes are lost, it becomes easier for people to mentally patch-over this loss [22]. We can expect the same effect on interleaved video stream, which is the ultimate goal of this thesis.

Interleaving can be applied to most audio coding schemes, and those schemes may also be modified to improve the effectiveness of interleaving. As with all the other error recovery techniques, interleaving has its own disadvantage. The process time of interleaving algorithm increases latency, and therefore limits the use of this technique for interactive applications, although it performs well for non-interactive use. For audio applications that do not need to be compressed before transmission, interleaving does not increase the bandwidth requirements of the stream. However, for video streams that are usually compressed before sending, some amount of extra bandwidth is needed, and the amount depends on the characteristics of the video encoding schemes and parameters chosen to compress the video streams.

2.2 Receiver Based Error Concealment Techniques

Receiver based error concealment techniques can be initiated by the receiver of the multimedia stream and do not require assistance from the sender. These techniques are of use when sender based recovery schemes fail to correct all loss, or when the sender of a stream is unable to participate in the recovery [8].

The task of error concealment schemes is to use approximation or interpolation techniques to produce a replacement when a packet is lost, and the replacement should be similar to the original. In this way the loss in the media streams is disguised. Under the situation of relatively low lost rates (< 15%) and small packets (4-40ms), these techniques are feasible to use.

However, receiver-based techniques break down when the loss length approaches the length of a unit in the data streams. For example, in an audio stream if the loss length goes up to the length of a phoneme (5–100ms), whole phonemes may be missed by the listener and there is no way for the error concealment techniques to disguise this loss. For this limitation, error concealment schemes usually cannot perform alone, but rather work in tandem with the sender-based repair techniques.

Figure 2.5 shows a taxonomy of receiver based recovery techniques. Error concealment is split into three categories: insertion-based schemes, interpolation-based schemes and regeneration-based schemes, which will be discussed in the following sections.

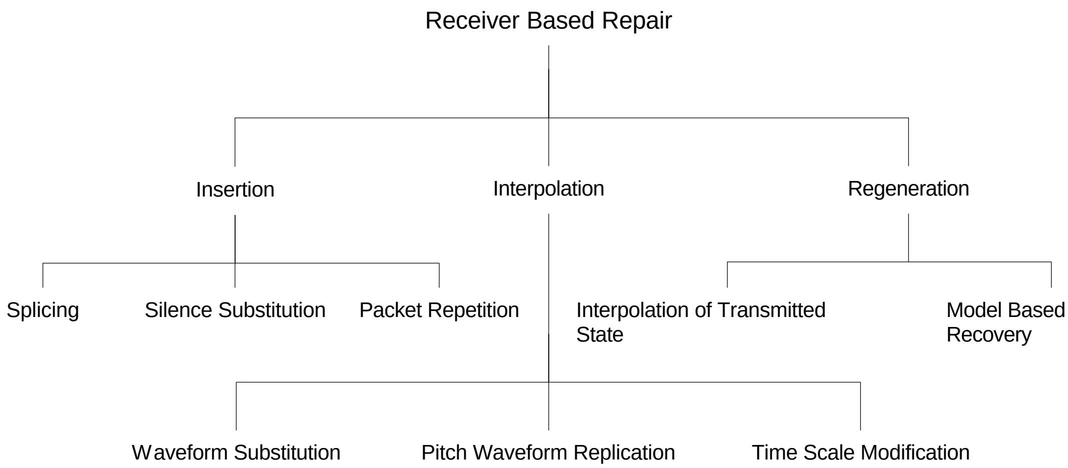


Figure 2.5 A Taxonomy of Error Concealment Techniques

2.2.1 Insertion Based Repair

Insertion based schemes repair losses by inserting a simple fill-in packet. The simplest case is splicing where a zero-length fill-in is used. An alternative is silence substitution where a fill-in with the duration of the lost packet is substituted [8], to maintain the timing of the stream. By using noise or repeating the previous packet as the fill-in, better results can be achieved. Those schemes have primarily been applied to audio applications. In our study, repetition is used combined with interleaving, to form a recovery scheme that has a better performance than any one of them alone.

- Splicing: Lost units can be concealed by splicing together the units on either side of the loss. No gap is left due to a missing packet, but the timing of the stream is disrupted. It has been evaluated to have typically poor performance, with intolerable results at loss rates as low as 3%.

- Silence Substitution: This scheme fills up the gap left by a lost packet with silence in order to maintain the timing relationships with the surrounding packets. It is only suitable for interleaved audio with low loss rates less then 2%, and short packets less than 4ms. As packet sizes increase, the performance of this scheme degrades rapidly, and with the packet size at 40ms, which is common in network audio conferencing tools, the quality becomes unacceptable. Despite this disadvantage on performance, silence substitution is widely used primarily because of its simplicity.

- Noise Substitution: To make silence substitution perform better, noise substitution is employed. The idea of this scheme is to improve performance by filling in background noise in the gap left by a lost packet, instead of just silence. As an extension for this, a proposed future revision of the RTP profile for audio/video conferences [8] allows for the transmission of *comfort noise* indicator packets. This allows the communication of the loudness level of the background noise to be played, resulting in better fill-in information to be generated.

- Repetition: In this scheme, lost units are replaced by previous consecutive units. It performs reasonably well, while has low computational complexity. In this thesis, we combine video repetition with our interleaving scheme to achieve a better perceptual quality.

The insertion based repair techniques are easy to implement, but typically have poor performance under moderate loss, with the exception of repetition, which, under some circumstances, can have good performance.

2.2.2 Interpolation Based Repair

Interpolation based repair techniques produce the replacement for a lost packet by interpolating from packets surrounding the loss. They account for the changing characteristics of the signal as an advantage of these techniques over insertion based techniques.

- ❖ Waveform Substitution: This scheme uses audio before, and optionally after, the loss to find a suitable signal to cover the loss [8]. Goodman *et al*. [23] studied the use of waveform substitution in packet voice systems. Both one- and two-sided techniques use templates to locate suitable pitch patterns at either side of the loss. The two-sided schemes, with interpolation, generally perform better than the one-sided schemes where the pattern is repeated across the gap, and both work better than silence substitution and packet repetition.

- ❖ Pitch Waveform Replication: This scheme is a refinement on waveform substitution by using a pitch detection algorithm at either side of the loss. As a result, this technique was found to work marginally better than waveform substitution.

- ❖ Time Scale Modification: It allows the audio at either side of the loss to be stretched across the loss. A scheme is proposed by Sanneck *et al.* [24] that finds overlapping vectors of pitch cycles on either side of the loss, offsets them to cover the loss and averages them where they overlap. Time scale modification is computationaly intensive, but the technique appears to work better than both waveform substitution and pitch waveform replication.

2.2.3 Regeneration Based Repair

This scheme has been used for audio streams only. It uses the knowledge of the audio compression algorithm to derive codec parameters, such that audio in a lost packet can be synthesised [8]. Regeneration based repair techniques are codec dependent, but they perform well because of the large amount of state information used in the repair. They are also computational intensive.

- Interpolation of Transmitted State: This scheme is used for the kind of codecs that are based on transform coding or linear prediction, because for those codecs it is possible that the decoder can interpolate between states. This has an advantage that there are no boundary effects due to changing codecs and the computational load remains approximately constant.

- Model-Based Recovery: In this scheme, the speech on one, or both, sides of the loss is fitted to a model that is used to generate speech to cover the period loss [8]. There is at least one reason that makes this technique work well: the small size of the interleaved blocks ensures that it is highly possible that the last received block has the speech characteristics that are relevant.

2.3 Retransmission Based Technique

Retransmission-based error control schemes are sometimes also referred to as ARQ (Automatic Repeat Request) schemes or *closed-loop techniques* [4]. A typical interactive audio application has a data rate of about 64 Kbit/s and an interactive compressed video usually has data rates of 1.5 Mbits/s (MPEG-1) [1]. For such interactive multimedia applications that have tight latency bounds and end-to-end delays, the extra delay imposed by the use of retransmission is often not acceptable. For this reason, retransmission based recovery is typically not employed for interactive applications such as audio and video.

However, a few attempts to adapt retransmission to the needs of loss-tolerant, delay-sensitive traffic have been made. The idea behind these schemes is to provide a partially reliable transport service, which does not insist on recovering all, but just some of the packet losses, thus providing higher throughput and lower delay than reliable transport service. Unlike TCP which provides reliability by retransmitting packets until they are acknowledged, a transport protocol that provides a partially reliable service must first detect the lost packet and then decide whether to recover it or not. Depending on which side that will do the detection and recovery, two basic techniques are possible: sender based and receiver based loss detection and recovery.

Dempsey and Liebeherr were the first to investigate retransmission for multimedia applications for the case of a unicast interactive voice transmission over local area networks [17, 18]. Given the round trip times in the order of 10 msec and inter-packet gaps of 20 msec, Dempsey demonstrated that a playout delay at the receiver of about 100 msec will obtain an acceptable quality voice transmission.

The previous example evaluated retransmission based recovery for audio applications, with video applications, where the bit rates is much higher, there is a problem of rate control for multiple receivers. One way to allow for rate control and scalability is to use a hierarchical coding schemes [1], where the signal is encoded in a base layer that provides a low quality image and additional complementary layers for improved image quality [1]. Each receiver needs to receive at least the base layer, and further choose to receive as many layers as the bandwidth available along the path allows. This approach is also referred to as *receiver-driven layered multicast* (RLM) [19, 20].

Compared with other error recovery or concealment techniques, retransmission based error control has its advantages as portable and low overhead. The disadvantages are the latency penalty for recovering packet losses and the extra bandwidth needed for retransmissions and acknowledgements.

2.4 Hybrid Error-Control Schemes

All of the error recover techniques have their advantages and disadvantages. Several researchers have been studying the effects of combining some of those techniques together, and some hybrid schemes have been proposed to maximize the quality of the data transmitted over the Internet. For example, the receiver based error concealment technique combined with some sender bases error recover techniques can be treated as a kind of hybrid scheme. Another good example is combining retransmission (ARQ) and FEC, referred to as hybrid ARQ type II. In this ARQ/FEC scheme, no redundant data is sent with the first transmission, but parity data are sent when a retransmission is required. This approach is very bandwidth-efficient for reliable multicast to a large number of receivers [1]. Error recovery by multicast retransmission of the original data packets requires retransmission of all lost packets. On the other hand retransmission of a single parity packet allows all receivers to recover their lost packet.

2.5 MPEG Encoding

Multimedia streams are compressed before being transmitted over the network. MPEG is a popular compression standard for this task. The MPEG standard was developed by the ISO/IEC JTC1/SC29WG11, a working group within the International Standards Organization, for compressing motion pictures and multimedia [28]. MPEG standards contain MPEG-1, MPEG-2, MPEG-4, MPEG-7 etc, each with different data rates and target applications. For example, MPEG-1 is intended for intermediate data rates on the order of 1.5 Mbit/s, while MPEG-2 is intended for higher data rates (10 Mbit/s or more), and MPEG-4 is intended for very low data rates (about 64 Kbit/s or less). MPEG-1 is used as the compression standard in this thesis, as consistent with other works that have applied error recovery techniques to video. For this reason, this section is concerned primarily with MPEG-1.

In MPEG, a video stream, also called a *sequence*, is simply a series of pictures taken at closely spaced intervals in time, as illustrated in Figure 2.6. MPEG defines a *group of pictures* (GOP) structure in which each GOP starts from an I- (intra) frame. The *macroblock* is the basic building block of an MPEG picture. It consists of a 16 x 16 sample array of luminance (grayscale) samples together with one 8 x 8 block of samples for each of two chrominance (color) components. The MPEG picture is not simply an array of macro blocks. Rather, it is composed of slices, where each slice is a contiguous sequence of macro blocks in raster scan order, and those macro blocks in a given slice have the same shade of gray. The structures of the sequence, group of picture, pictures, slices, macroblock and their relationships are illustrated in Figure 2.6.

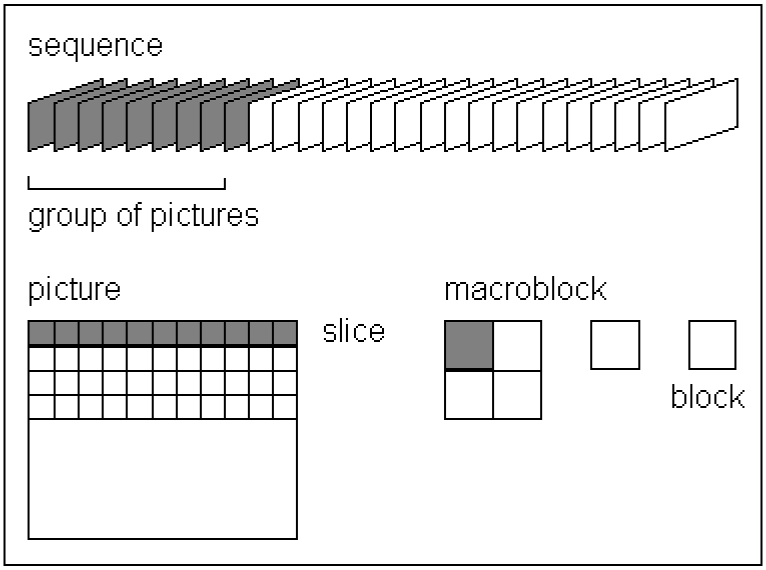


Figure 2.6 Basic Concepts in MPEG-1

Except for the special case of a scene change, the sequence of pictures tends to be quite similar from one to the next. The compression techniques used by MPEG that take advantage of this similarity or predictability are usually called interframe techniques. Compression techniques that only use the information in the current picture are called intraframe techniques. Both interframe and intraframe techniques are used in MPEG video compression algorithms. As a result, MPEG has four different types of pictures, or frames, which are compressed using different techniques.

