Video Compression and Streaming over Packet-switched Networks

by

Wai-tian Tan

B.S. (Brown University) 1992
M.S. (Stanford University) 1993

A dissertation submitted in partial satisfaction of the
requirements for the degree of
Doctor of Philosophy

in

Engineering - Electrical Engineering
and Computer Sciences

in the

GRADUATE DIVISION
of the
UNIVERSITY of CALIFORNIA at BERKELEY

Committee in charge:

Professor Avideh Zakhor, Chair
Professor Lawrence A. Rowe
Professor Stanley Klein

Fall 2000

The dissertation of Wai-tian Tan is approved:

Chair Date

Date

Date

University of California at Berkeley

Fall 2000

Video Compression and Streaming over Packet-switched Networks

by

Wai-tian Tan

Abstract

Video Compression and Streaming over Packet-switched Networks

by

Wai-tian Tan

Doctor of Philosophy in Electrical Engineering and Computer Sciences
University of California at Berkeley

Professor Avideh Zakhor, Chair

This dissertation deals with video streaming over (a) best-effort unicast and multicast packet networks, and (b) packet networks with traffic differentiation. First, we consider the problem of developing a scalable video compression scheme that is resilient to random packet loss. By using three-dimensional subband decomposition and data partitioning in the subband coefficient domain, we demonstrate a packetization scheme in which each packet contains information about disjoint partitions of subband coefficients. In such a way, individually decodable packets of equal visual importance can be obtained. In actual experiments over the Internet, our scalable compression technique is found to achieve significantly lower average distortion and smaller variation in image quality compared to schemes using non-resilient scalable compression.

Second, we explore the transport protocols necessary for transporting scalable video over best-effort unicast and multicast packet networks. For unicast, we construct a TCP-friendly transport protocol that shares bandwidth fairly with multiple instances of itself and TCP. Our TCP-friendly protocol, however, differs from TCP in being low-latency and without abrupt changes in communication rate. For multicast, we propose the layered Forward Error Correction (FEC) scheme that allows different receivers in a multicast environment to efficiently receive different levels of protection commensurate with their respective channel conditions. For rate control at the receivers, we propose an equation-based approach that computes network usage as a function of measured network characteristics. We show that equation-based rate control achieves more fair bandwidth sharing as compared to existing multicast rate control schemes. Simulation and MBONE experiments have shown that video quality is significantly improved when layered FEC is used.

Third, we address the problem of transmitting compressed MPEG video over packet networks with traffic differentiation in terms of loss and delay. To this end, we present a framework in which MPEG-compressed video is divided into sub-streams of different delay and loss sensitivities. These sub-streams are then transported using different quality of service to improve overall network utilization. We present a number of packet classification schemes for MPEG bit-stream based on delay and loss characteristics of the data, and compare those schemes using commercial DVD contents. It is shown that significant reduction in distortion or delay can be achieved compared to schemes that treat the MPEG stream as homogeneous in importance.

Professor Avideh Zakhor
Dissertation Committee Chair

Contents

List of Figures

List of Tables

Acknowledgements

I would like to thank my advisor Prof. Avideh Zakhor for her guidance throughout my graduate career at Berkeley. In particular, her pragmatic approach to identifying research topics and meticulous attention to details have greatly shaped this work in its current form.

This work has also benefited from helpful discussions with several people. I would like to thank Anindo Banerjea, Sally Floyd, and Jean Bolot for many insightful discussions and comments. I would also like to thank Prof. Lawrence A. Rowe, Prof. Stanley Klein, and Prof. David Messerschmitt for helpful comments during the Qualifying Examination or in reading earlier drafts of this dissertation.

I would like to thank Intel Foundation for financial support through a graduate fellowship program for the 1999-2000 year, and also funding from AFOSR Muri grant FDF49620-00-1-0327-05/05, NSF grant ANI-9905799, the Cal State Micro program, Sun Microsystem, Philips and Intel.

The companionship and support from students in the Video and Image Processing Lab are gratefully acknowledged. I would like to thank Samson Cheung, Nelson Chang, Ralph Neff, Edward Chang, Thinh Nguyen, and Vito Dai, with whom I am fortunate to have spent a few years with.

Finally, I am most grateful for my parents and my fiancée Joanne. Little could I achieve without their love, support and encouragement.

Chapter 1

Introduction

Recent years have witnessed phenomenal growth of the Internet. The number of hosts has reached 88 million in August 2000 versus 30 million in January 1998 and 4.9 million in January 1995[1]. One important driving force behind the rapid growth is the proliferation of multimedia content types such as image, audio, and video, which have become important parts of many current and emerging Internet applications. Web-pages nowadays commonly contain images, audio, and sometimes streaming video.

This use of multimedia data is partly enabled by the increasing processing power of modern computers, and the advent of broad-band technologies such as cable modem and digital subscriber line Internet connections. However, effective video communications over the Internet, or best-effort packet networks in general, has remained a

[1]Numbers for 1998 and 2000 are