- *I-frames (Intra-coded frames)* are self contained, coded independently, entirely without reference to other pictures. I frames are points for random access in MPEG streams. I-frames use 8 x 8 blocks defined within a macro block, on which a Discrete

Cosine Transform (DCT) is performed. The compression rate of I-frames is the lowest within MPEG.

- *P-frames (Predictive-coded frames)* require information of the previous I-frame and /or all previous P-frames for encoding and decoding. The coding of P-frames is based on a case of *temporal redundancy*, where areas in successive images often do not change at all or may be shifted. P-frames have a better compression rate than I-frames.

- *B-frames (Bi-directionally predictive-coded frames)* require information of the precious and following I- and/or P-frames for encoding and decoding. The highest compression rate is achieved by using these frames. A B-frame is defined as the difference of a prediction of the past image and the following P- or I-frame.

- *D-frames (DC-coded frames)* are intraframe-encoded. They can be used for fast forward or fast rewind modes. D-frames consist only of the lowest frequencies of an image. They only use one type of macro block and only the DC-coefficients are encoded. D-frames are not used in our study, since fast forward and fast rewind features are not needed in this thesis.

I frames must appear periodically in a video stream. The encoder will cycle through each frame and decide whether to do I, P, or B encoding. The order will depend on the application, but roughly within every twelve frames, an I-frame must be created. A GOP starts from an I frame, followed by several P- or B- frames. Figure 2.7 illustrates the structure of a GOP and the coding dependencies among those I-, P- or B-frames. In this example, the frame pattern IBBPBBPBB is used.

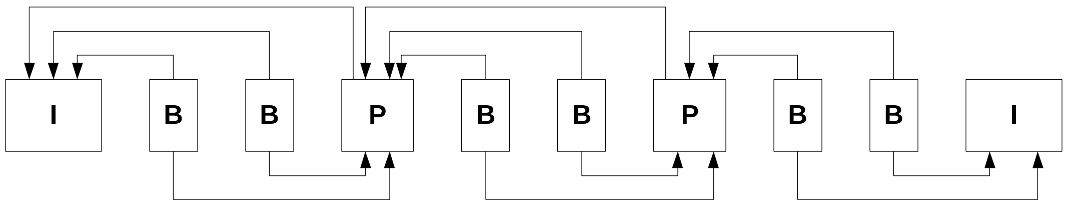


Figure 2.7 Coding Dependency within a GOP

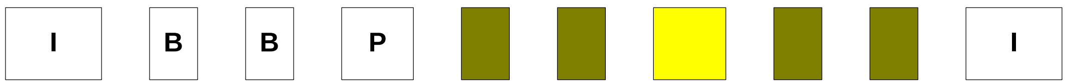


Figure 2.8 Results of loss of the second P frame in a GOP

During transmission of a video stream compressed by MPEG over network, if one frame is lost, the decoding at the receiver will have different results, depending on the type of the lost frame. If the lost frame is a B-frame, then only a small gap results in the video stream, since no other frames are decoded dependent on B frame. However, if the lost frame is an I- or P-frame, not only this frame is lost, other P- or B-frames which are encoded based on this lost frame will also be lost, leaving a big gap in the video stream. The worst case is the loss of an I-frame, which will result in all the frames in one GOP being lost. In this case, the perceptual quality of the received stream will degrade dramatically. The loss of the first P-frame in a GOP will make all the frames after the first I-frame lost, and even the loss of the second P-frame will result in the loss of 5 consecutive frames, as shown in Figure 2.8.

To solve the propagation characteristic of frame loss in video streams, we use interleaving to spread out the consecutive loss results from the loss of only one I- or P-frame. The details of our approach will be discussed in Chapter 3.

2.6 Network Properties

In this section, we discuss the effects of network properties on multimedia over the Internet, especially concentrating on loss characteristics and service models. We start by presenting the service provided in the current Internet for multimedia applications, followed by the discussion of loss characteristics for IP-based networks.

2.6.1 Services in Current Internet

Most multimedia applications require a service with high throughput, low network delay and low jitter. In order to meet these requirements, it is possible to use a network service that directly provides the reliability for multimedia applications without additional error control mechanisms in the transport layer. This can be achieved by reserving network resources, or by dimensioning the network in a way that the residual error probability is sufficiently small [1]. However, today, neither the Internet nor any transport protocol provides such a service. TCP provides resequencing, flow and congestion control, and recovery of all lost packets, but at the cost of increased delay and reduced throughput. UDP offers no guarantees, but introduces minimal delay and reduction in throughput.

Although multimedia applications can tolerate a limited amount of data loss, studies show that perceptual quality for packet audio or video traffic without any form of loss repair, is degraded at loss rates as low as 1%-5%, and the limits to comprehensibility are reached at around 10% loss [25]. However, experiments have shown that packet drop rates between 7-15% on the Internet are common, with occasional drop rates as high as 50% [1]. UDP cannot respect the loss tolerance of most multimedia applications.

Furthermore, applications using UDP may flood the network and/or the receiver with packets, because UDP has no congestion control and flow control.

2.6.2 Loss Characteristics in IP-based Networks

IP networks offer a datagram service with best-effort service having no guarantees on loss rate, delay and in-sequence delivery. To assess the suitability of the existing IP best effort service for supporting audio-visual applications, it is of interest to have detailed knowledge about typical quality impairments.

Many researchers have been investigating the loss characteristics of the current Internet. Handley [11] shows in his examination of MBone performance that 50% of the tested receivers have a mean loss rate of about 10% or lower, and that around 80% of receivers has some interval during the day when no loss was observed. However, 80% of tested sites report some interval during the day when the loss rate was greater then 20%, which is generally regarded as being the threshold above which audio without error recovery becomes unintelligible, and about 30% of sites reported at least one interval where the loss rate was above 95% at some time during the day. Another result shows that 80% of all reports give loss rates of less than 20%. The measurement in [13] shows a relatively high loss probability in the access area and rather low loss probabilities in the backbone area.

In such scenarios where loss rates are relatively low (< 20%), error control schemes are attractive. However, high loss rates limit the effectiveness of these error recovery schemes.

Chapter 3 Interleaving Approaches and Implementations

In this chapter, we discuss our interleaving approaches in detail. We first describe the effects of frame loss on video streams over a network, then our whole-interleaving approach followed by the discussion of our partial-interleaving approach, and finish with our implementation of both approaches.

3.1 Effects of Packet Loss On a Video Stream

During the transmission of packet video over the Internet, packet loss results in gaps in the stream, which degrades the perceptual quality of the video. In the case of frame losses in a video stream, the video becomes less smooth, and end users will notice some pauses in the video stream. In order to keep the temporal synchronization in a video stream, especially for 2-way video, and also to keep synchronization with the parallel audio stream, a lost frame in a video stream is usually replaced by the immediately previous frame. However this does not make the video smoother.

Small amounts of loss, especially with a fast transmission rate, will often be tolerable by end users since it is highly possible that the information in the lost frame is similar to the adjacent frames. However, in case of multiple consecutive frame loss, the video “pauses” longer and the information in those lost frames is lost.

Study has shown that majority of the packet loss events on the Internet appear to be single losses, as a result observed by Gerek and Buchanan [26], illustrated in Figure 3.1. However, consecutive loss causes more severe damage to perceptual quality of video than does single loss. Furthermore, for video streams compressed using inter-frame encoding techniques, such as MPEG, a single loss may result in multiple consecutive losses due to the propagation of loss.

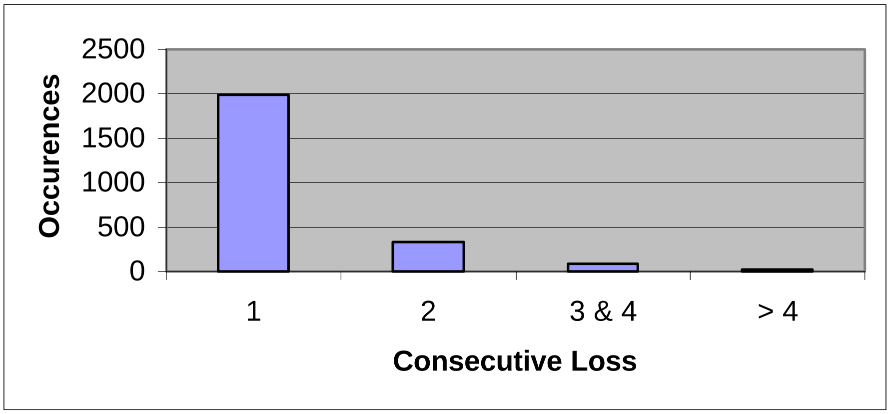


Figure 3.1 Consecutive Loss Distribution

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

MPEG is often used for compression of video streams. In MPEG-1, a sequence of frames is divided into groups of pictures (GOP). For example, a typical GOP pattern with 9 frames is: IBBPBBPBB. Within one GOP, B-frames are encoded depending on I- and P-frames, and P-frames are dependent on an I-frame and/or another P-frame. One single frame loss sometime results in a sequence of frames in one GOP being "lost", since these frames may be decoded dependent on the lost frame. This propagation of loss in MPEG can seriously degrade the perceptual quality of a video stream.

The basic idea behind interleaving is to spread out one big gap in the media stream into several small gaps, which are separated by guaranteed distance. In this way the effect of the loss of multiple consecutive frames will be ameliorated, and the perceptual quality will be increased. Previously interleaving has been applied to audio streams [8]. In this thesis, we apply it to video streams. One important factor in interleaving is the granularity on which the interleaving algorithm should be operating. In a video stream, the interleaving could be a GOP, or several frames, as in audio interleaving. High compression rates in MPEG are gained by using the inter-frame encoding, the techniques which take advantage of the similarity among consecutive frames. Therefore, a finer granularity of interleaving will result in less overhead. In this thesis, instead of interleaving based on several frames, we propose a technique to interleave on the basis of one frame, which we call *whole-interleaving*. We implement this approach and confirm its effects on error recovery of video stream by simulation and user study. Furthermore, since each frame consists of macro blocks, which are the basic MPEG structure, we propose another interleaving scheme called *partial-interleaving*, in which each frame is cut into groups of macro blocks, and a group of macro block is used as the basic unit for interleaving. Partial-interleaving is also implemented in this thesis.

Next we present our whole-interleaving and partial-interleaving approaches in detail, followed by a discussion on the overhead due to video interleaving.

3.2 Whole-Interleaving

In our whole-interleaving approach, the whole frame is used as the basic unit of interleaving. At the sender, frames in a video stream are first interleaved, with the original consecutive frames being separated by a specific distance that is given by the interleaving algorithm. After arriving at the receiver, frames are then reconstructed to their original order. If consecutive loss occurs in the interleaved stream during transmission, or as a result of single loss propagation, after reconstruction at the receiver, a big gap in the stream caused by the consecutive loss or propagated loss will be spread out into several small gaps that are separated by the distance value. A parameter to interleaving is the distance the smaller gaps are separated by the interleaving scheme. For example, with distance=2 , GOP size = 9 and a GOP pattern of IBBPBBPBB, the interleaving stream will be looking like the following sequence, in which a number indicates the position of one frame in a video stream:

1 3 5 7 9 11 13 15 17 2 4 6 8 10 12 14 16 18

And with distance=5, the interleaved frame will appear like the following

1 6 11 16 21 26 31 36 41 2 7 12 17 22 27 32 37 42 …

In above sequences, each underscore line indicates a group of 9 frames generated by adding the value of *distance* to the number of a previous frame in the stream. This method is repeated until there are 9 frames in the group. After that, a new group starts and the number of next frame is then picked from the smallest available frame number. The next frame is obtained by adding the value of *distance* to the number of current frame, until the new group fills up with 9 frames. The number "9" is chosen here because it is the value of GOP size. The reason for doing this is that in the interleaved stream, even if all 9 frames in a GOP are lost, in the reconstructed stream, those losses are guaranteed to be non-consecutive and separated by the value of *distance*.

Our interleaving approach is combined with repetition error-recovery technique, in which a lost frame is recovered by repeating the previous consecutive frame. In the reconstructed stream, repetition is applied to a lost frame. For example, say we use the GOP pattern IBBPBBPBB, Figure 3.2 shows the situation of one I frame lost in a video stream without interleaving, and Figure 3.3 demonstrates whole-interleaving with a distance=2 and the result of the same I frame lost within the same video stream.

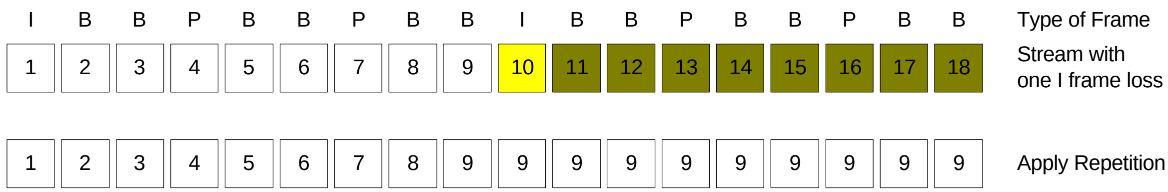


Figure 3.2 Effects of a lost I-frame with no interleaving

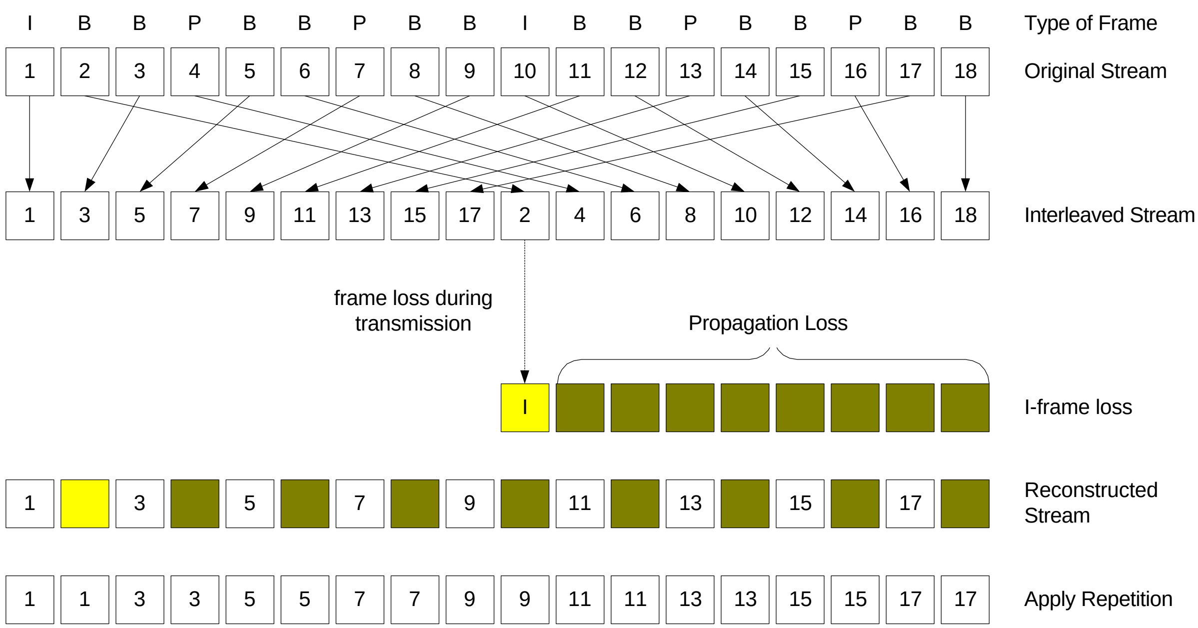


Figure 3.3 Effects of a lost I-frame with whole-interleaving /distance=2

It may seem that whole-interleaving with higher distance values may have a more even distribution of loss than with lower distance values (for example, distance=5 vs. distance=2), since lost frames are separated with further distance. However, for some loss distributions, using interleaving, one big gap in the stream will be spread out, but several single losses that are non-consecutive in the interleaved stream may appear to be consecutive in the reconstructed stream, as shown in Figure 3.4 with distance=2, which may negate part of the error recovery effects of interleaving.

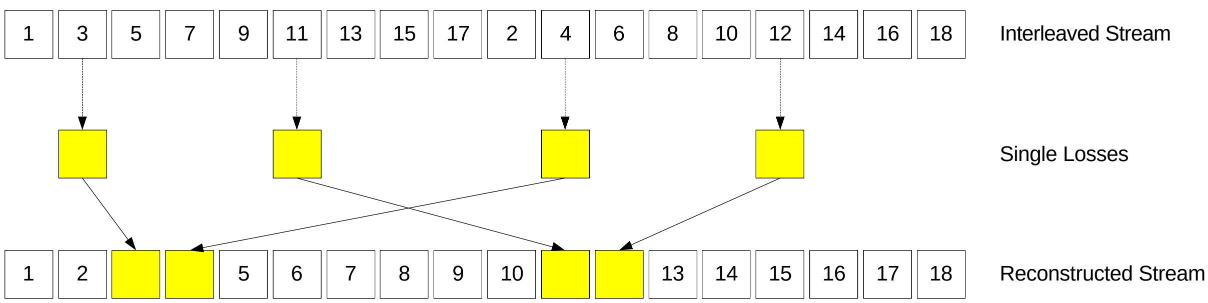


Figure 3.4 A Special Case of Single Losses in the Interleaved Stream, distance=2

This phenomenon could happen in any interleaved stream with any given value for distance. The effects on perceptual quality mostly depend on the kind of lost pattern in the video stream. In our user study, we used the random number generator in Perl to simulate the lost frames in a stream, and in both test cases with error rates of 5% and 20%, whole-interleaving with distance=2 shows a better performance than with distance=5. This will be covered in detail in Chapter 4.

3.3 Partial-interleaving Approach

In MPEG, the basic building block of a frame is the macroblock. It is possible to interleave groups of macro blocks, also called sub-frames, instead of the whole frame. The advantages for this approach are a) less bandwidth overhead than whole-interleaving, and b) in case of frame loss, the loss in one frame is spread out, so that only part of some frames are lost, not the whole frame. Figure 3.5 shows sample 4-way partial-interleaving with one frame lost and the reconstructed frame. Partial-interleaving is combined with repetition error recovery, in which the lost part of one frame is replaced by the same part in the previous frame.

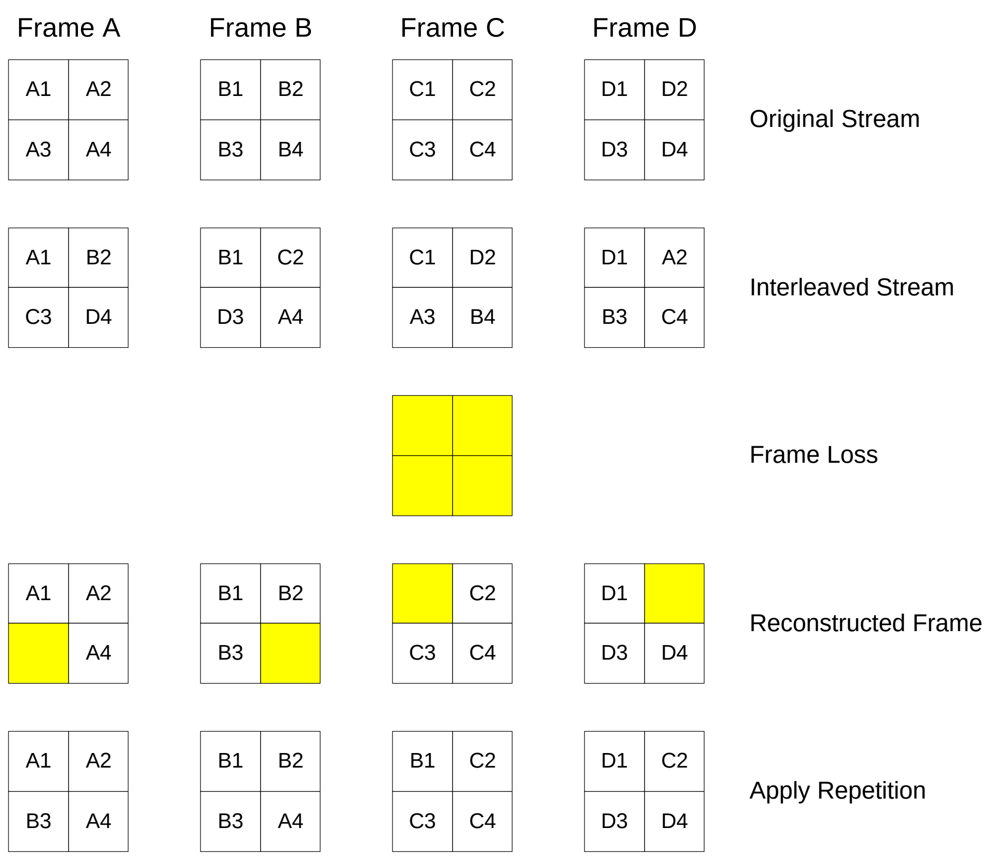


Figure 3.5 A Sample 4-way Partial Interleaving

The sample shown in Figure 3.5 is 4-way partial interleaving, which means each frame is divided into 4 groups of macro blocks or sub-frames. We define the *interleaving factor* as the number of sub-frames in each frame. A higher interleaving factor has the advantage of a lower overhead, but at the cost of increasing the processing time. The total number of sub-frames interleaved can be computed as the number of frames in the video stream, multiplied by the interleaving factor. A higher interleaving factor results in more sub-frames to be processed, thus more processing delay.

In MPEG-1, getting only part of a frame means that the whole frame cannot be reconstructed. With repetition techniques, it is possible to reconstruct these frames but the algorithm for doing so may be too complicated to be processed in real-time. However, there are solutions that are possible to solve this problem of partial interleaving. First, the partial interleaving algorithm can be built into hardware, and this move from software to hardware may make it possible to meet the real-time requirement for video streams. Changing the compression standard from MPEG-1 to MPEG-4 may also help, since MPEG-4 is content or object-based which makes it easier to reconstruct a frame with partial information.

3.4 Bandwidth Overhead

Although whole-interleaving does not add redundant data directly to the video stream, the video compression may be less effective (adding more bandwidth) because it separates originally consecutive frames with a larger distance (as shown in Figure 3.6 with distance=2), and therefore the similarity between the consecutive frames in the interleaved streams becomes less than in the original frames. The key aspect of moving picture compression is the similarity between frames in a sequence. As a general rule, less activity (smaller differences) between consecutive frames leads to better compression and smaller file sizes, and vice versa. The ideal case is that there is no motion, thus no difference, and the sequence can be coded as a single picture followed by a few bits for each following picture telling the decoder to simply repeat the frame, this results in the smallest possible file size for a video sequence.

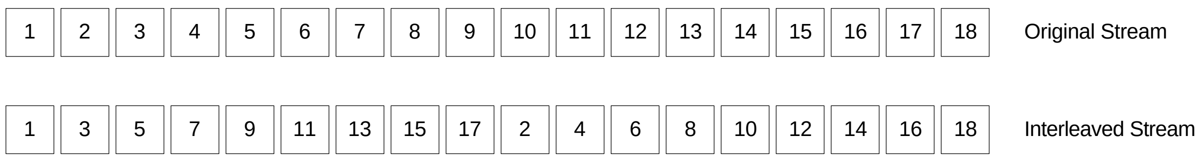


Figure 3.6 Original Stream VS. Interleaved Stream in Whole-interleaving

In whole-interleaving, the larger *interleaving distance* value will result in a larger overhead and bandwidth, since originally consecutive frames are further separated by the value of *distance*.

The same phenomena applies to partial-interleaving, but since it is the sub-frames that are being interleaved, the situation is different. Figure 3.7 shows the original stream and interleaved stream in partial-interleaving, with an interleaving factor of 4. In the interleaved stream, each frame consists of different sub-frames as compared to the original stream. However, as seen from the sub-frame level, the similarity between consecutive sub-frames stays almost the same. For example, for Frame A and Frame B in the interleaved stream, the similarity between sub-frame at the same position, such as A1 and B1, B2 and C2, C3 and D3, D4, is kept the same as in the original stream, except for the pair at the bottom right corner, D4 and A4, which is separated apart by the value of the interleaving factor (in this case, 4).

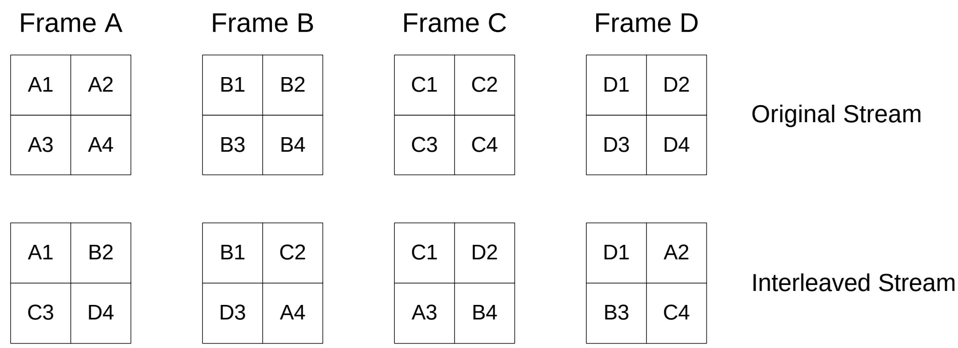


Figure 3.7 Original Stream Vs. Interleaved Stream in Partial-interleaving

In partial-interleaving, the overall fraction of the similarities in the original stream kept the same as in the partial-interleaved stream can be expressed approximately with the following equation:

$$\frac{Interleaving\ Factor - 1}{Interleaving\ Factor}$$

Because of the remaining small fraction of changes of similarities, partial-interleaving still causes bandwidth overhead, but much less than that caused by whole-interleaving, since in the latter, the fraction of changes of similarity equals 1, meaning all the similarities are changed. From the above equation, as the interleaving factor increases, the fraction increases, and the amount of bandwidth overhead decreases.

In this thesis, we choose interleaving factors of 3, 6, 12, 15 and 30. As an example, the relationship between the interleaving factor and the overhead of the file size for movie clip simpson.mpg, which is part of a cartoon recorded from TV, is as follows:

Interleaving Factor	Overhead of file size
3	5.1%
6	3.9%
10	2.1%
12	1.7%
15	0.3%

3.5 Implementation

In this section, we first give a brief introduction to the simulation tools used to implement interleaving on video. Then, we give details on the implementation of whole-interleaving, followed by the partial-interleaving implementation. Finally, we discuss the methods we use to simulate loss in the video streams.

3.5.1 Simulation Tools

Three MPEG-1 tools are used in our simulation, namely *mpeg_encode*, *mpeg_play* and *mpeg_stat*. They are used in both the whole-interleaving and partial-interleaving approaches. Another two utilities, *pnmcut* and *pnmpaste*, are only used in partial-interleaving to operate on slices of frames. In this section we only give a brief introduction to those tools. For detailed information please refer to Appendix A.

Most MPEG tools we used are from UC Berkeley, developed by researchers and students in the Berkeley Plateau Multimedia Research Group [29]. *mpeg_encode* is used to encode MPEG-1 bitstreams, while *mpeg_play* is to decode and display MPEG-1 encoded streams on systems running X11. *mpeg_stat* decodes MPEG-1 encoded bitstreams and collects varying amounts of statistics.

In *mpeg_encode*, the quality that a video encoded is controlled by a Q-Scale (quantization scale) factor in a parameter file defined by a user. The quantization scale values give a trade-off between quality and compression. Using different Q-Scale values has little effect on encoding speed. MPEG quality numbers are defined as peak signal-to-noise ratios in *mpeg_encode*, and the larger numbers give better compression, but worse quality. The following equation shows the definition of mpeg quality numbers:

$$20 \log_{10} \frac{255}{\sqrt{MSE}}$$

Where MSE is the mean square error.

pnmcut and *pnmpaste* are two utilities used by partial-interleaving. *pnmcut* reads a portable anymap frame (pnm) as input, extracts the specified rectangle, and produce a portable anymap as output. As introduced in the mpeg_encode section, we use ppm as our source frame format in a video stream, and ppm is a subset of pnm. Thus, any tool that operates on pnm will also be suitable for ppm. A pair of x, y coordinates is given by the user to indicates a starting point of the cut operation, while a pair of width and height coordinates defines the size of the slice that is going to be cut off from the frame.

pnmpaste reads two frames in portable anymap format as input, inserts the first frame into the second at the specified location, and produces a format in portable anymap format the same size as the second as output. This tool is most useful in combination with pnmcut. In our implementation of partial-interleaving, this utility is used at the resequence step at the sender and the repetition step at the receiver, in the latter case a lost part of a frame is recovered by pasting the sub-frame at the same position as the previous frame.

At the sender, during the interleaving process, *pnmcut* cuts each frame in a video stream into sub-frames, defined by the interleaving factor, and *pnmpaste* is then used to paste the sub-frames to certain positions defined by the partial-interleaving algorithm. When the video stream is reconstructed at the receiver, *pnmcut* cuts each frame into same amount of sub-frames, and *pnmpaste* pastes those sub-frames back to their original position.

3.5.2 Whole-interleaving Implementation

The video clips used in this thesis are recorded from TV and then encoded into mpeg-1 format. Perl scripts are written to do the whole process automatically.

❖ Step 1: First we use mpeg_play to decode an .mpg file into a sequence of ppm frames, and treat those frames as source frames of a video clip. Also, we keep each video clip about 20 seconds long, so that the total length of all the video clips in the user study on perceptual quality will not be too long.

❖ Step 2: Next, these ppm frames are interleaving using our whole-interleaving approach. This process can be treated as switching the columns and rows of a matrix. The original matrix is a GOP_size by distance, representing the original stream, with the distance as the number of columns and GOP_size as the number of rows. The columns and rows of the original matrix are then switched and a new matrix, representing the interleaved stream, the size Distance by GOP_size. For example, if we select distance = 2, GOP pattern as IBBPBBPBB, then the group size is 9. The whole interleaving process is depicted in Figure 3.8.

❖ Step 3: We then map the sequence of .ppm files into a sequence of video frames, using the GOP pattern as IBBPBBPBB, based on a frame's position in the sequence.

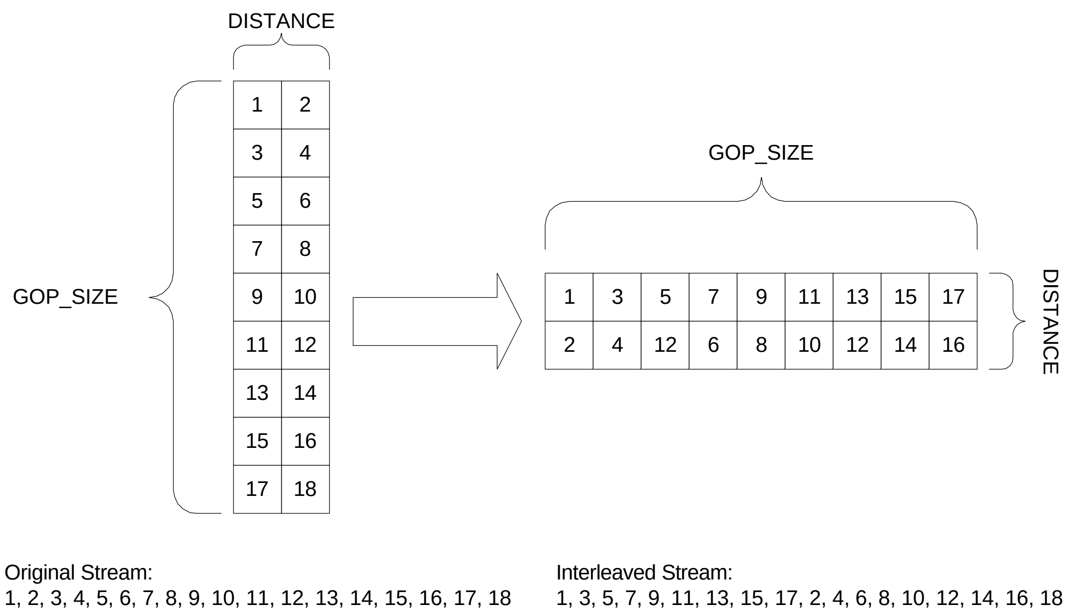


Figure 3.8 Process of Interleaving Frames in Whole-Interleaving

❖ Step 4: Next, we apply a simulated loss rate to the video stream. We carefully choose the loss rate based on the work of by Gerek and Buchanan [26]. They gathered the data of 102 network data transmissions over the Internet across the USA and New Zealand [26]. 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 3.9 shows the loss rate distribution.

We can see most loss rates are less than 5% or greater than 20%. However, if the loss rate of a video stream is greater than 20%, the perceptual quality of the video becomes so poor most users just give up or ask for retransmission. In our work we concentrate on loss rates not greater than 20%. The loss rates used in our simulation are 2%, 5%, 10%, and 20%. The propagation of loss in MPEG is not counted for these loss rates, so the real fraction of lost frames in case of each loss rate is even higher. If the lost frame is

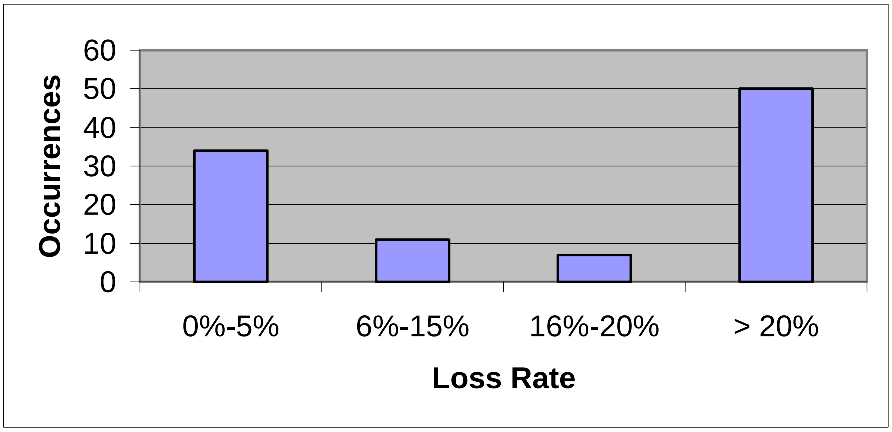


Figure 3.9 Loss Rate Distribution

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.

an I frame, then all the frames in its GOP are lost; if the lost frame is the first P frame, all the frames after the first I frame are lost also; if the lost frame is the second P frame, then all the frames after the first P-frame in the current GOP are treated as lost frames. Only in the case of a loss of a B-frame does no propagation loss occurs, in this case only the B-frame itself is lost.

- Step 5: Next we resequence the interleaved stream into its original order, which is the opposite process of our interleaving algorithm used in the interleaving step. If a frame is treated as lost, the previous frame is then repeated.

- Step 6: The last step is to use mpeg_encode to encode the stream into an MPEG-1 video clip, which is a .mpg file, and use it in our user study.

Compared with the original .mpg file, the resulting .mpg file has some overhead in file size, usually about 15%. Although it is a little bit high, compared with other error recovery techniques, this amount of overhead is tolerable. We can decrease the 15% overhead to almost nothing, yet at the expense of little quality degradation for the encoded video stream by reducing the MPEG quality. The MPEG quality, which is the peak signal-to-noise ratio, is defined in mpeg_encode as:

$$20\log_{10}\frac{255}{\sqrt{MSE}}$$

Where MSE is the mean square error.

Larger quality numbers give better compression, smaller file size, but worse quality. Figure 3.9 shows the relationship between the file size and MPEG quality number. We can see that with the additive decrease in quality (the same as a linear increase in the quality number), the file size decreases exponentially, especially in the range (1, 5) of the MPEG quality number. Human eyes have the limitation to distinguish between the quality of different video clips if the difference is too small. If the perfect video is encoded with quality number 1, and the lower quality video is encoded at quality number 2, users usually cannot decide which one has a better quality. This is further confirmed by our user study on perceptual quality, which will be discussed in detail in chapter 4. However, the difference in the file sizes of these two videos is large. From Figure 3.10 [5], the file size for video of quality number 1 is about 16.5, while the file size for quality number 2 is about 13.7, approximately a 16.9% decrease.

In our work, we encode the non-interleaved stream with quality number 1, and the interleaved stream with quality number 2. Later in chapter 4, our user study shows that with no loss, the two streams with different quality have almost the same evaluation by the users, and under the situation of packet loss, the lower quality interleaved video clips achieve higher evaluation than those higher quality non-interleaved ones. The different quality encoded stream only applies to the whole-interleaving implementation. For partial-interleaving, since the overhead of file size is low enough, the process of lowering the MPEG quality is not needed.

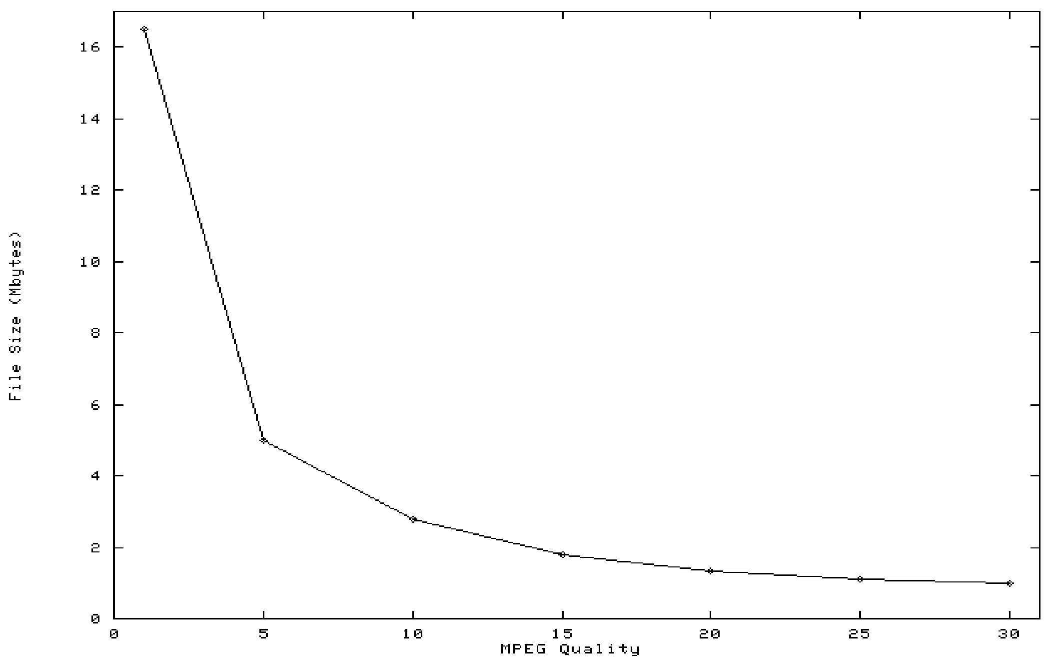


Figure 3.10 MPEG File Size vs. MPEG Quality [5]

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.

3.5.3 Partial-interleaving Implementation

Most steps in partial-interleaving are similar to those in whole-interleaving, except in step 2 and step 5, when the interleaving algorithm is applied to the video stream.

At the sender, in step 2, for whole-interleaving the .ppm streams are resequenced by a matrix conversion operation. In partial-interleaving, resequencing is more complicated. The utility *pnmcut* is used to cut each frame into number of sub-frames, and the number is given by the interleaving factor. Then *pnmpaste* is used to paste sub-frames into the right position. For example, if the interleaving factor is 6, the operation on the 3rd frame in the stream during the interleaving step is shown in Figure 3.11.

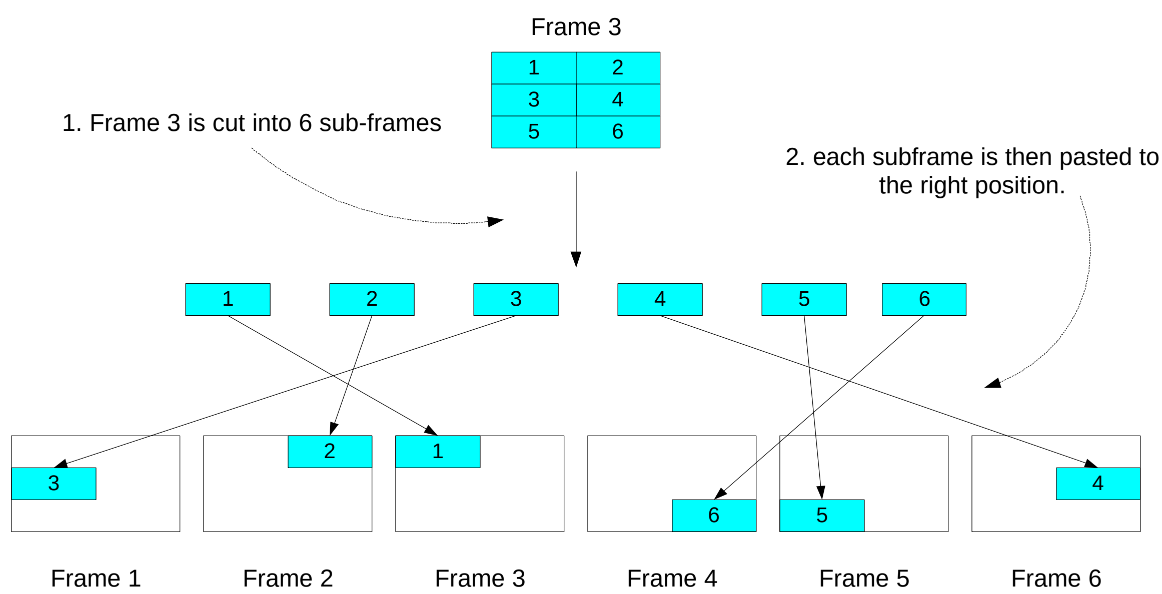


Figure 3.11 Interleaving of Frame 3, with Interleaving Factor = 6

At the receiver, in step 5, which is the step to resequence the stream, the above process shown in Figure 3.10 in then applied in a reverse order, with sub-frames from different frames put back together to form the original frame #3, along with other frames. If for example in Figure 3.11, the frame #2 is lost, then the sub-frame2 of frame#3 is then lost, and is recovered by pasting the sub-frame2 of frame #2.

3.5.4 Simulation of Loss

In our work, we select loss rates of 2%, 5%, 10% and 20%. In both whole-interleaving and partial interleaving, to simplify the problem, an assumption is made that one packet contains only one frame. So the packet loss rate equals the frame loss rate. The simulation of these loss rates on a video stream is done by writing a perl script and using the random generator to generate random numbers with range [0, 1] for each frame in the stream. For example, given a loss rate of 20%, if the number is less than or equal to 0.02, the frame is treated as a lost frame and is discarded.

For example, say we apply a loss rate of 10% to a video clip of a hockey game. The video is 20 seconds long and contains 600 frames. The following is the lost frames decided by a perl script, with each underline indicating a consecutive loss:

5	9	17	28	29	30	43	44	61	82	91
102	111	115	116	120	137	138	139	147	154	160
173	175	180	205	210	216	218	225	236	246	257
262	275	276	292	322	326	340	371	377	380	389
391	395	400	405	428	436	450	458	459	468	478
488	507	534	550	560	562	563	568	573		

The pattern of the loss is illustrated in Figure 3.12. The majority of the loss is single loss, which is about 87.2% of the total loss. Two consecutive losses and three consecutive losses is of frequency 9.1% and 3.6% respectively. The loss pattern in Figure 3.12 is very similar to the pattern in Figure 3.1, which is the result of packet loss on the Internet observed by Gerek and Buchanan [26]. This means our simulation of loss is close to the real loss pattern that is typical on the Internet. For a given loss rate, the same loss pattern is generated and applied to both non-interleaved and interleaved streams.

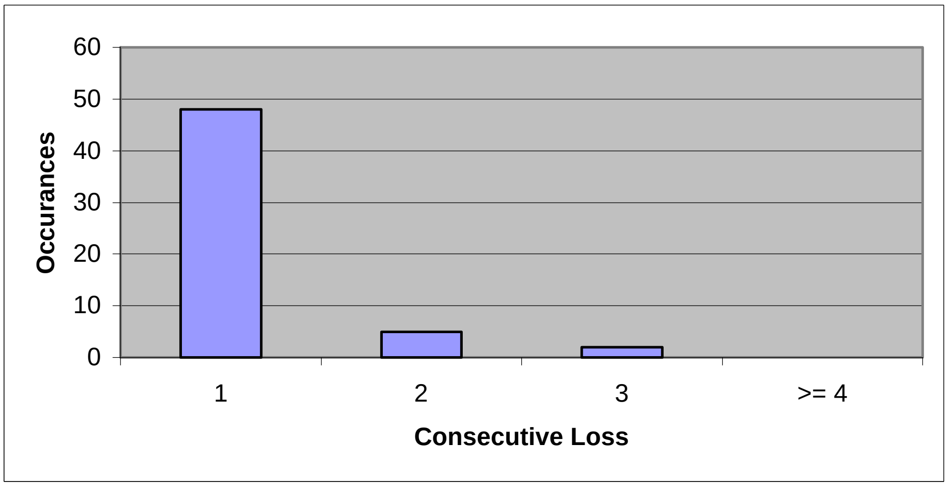


Figure 3.12 Consecutive Loss Pattern is hockey game video, with a loss rate of 10%.

Chapter 4 User Study on Perceptual Quality and Result Analysis

4.1 User Study on Perceptual Quality of Whole-Interleaving

Perceptual Quality (PQ) is the subjective quality of multimedia perceived by the user [27]. Since it is the end-user who will determine whether a service or application is a success, it is vital to carry out subjective assessment of the multimedia quality delivered through these. In this thesis, we evaluate the effects of our whole-interleaving approach on the quality of video streams by a user study on perceptual quality.

Subjective opinions of video quality are formed through the influence of many different factors. In this user study, we focus on the effects of whole-interleaving on the perceptual quality of video streams. So only the factors that are specific to our interleaving are tested, including loss rate, movie type, interleaving distance and MPEG quality of the clips. Other factors that could affect the quality, but are not related to our interleaving algorithm, such as the frame rate, size of the movie screen, and different monitors on different computers, are kept the same throughout the user study. We use 30 frames/sec as the frame rate for each tested video clip, and all the frames in the video streams are of size 320 x 240, as defined in MPEG-1. Also the user studies are carried on one same computer running SuSE Linux 6.4 i686. Only one user takes the test at one time and the same assistant is helping each user throughout all the tests.

One factor tested in the user study is the movie type, as determined by the movie content. In our study, the movie type is determined by the frequency of scene changes and the intensity of object actions among frames. Since the pattern of similarity among consecutive frames varies considerably from one movie type to another, our interleaving technique may have different effects on different types of movies. We test two typical types of movies in our user study, one with lots of scene changes, such as a sport or an action movie, and one with few scene changes or stable object in each frame, such as a news broadcast. Two movie clips, hockey.mpg, a sport video clip, and cnn.mpg, a news clip, are chosen to represent theses two movie types, respectively.

A user study on perceptual quality typically requires the viewer to watch short movie sequences of approximately 10 seconds in duration, and then rate this material [27]. It is not clear whether a 10-second video sequence is long enough to experience packet loss. A longer video clip, 1 minute for instance, may give views more time to make a judgement. However, as we need users to see dozens of video clips with almost the same contents, it is better to keep each clip short enough to avoid making users tired, yet long enough to let the user make an informed decision. In our user study, we encoded each video clip to be about 20 seconds long.

From our past experiences, the user study usually should be limited to about 10 minutes long, in order to prevent the viewers from getting bored and losing the patience to give proper scores. This limits the total number of video clips to be less than 30, since each video is about 20 seconds. However, the problem is that many parameters that could change the effect of whole-interleaving need to be evaluated in our user study, including different loss rates, with or without interleaving, one higher and one lower distance (a parameter defined in whole-interleaving approach), MPEG quality number 1 and MPEG quality number 2, and also two movie types. These parameters are shown as follows:

Loss Rate:	0%,	2%,	5%,	10%,	20%
Interleaved:	Yes,	No			
Distance of Interleaving:	2,	5			
Quality Number:	1 (higher quality),		2 (lower quality)		
Movie Type:	sports		news		

If all there parameters are to be fully evaluated, the number of video clips required goes up to 5 x 2 x 2 x 2 x 2 = 80. 80 movies are too many for an individual user to sit through. We partially combine those parameters and result in the test sequence of totally 24 movie clips as in table 4.1:

Movie Clip	Movie Name	If Interleaved	Loss Rate	Quality Number	Distance
1.mpg	Hockey	No	No Loss	1	-
2.mpg	Hockey	No	No Loss	2	-
3.mpg	Hockey	Yes	2%	2	2
4.mpg	Hockey	No	2%	1	-
5.mpg	Hockey	Yes	5%	2	2
6.mpg	Hockey	No	5%	1	-
7.mpg	Hockey	Yes	5%	2	5
8.mpg	Hockey	No	10%	1	-
9.mpg	Hockey	Yes	10%	1	2
10.mpg	Hockey	Yes	10%	2	2
11.mpg	Hockey	Yes	20%	2	2
12.mpg	Hockey	Yes	20%	2	5
13.mpg	Hockey	No	20%	1	-
14.mpg	CNN	No	No Loss	1	-
15.mpg	CNN	No	No Loss	2	-
16.mpg	CNN	Yes	2%	2	2
17.mpg	CNN	No	2%	1	-

18.mpg	CNN	Yes	5%	2	2
19.mpg	CNN	No	5%	1	-
20.mpg	CNN	No	10%	1	-
21.mpg	CNN	Yes	10%	2	2
22.mpg	CNN	Yes	10%	1	2
23.mpg	CNN	Yes	20%	2	2
24.mpg	CNN	No	20%	1	-

Table 4.1 Test Sequences of Movie Clips for User Study

There are two groups of video clips in the above test sequence, namely hockey and CNN, with each representing a movie type. Movie clips from 1.mpg through 13.mpg are encoded using the same hockey clip, and the rest of the .mpg files are encoded with the same CNN news clip. From this sequence of tests, we are able to make the following comparisons:

- ❖ For each loss rate of 2%, 5%, 10% and 20%, we compare the perceptual quality of movie clips with and without interleaving. For interleaved movie clips, both interleaving and repetition are used to recover the loss, while for non-interleaved streams, only repetition is applied, with the lost frame recovered by simply repeating the previous frame.

- ❖ Besides the comparison between interleaving and non-interleaving, we also make test cases to compare the effects of higher and lower distance values to the whole-interleaving algorithm. Video clips interleaved with distance=5 at a loss rate of 5% and 20% are included in the test sequences.

- ❖ Two types of movies, namely *hockey* and *CNN*, are present in the test sequence in order to compare the effects of interleaving on movies with different scene change frequencies and object action intensities.

- ❖ The comparison of quality between clips encoded with MPEG quality number 1 and MPEG quality number 2 can also be made. The same clips are encoded with quality number 1 and 2 in the cases of no loss rate and a loss rate at 10%.

The numbers in the first column of table 4.1 indicate the actual order of those clips in the user study. Users can also play any clip several times if they wish to do so. The order of the movie sequence requires special attention, since it may change the user study results. The policies used in our study to decide on the order in which to display the clips are:

- ♦ First, each user needs a baseline at the very beginning to compare the following movies, and this baseline should be same to all users. For this reason, in our user study, the first movie shown for both *hockey* video group (1.mpg) and *CNN* video group (14.mpg) is a *perfect* movie, meaning that it is encoded with quality number 1 and has no loss. Users are aware of this information before they start watching all the movie clips.

- ♦ Secondly, users tend to evaluate the quality of the current clip based on the previous 1 or 2 clips that are just played out. If two video clips need to be compared, it is better to put them as close as possible, instead of placing them farther apart. This explains the reason that in our test sequence, the clips with the same loss rate are always consecutive.

- Users may try to guess the internal pattern of the order of test clips, for example, from higher quality to lower quality, and their decision may thus be affected. For this reason the order should also contain some randomness. In our test sequence, video clips with the same amount of loss are distributed randomly in the sequence. And users are told before they start that the order of the movie clips is random with no internal order.

The assessment methodology in our user study is based on the ITU recommended video quality measurement scales, as shown in Figure 4.1. We expand this 5-scale policy to a range of [0, 100]. Users evaluate each movie clip by writing down a score on a printed evaluation form. They first choose a scale, and then within each scale pick up a specific number, also shown in Figure 4.1. This policy gives more a detailed evaluation of the video quality, so if two video clips are within the same quality scale but still have different perceptual quality differences, they can be further distinguished.

Image quality	*ITU Scale*	*Our Score*
Excellent	5	80 – 100
Good	4	60 – 79
Fair	3	40 – 59
Poor	2	20 – 39
Bad	1	0 – 19

Figure 4.1 ITU Recommended Image Quality Measurements Scales and our scores policies [27]

4.2 Evaluation on Perceptual Quality of Partial-Interleaving

In our study, partial-interleaving is proposed and implemented as an alternative way to do video interleaving. We apply partial-interleaving on hockey game clips and CNN new clips, with interleaving factors of 6, 10, 12, 15, 30, respectively.

We compared the perceptual quality of those partial-interleaved groups of video clips and non-interleaved groups of video clips. In the case of packet loss, both groups have poor perceptual quality, though in different ways: For non-interleaved groups of clips, packet losses cause pauses in a video stream, and the transitions between consecutive frames are less smooth; For partial-interleaved video clips, although the transitions between consecutive frames are smoother, each frame with lost parts becomes less smooth, and the overall quality is still harmed. It is unclear if partial-interleaving can improve the perceptual quality of a video stream with packet loss. For this reason, in our study, only whole-interleaving is evaluated using a user study on perceptual quality, and the improvement and evaluation of partial-interleaving are left as future work.

4.3 Result Analysis

In this section, statistic results of the user study are shown and the effects of the whole-interleaving scheme are analyzed in several aspects, including user background, MPEG quality, loss rates, movie types and interleaving distance.

4.3.1 User Background

In our user study, users first fill out a form consisting of several questions about their background. All the questions are optional multiple choice, as follows:

- What's your age?

a) Less than 18 b) 18 to 21 c) 22 to 25 d) 26 to 30 e) Above 30

- How often do you use computer?

  a) < 2 hours each day. b) 2 – 5 hours each day.

  c) Average 8 hours each day.
  d) Average > 8 hours each day.

- Have you ever watched video clips on a computer?

  a) Never b) Occasionally c) Often d) Frequently

- Have you ever watched/listened to continuous media over the internet?

  a) Never b) Occasionally c) Often d. Frequently

A total of 31 people participated in the user study. The majority of the users were undergraduate or graduate students in college, the others from hardware or software companies. Most of the people had high familiarity with computers. About 77% of the groups were using computers 8 hours or more per weekday, the others were using a computer about 2-4 hours each day, and none of people used computers less than 2 hours per day. Within these 31 people, 2 were in the age range from 18 to 21, 8 were between 22 to 25 years old, 14 were between 26 to 30, 5 were above 30, and 2 people chose not to answer this question. Half of the people chose “often” or “frequently” to describe their frequency of watching video clips on computers, with the other 50% of the people had watched video clips occasionally, and none of the people had never done so. All of the people had watched or listened to continuous media over the Internet, however, 20 people (about 65%) said they did it occasionally, and only 35% had been doing this frequently. Figure 4.2 shows the difference between the familiarity of video clips and continuous media over the Internet.

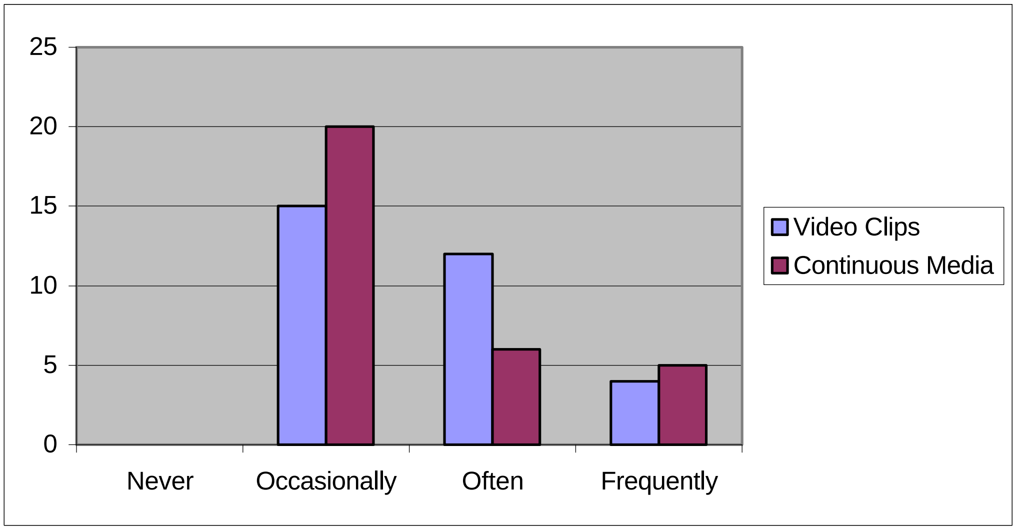


Figure 4.2 Difference of Familiarity between video clips and Continuous Media

The test group in our user study can be summarized as a group of young academic or high-tech people that are highly familiar with computers. Most of them are familiar with video clips on computers, and familiar with continuous media over the Internet, with video clips on computers more popular among this group than continuous media over the Internet.

4.3.2 MPEG Quality

We collected data from the user study and grouped them into 24 data sets, one for each tested video clip. Each data set has 31 samples, and the average value with a 90% confidence interval over the 31 samples is computed and used as the evaluation of the perceptual quality for this video clip.

As described in Chapter 3 (our interleaving approaches), the whole-interleaving algorithm causes has about a 15% increase in file size. We reduce this overhead by encoding the interleaved stream with a slightly higher quality number. In *mpeg_encode*, a higher quality number means a lower video quality, and vice versa. The .mpg file size of a video clip, encoded with the MPEG quality number within the range of [1, 5], decreases exponentially as the quality number increases linearly. The perceptual quality of videos that are encoded with different MPEG quality, however, are quite similar to each other and often cannot be distinguished by end users, especially within the range [1, 5]. In our implementation of the whole-interleaving scheme, we encode interleaved clips with MPEG quality number 2, and non-interleaved clips with MPEG quality number 1.

In order to prove this method, in our user study in the cases of no loss and 10% loss, both the *hockey* video clip and the *CNN* video clip are encoded into two quality versions, one with MPEG quality 1 and the other with MPEG quality 2. Figure 4.3 shows the comparison between those two versions of hockey clips at loss rates of 0% and 10%. And the same comparison for CNN clips is illustrated in Figure 4.4.

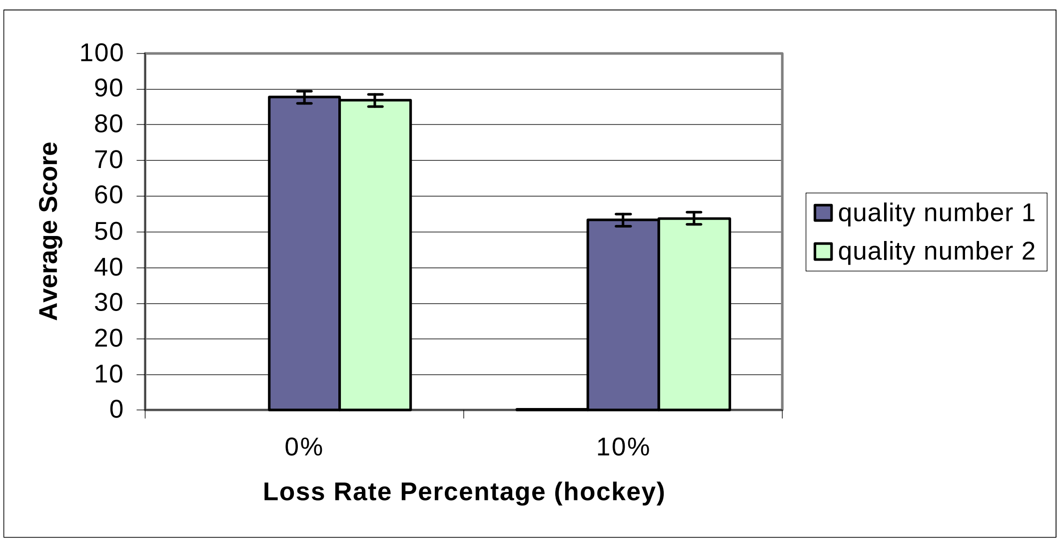


Figure 4.3 Comparison of *hockey* clips with MPEG Quality 1 and 2

In both figures, the x-axis indicates the loss rate, while y-axis is the average score for a certain video clip as given by the users. For each loss rate, the left column represents the clip encoded with quality number 1 and the right with quality number 2. The small error bar on each column indicates the 90% confidence interval of the mean value. For both hockey and CNN, at both loss rates, the average perceptual quality score of the two MPEG quality number versions are statistically the same, since the confidence intervals overlap.

Further more, in our study, in both the hockey and CNN sequences, the very first movie is the best quality movie, encoded with quality number 1 and with no loss. All the users are told of this information before they start. The perfect movie is followed immediately by the same movie encoded with quality number 2 and with no loss. Users do not have this specific information before beginning the test, however. The results we get, for example for the hockey clips, are that among these 31 users, only 10 gave a higher score to the first movie and a difference of no more than 5 points, 19 people gave the same scores to both movie clips, and 2 people even gave higher scores to the second clip. These results suggest our method of decreasing interleaving overhead bandwidth by decreasing the MPEG quality may be effective, and the interleaved stream with lower quality gained better performance in our study, which is shown in the following result analysis sections.

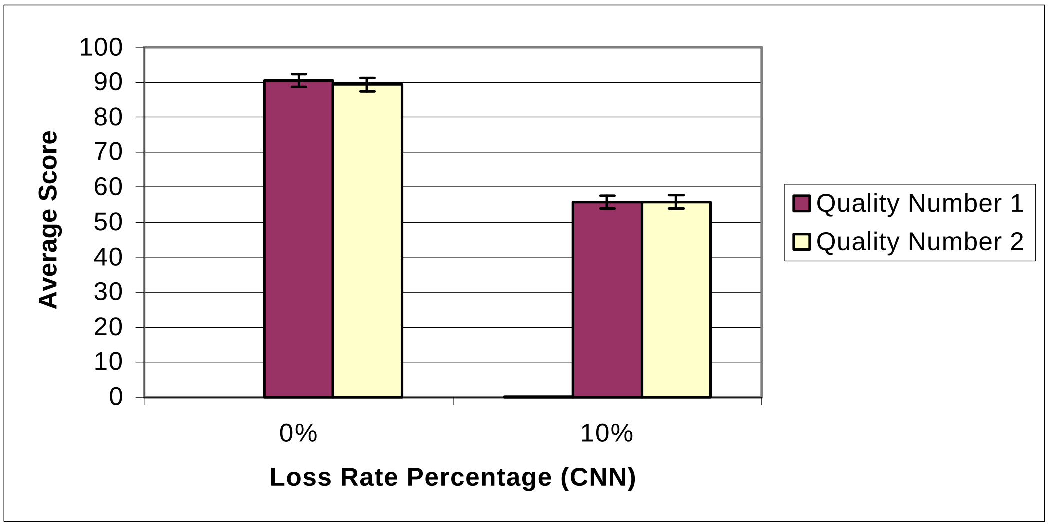


Figure 4.4 Comparison of *CNN* Clips with MPEG Quality 1 and 2

4.3.3 Loss Rate

Five loss rates are used in the user study: 0%, 2%, 5%, 10% and 20%. Figure 4.5 shows the effects of interleaving for the hockey clips under those 5 loss rates. And the effects of interleaving for CNN news clips are shown in Figure 4.6. All the interleaved clips in these two figures are encoded with MPEG quality 2, and the non-interleaved ones are encoded with MPEG quality 1.

From Figure 4.5 it is still not clear if interleaving can improve the quality of hockey clips with a loss rate of 2%. We can see that at a low loss rate of 2%, although the average score for the interleaved clips is higher than that of the non-interleaved ones, the 90% confidence intervals for the two values are overlapping. A larger user study with more participants may be necessary and is listed as an area of future work for this thesis.

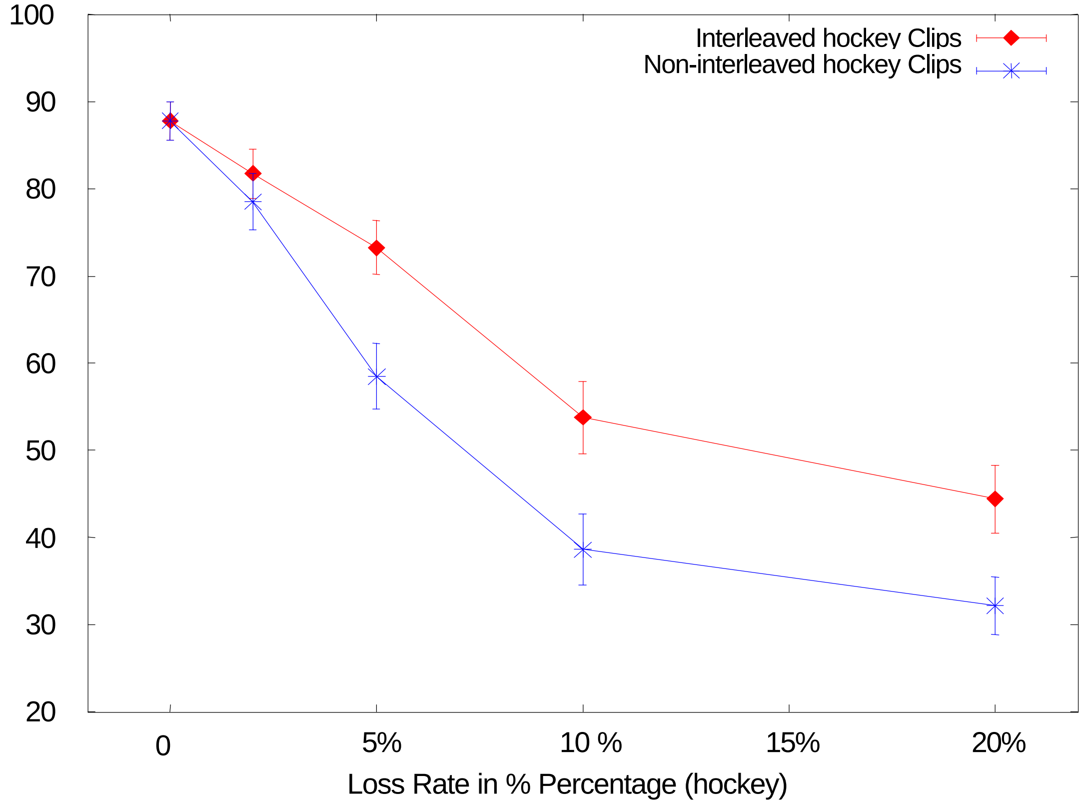


Figure 4.5 Comparison of Interleaved and Non-interleaved “hockey” Clips

Figure 4.5 also shows clearly that interleaving improves the quality of the hockey clips at loss rates of 5%, 10%, and 20%. At a loss rate of 5%, interleaving improves the average score of hockey clip by about 25%, and this improvement goes even higher to 39% at a loss rate of 10%, and 38% at a loss rate of 20%.

Figure 4.6 shows, however, that interleaving improves the quality of CNN news clips at a loss rate of 2%, the average score of the interleaved CNN clips being 12% higher than the non-interleaved clips. But for a loss rate of 5%, although interleaving increases the average score by 11%, the means with 90% confidence are slightly overlapped. We can expect that interleaving is effective in this case also, but it needs to be statistically proven by a future larger user study. At loss rates of 10% and 20%, the quality of the interleaved CNN news clips improves over non-interleaved CNN news clips by about 12% and 44%.

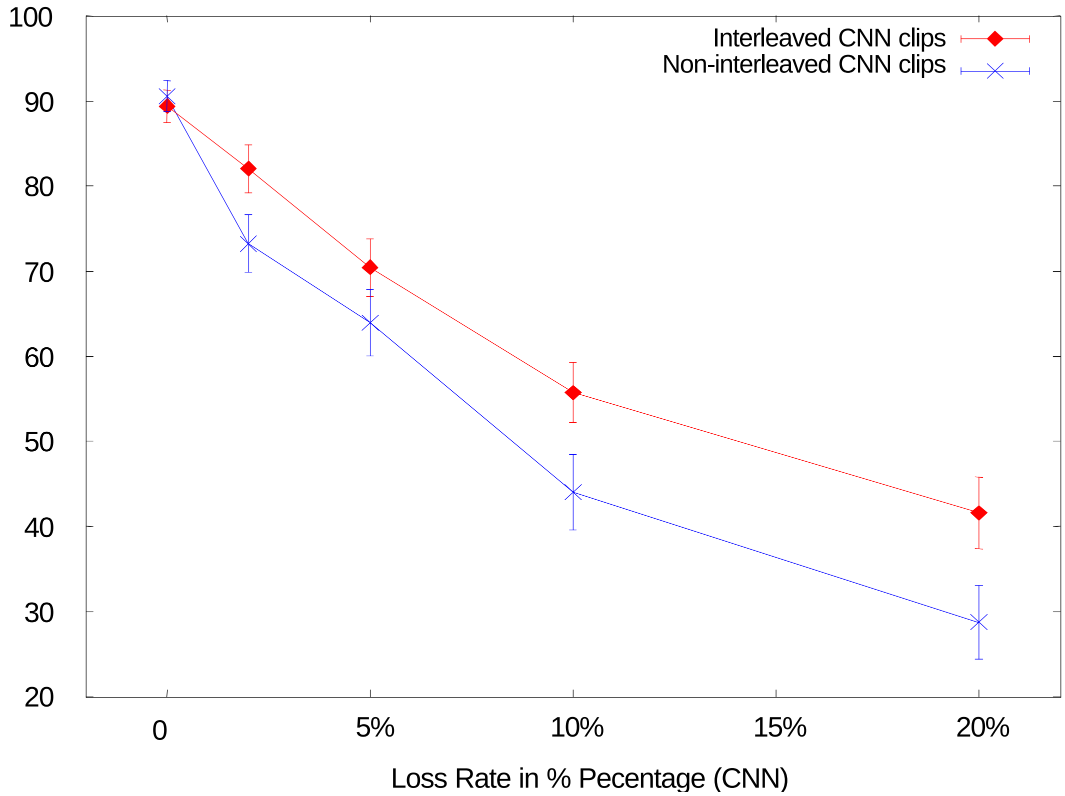


Figure 4.6 Comparison of Interleaved and Non-interleaved “CNN” Clips

4.3.4 Movie Type

Hockey clips and CNN news clips are used in our study to represent two different movie types: hockey has lots of scene changes and action, while CNN has stable objects on the screen and very little action or scene changes. We apply interleaving on both types and see if the movie type can affect the benefits to perceptual quality from our interleaving techniques. Figure 4.7 shows the average increase on scores for hockey and CNN news clips, at loss rates of 2%, 5%, 10% and 20%.

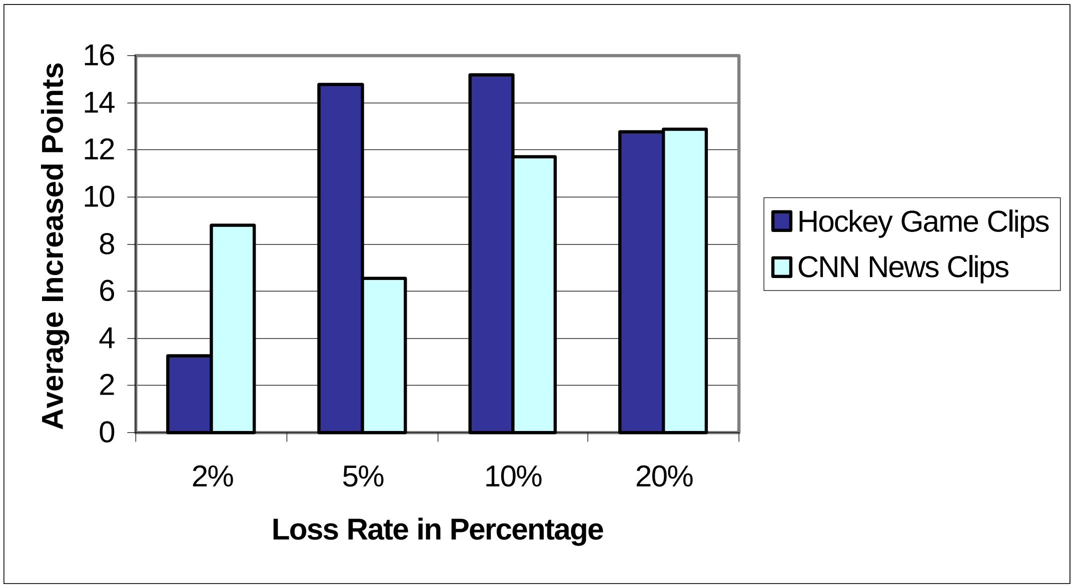


Figure 4.7 Comparison of Average Increase for Hockey and CNN Clips

In Figure 4.7, at the 2% loss rate, interleaving has a higher benefit on CNN clips, with an average increase of about 9 points, than hockey clips, with an increase of about 3 points. At the 5% loss rate, this situation changes: the average increased score for hockey clips, which is about 15 points, is more than two times as the 7 points increase for CNN clips. This status continues as the loss rate goes up to 10%, with 15 points for hockey game and 12 points for CNN news, but the gap between those two becomes smaller. And at a loss rate of 20%, the effects of interleaving on both types tend to be almost the same at about 12 points. To see the effects on an individual video type, as for the hockey video, the benefit to perceptual quality is pretty low at a loss rate of 2%, but for a loss rate of 5%, the increase is much higher and reaches the maximum at a loss rate of 10%, and at a loss rate of 20%, this benefit goes down slightly. For the CNN clips, however, the benefit to perceptual quality starts at a high level at a loss rate of 2%, at a loss rate of 5% the benefit slips down a little bit, but then goes up at loss rate of 10% and 20%.

In our user study, at very low loss rate (around 2%), interleaving technique has more benefits on CNN news than on hockey. However, at loss rate as low as 2%, the quality of the non-interleaved clips is still high, so the effects of average increased points on those clips are smaller than the same amount of increased points on clips with lower quality at the higher loss rate. Overall interleaving may have a greater benefits to movie types with many objects movements and scene changes, such as hockey, than to the movies with little action or scene changes, such as CNN news.

For loss rates higher than 20%, although interleaving may still increase the quality of video to some extent, the resulting quality will likely be unacceptable by end users.

4.3.5 Interleaving Distance

In our user study, two interleaving distances, 2 and 5, were chosen to test the effects of higher and lower distance values on interleaving. At loss rates of 5% and 20%, the hockey clip is interleaved using a distance of 2 and a distance of 5. The comparison of those two interleaving distances, together with the non-interleaved clips is shown in Figure 4.8. For both loss rates, the non-interleaved clips have a worse perceptual quality, with interleaving with distance=2 having the highest average score, and interleaving with distance=5 being ranked in the middle. At a loss rate of 5%, the distance=5 interleaving seems only slightly helpful, since the average scores of non-interleaved and distance=5 interleaving are almost the same, with the latter one slightly higher. At a loss rate of 20%, the distance=5 interleaving does have some effect on improving perceptual quality, but the distance=2 interleaving still performs better.

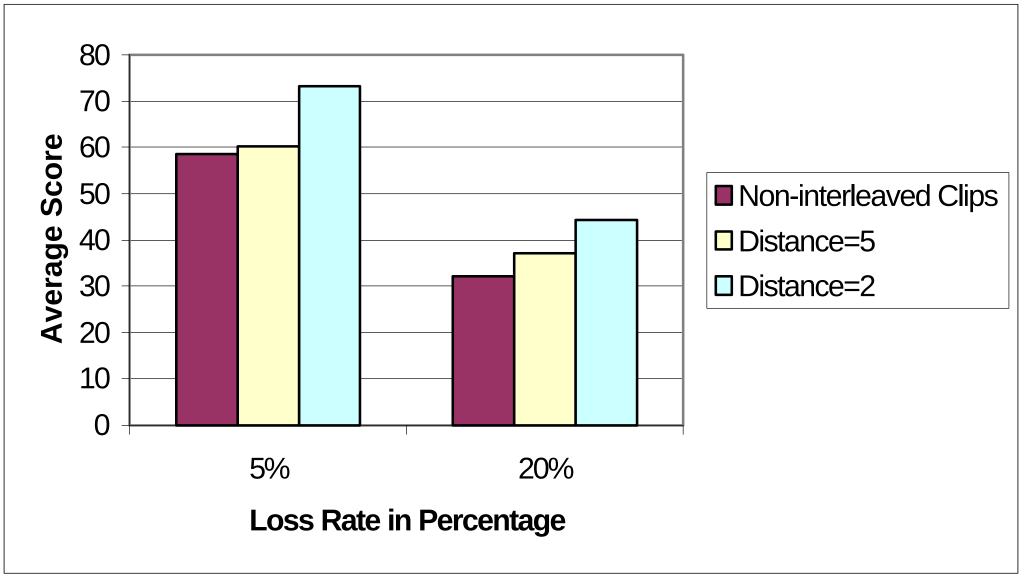


Figure 4.8 Interleaving distance = 2 vs. distance = 5.

A possible reason for this is that in our implementation, at low loss rate, the lost frames tend to be more spread out than those at higher loss rates. So two frames with number n and $n + 5$ have higher possibility to be chosen at the same time, than two frames with number n and $n + 2$. Although the losses of theses two frames appear to be two single losses in interleaved streams, after resequencing at the receiver, the two frames appear next to each other and thus results in one consecutive loss, degrading perceptual quality more than the single losses.

Chapter 5 Future Work

In this thesis, our partial-interleaving approach is implemented but the effects of this method remains to be investigated, possibly by another user study. High processing overhead is introduced by partial-interleaving, and the usage of the *pnmcut* and *pnmpaste* is one reason for this. New methods that can further decrease this processing overhead could be developed as possible future work.

In our user study on perceptual quality for whole-interleaving, the effects of interleaving at some low loss rates is not certain. For example, for the hockey clip, at 2% loss, although interleaving does improve perceptual quality, the 90% confidence intervals over those two values are slightly overlapped. A larger user study with more participants is necessary to better determine the true effects.

In our work, the implementation of both interleaving methods are based on an assumption that one packet contains only one frame. This is often not practical in real multimedia transmission over the Internet. An extension of our implementation with multiple frames per packet and the further evaluation are another possible future work.

Another area for future study is to compare different video loss recovery techniques, such as redundancy, retransmission and interleaving. Since each technique results in different overhead in delay and bandwidth, the effects of this overhead on the network, for example network congestion, link utilization and goodput, remains uncertain and worth further evaluation.

MPEG-1 is used in this thesis as the video compression standard. In the future, other compression standards can be employed to test on the effects of interleaving, for instance, MPEG-2 and MPEG-4.

So far, our interleaving methods are applied to a video stream only. A more complicated yet more challenging research area is to apply interleaving on video and audio streams together. In this case, not only does each of the two media types need to be interleaved, but the synchronization between these two streams needs to be taken care of. This will push the interleaving to a stage that is closer to a real-world implementation.

Chapter 6 Conclusions

In this thesis, we apply interleaving on video streams in order to ameliorate the effects of packet loss over the Internet. The idea is to re-sequence units in a video stream at the sender, transmit them through network to the receiver, and reconstruct the video to its original order. In this way, big gaps in a video stream due to consecutive frame losses, can be spread out to several small gaps, thus improving the perceptual quality of the video stream. Consecutive frame losses can also be the result of propagation loss due to one single frame loss, when video streams are encoded by compression standards using inter-frame compression techniques, such as MPEG.

Two interleaving approaches, whole-interleaving and partial-interleaving are proposed and implemented in this thesis. The difference between those two methods lies in the size of the units to be fed into the corresponding interleaving approaches. Whole-interleaving takes each whole frame as an interleaving unit, while in partial-interleaving, every frame is cut into several equal size sub-frames, with each sub-frame containing the same number of macro blocks, and it is the sub-frame that is used as the basic interleaving unit. An interleaving factor is defined in partial interleaving as the number of sub-frames in each frame.

Since whole-interleaving only deals with whole frames without worrying about the inner structure of each frame, it can be done faster than partial-interleaving. Partial-interleaving has low bandwidth overhead, usually less than 5%. Whole-interleaving, however, may cause up to a 15% overhead. The size of MPEG videos can be decreased by sacrificing some quality. For high quality video, the quality difference between two videos is often undistinguishable while the size can vary significantly. This feature is used in our thesis to decrease the overhead for whole-interleaving by encoding interleaved streams with a quality number slightly lower than non-interleaved streams.

We conducted a user study on perceptual quality to investigate the effects of the whole-interleaving algorithm. The user study was designed to test four aspects of our interleaving method. First, to compare the perceptual quality difference of video encoded at two quality numbers, the same video clip is encoded one with MPEG quality 1 and the second with MPEG quality 2. As a result, the two perceptual qualities of those two videos are evaluated as equal by users, confirming our method to decrease the overhead of whole-interleaving. Second, the effects of interleaving at loss rates of 2%, 5%, 10% and 20% are compared in the user study. Our interleaving algorithm can improve the perceptual quality by about 25% at a loss rate of 5%, and about 40% at loss rates of 10%, and 20%. The third aspect of the user study is to observe the effects of interleaving on different movie types. Two video clips, a hockey game and CNN news show, are used to represent two movie types, one with lots of object movement and scene changes, and the other with stable screen and few object movements. The results show that although interleaving has better effect on CNN new clips when the loss rate is only 2%, at other higher loss rate, the hockey clip gains more improvement in video quality than the CNN news clip. Lastly, we examined the interleaving distance, which is the distance of the small gaps after one big gap is spread out. Two distance values, 2 and 5, are used in our study, and as indicated by the result of the user study, interleaving with distance 2 has much better performance than interleaving with distance 5.

In summary, we propose and implement two interleaving methods using different interleaving units, in which video streams are re-sequenced at the sender and reconstructed back to the original order at the receiver. The partial-interleaving method has less than a 5% overhead in bandwidth. The whole-interleaving method can be applied in real-time, but has about a 15% overhead in bandwidth. This overhead can be decreased by encoding the interleaved stream with slightly higher quality number, and our user study shows that this method of decreasing overhead in interleaved video stream is acceptable to users and the interleaving can significantly improve the perceptual quality of video streams under various loss rates.

Appendix A: Simulation Tools

Three MPEG-1 tools are used in whole-interleaving and partial interleaving, *mpeg_encode, mpeg_play and mpeg_encode*. Another two utilities, *pnmcut* and *pnmpaste*, are only used in partial-interleaving to operate on slices of frames.

Most MPEG tools we used are from University of California, Berkeley, developed by researchers and students in the Berkeley Plateau Multimedia Research Group [6], including *mpeg_encode, mpeg_play and mpeg_stat*. Other tools include *pnmcut* and *pnmpaste*.

mpeg_encode

This tool is used to encode MPEG-1 bitstreams using the platform such as SunOS 4.x, DEC Alpha running OSF1, DECstation 5000 runnin Ultrix and HP 9000 series. It was developed by Kevin Gong at UC Berkeley and later been modified into different versions. The version we are using is *mpeg_encode* 1.5. The encoder is invoked by the following command:

mpeg_encode [options] parameter-file

The parameter file includes a list of input files and other parameters that are necessary to be used by the encoder. The file must contain the following information:

PATTERN <pattern> is to set the sequence of I, P, and B-frames. The pattern affects speed, quality, and compression. Usage:

PATTERN <IPB pattern>

GOP: A Groups of Pictures is a roughly independently decodable sequence of frames. An MPEG video stream is made of one or more GOPs. Here one may specify how many frames each GOP should be. A GOP must start with an I-frame, and the encoder will enforce that by taking this number as the minimum number of frames in a GOP. Usage:

GOP_SIZE num

Where num = the number of frames in a GOP.

Search Window defines the window in which motion vectors are searched for. The window is a square. One can specify the size of the square, and whether to allow half-pixel motion vectors or not. Usage:

PIXEL <FULL or HALF>

RANGE num [numB]

HALF means the half-pixel vectors are allowed, which result in better quality and better compression.. The search window is +/- num pixels in the X and Y directions.

Q-Scale: The quantization scale values give a trade-off between quality and compression. Using difference Q-Scale values have little effect on encoding speed. The Q-Scale Values can be set separately for I-, P-, and B-frames. Usage:

IQ-Scale num

PQ-Scale num

BQ-Scale num

Larger numbers give better compression, but worse quality. In the following, the quality numbers are peak signal-to-noise ratio, defined as:

$$20 \log_{10} \frac{255}{\sqrt{MSE}}$$

Where MSE is the mean square error.

Search Techniques: There are several different motion vector search techniques available for both P-frame search and B-frame search. Using different search techniques presents little difference in quality, but a large difference in compression and speed. There are 4 types of P-frame search: Exhaustive, TwoLevel, SubSample, and Logarithmic; and 3 types for B-frame search: Exhaustive, Cross2, and Simple. Usage:

PSEARCH_ALG ptechnique

BSEARCH_ALG btechnique

Slice is an independently decodable unit in a frame. It can be as small as one macroblock, or it can be as big as the entire frame. Barring transmission error, adding slices does not change quality or speed; the only effect is slightly worse compression. Usage:

SLICES_PER_FRAME num

Where num is the number of slices in a frame.

Original or Decoded: The encoder can use either the original frames as reference frames, or the decoded frames. Using the decoded frames gives better playback quality, but is slower and seems to give worse compression. It also cause some complications with the parallel encoding. Original is recommended. Usage:

REFERENCE_FRAME ORIGINAL

Output defines the name of the output file. Usage:

OUTPUT < output file >

Specifying Input Files: The encoder can expect five base types of input files: PPM, PNM, JMOVIE, JPEG, and YUV. The PNM format stands for portable anymap file format, and PPM stands for portable pixmap file format. PPM is a subset of PNM. We use PPM as the base file format to hold the source video streams before they are encoded. The names of input files and the directory and in which the input files are located must be defined by parameter input and input_dir. The way to convert a file to the base file format should also be specified using the parameter input_convert.

A sample parameter file to generate movie clip hockey1.mpg is as follows:

```
PATTERN IBBPBBPBB
GOP_SIZE 18
PIXEL HALF
IQSCALE 5.06
PQSCALE 3.80
BQSCALE 8.18
RANGE 10
PSEARCH_ALG LOGARITHMIC
```

```
BSEARCH_ALG SIMPLE
REFERENCE_FRAME ORIGINAL
INPUT_CONVERT *
BASE_FILE_FORMAT PPM
SLICES_PER_FRAME 1
OUTPUT hockey1.mpg
INPUT_DIR /users/yaliz/ppm_hockey1
INPUT
hockey1_*.ppm [00001-00602]
END INPUT
```

mpeg_play

mpeg_play decodes and displays MPEG-1 encoded bitstreams on systems running X11. The player will create a new window, display the bitstream, and exit. Any error messages or notices are sent to stderr. This is the tool we use to play test movie clips in our user study on perceptual quality.

Another usage of *mpeg_play* in the implementation is to dither the encoded video streams back to ppm file format, which is the source frame format. *mpeg_play* can also define the starting frame and ending frame to be played or dithered. Given a frame rate, this function of *mpeg_play* gives us flexibility to choose which section of the movie we want to process and how long the movie clips will last. For example, the frame rate is 30 frame/s, then a 20 seconds movie clips will need about 20 x 30 = 600 frames. The usage of this function is:

```
mpeg_play -start startframenum -end endframenum -dither ppm [moviename]
```

mpeg_stat

mpeg_stat decodes MPEG-1 encoded bitstreams collecting varying amounts of statistics. The basic information is the pattern of frames used, number of bytes for each frame type, the specified parameters, and lengths of vectors. For each frame type, the average size, compression rate, Q-factor, and time to decode are given. It is invoked as:

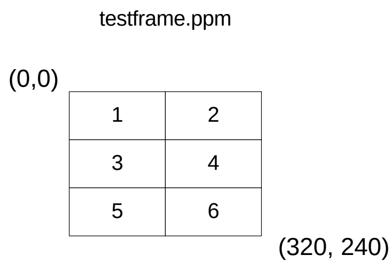
```
mpeg_stat    moviename_in_mpg
```

pnmcut

pnmcut reads a portable anymap frame (pnm) as input, extracts the specified rectangle, and produce a portable anymap as output. As introduced in the *mpeg_encode* section, we use ppm as our source frame format in a video stream, and ppm is a subset of pnm format. Thus any tool that operate on pnm will also be suitable for ppm format. The use of *pnmcut* is as follows:

```
pnmcut  x  y  width  height  [pnmfile]
```

The pair of x and y decides a starting point of the cut operation, and the width and height defines the size of the slice that's going to be cut off from the frame. The normal resolution of an MPEG-1 frame is 320 x 240 pixels, and each frame is of 20 x 15 = 300 macroblocks. For example a partial-interleaving factor of 6 (2 x 3), each sub-frame will be 160 x 80 pixels. Figure 3.6 shows the corresponding value of x, y, width and height for each sub-frame:



	X	Y	width	height
1.ppm	0	0	160	80
2.ppm	160	0	160	80
3.ppm	0	80	160	80
4.ppm	160	80	160	80
5.ppm	0	160	160	80
6.ppm	160	160	160	80

Figure 3.6 A sample of 6-way interleaving with pnmcut

pnmpaste

pnmpaste reads two frames in portable anymap format as input, inserts the first frame into the second at the specified location, and produces a format in portable anymap format the same size as the second as output. This tool is most useful in combination with *pnmcut*. In our implementation of partial-interleaving, this utility is used at the resequence step at the sender and the repetition step at the receiver, in the latter case a lost part of a frame is recovered by pasting the sub-frame at the same position of previous frame. It is invoked in the following matter:

```
pnmpaste -replace frompnmfile x y [intopnmfile]
```

Where x and y is the location at which the first pnm file will be pasted to the second pnm file.

References

[1] G. Carle and E. W. Biersack, "Survey of Error Recovery Technologies for IP-based Audio-Visual Multicast Applications", IEEE Network, 11(6):24-36, November/December 1997.

[2] C. Perkins and O. Hodson, *"Options for Repair of Streaming Media"*, The Internet Society RFC 2354, 1998.

[3] R. Osso, and S. Zamir, "*Handbook of Emerging Communications Technologies, the Next Decade"*, CRC Press, ISBN 0-8493-9594-1, 2000.

[4] "*A Partially Reliable Transport Protocol, Working Draft"*, http://www.cs.kau.se/~katarina/prtp/draft980703.html

[5] Y. Liu and M. Claypool, "*Using Redundancy to Repair Video Damaged by Network Data Loss*", In Proceedings of IS&T/ACM/SPIE Multimedia Computing and Networking 2000 (MMCN00), January 25-27, 2000, San Jose, California, 2000.

[6] "*Berkeley MPEG-1 Video Encoder*", http://www.artsoft.com.br/linux/ldpp/en/app/00ipdf.htm

[7] J. L. Mitchell, W. B. Pennebaker, C. E. Fogg and D. J. LeGall, "*MPEG Video Compression Standard*", Chapman & Hall, ISBN 0-412-08771-5, 1996.

[8] C. Perkins, O. Hodson, V. Hardman, "*A Survey of Packet-Loss Recovery Techniques for Streaming Audio"*, IEEE Network Magazine, September/October 1998.

[9] R. Steinmetz and K. Nahrstedt, "*Multimedia Computing, Communications & Applications"*, Prentice Hall International, Inc, ISBN 7-302-02414-6, 1998.

[10] J. C. Bolot, *"End-to-end Packet Delay and Loss Behavior in the Internet",* In Proceedings of ACM Sigcomm '93, San Francisco, CA, 1993.

[11] M. Handley, *"An Examination of MBONE Performance",* Technical Report, University of Southern California, Information Sciences Institute USC/ISI, ISI/RR-97-450, January 1997.

[12] V. Paxson, *"End-to-end Internet Packet Dynamics",* Computer Communication Review, Proceedings of ACM SIGCOMM'97 Conference, Cannes, France, September 1997, October 1997.

[13] M. Yajnik, J. Kurose, and D. Towsley, *"Packet Loss Correlation in the Mbone Multicast Network",* In Proceedings of IEEE Blobal Internet, London, UK, Novermber 1996, IEEE.

[14] S. Lin,and D. J. Costello, *"Error Correcting Coding: Fundamentals and Applications"*, Prentice Hall, Englewood Cliffs, NJ, 1983.

[15] J. Ronsenberg, *"Reliability enhancements to NeVoT"*, Bell Laboratories, December, 1996.

[16] J. Rosenberg and H. Schulzrinne, *"An RTP payload fromat for generic forward error correction"*, Network Working Group, RFC 2733, December 1999.

[17] B. J. Dempsey, M. T. Lucas, and A. C. Weaver, *"An Empirical Study of Packet voice Distribution Over a Campus-Wide Network"*, In Proceedings of the IEEE 19th Conference on Local Computer Networks, October 1994, Minneapolis, Minnesota, 1994.

[18] B. Dempsey, J. Liebeherr, and A. Weaver, *"On Retransmission-based Error Control for continuous Media Traffic in Packet-Switching Networks"*, Computer Networks and ISDN Systems, 28(5):719 – 736, March 1996.

[19] S. Mccanne, V. Jacobson and M. vetterli, *"Receiver-driven Layered Multicast"*, In SIGCOMM 96, pp 117-130, Stanford, CA, August 1996.

[20] S. McCanne, M. vetterli and V. Jacobson, *"Low-complexity Video Coding for Receiver-driven Layered multicast"*, SSC/1997/001, EPFL, Lausanne, Switaerland, January 1997.

[21] J. L. Ramsey, *"Realization of Optimum Interleavers"*, IEEE transactions on Information Theory, IT-16:338-345, May 1970.

[22] G. A. Miller and J. C. R. Licklider, *"The Intelligibility of Interrupted Speech"*, Journal of the Acoustical Society of America, 22(2):167-173, 1950.

[23] D. J. Goodman, G. B. Lockhart, O. J. Wasem and W. C. Wong, *"Waveform Substitution Techniques for Recovering Missing Speech Segments in Packet Voice Communications"*, IEEE Transactions on Acoustics, Speech, and Signal Processing, ASSP-34(6):1440-1448, December 1986.

[24] H. Sanneck, A. Stenger, K. Bem Younes and B. Girod, *"A New Technique for Audio Packet Loss Concealment"*, in IEEE Global Internet 1996, pages 48-52, IEEE, December 1996.

[25] V. Hardman, A. Sasse, M. Handley, and A. Watson, *"Reliable Audio for Use over the Internet"*, Proceeding of INET 1995, Hawaii, Internet Society, Reston, Virginia, 1995.

[26] Jason Gerek, William Buchanan, *"MMlib – A Library for End-to-End Simulation of Multimedia over a WAN"*, MQP CS-MLC-0001, Advisor: Mark L. Claypool, May 1998.

[27] 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.

[28] Chwan-Hua Wu and J. David Irwin, "*Emerging Multimedia Computer Communication Technologies*", Prentice Hall PTR, ISBN 0-13-079967-X, 1998.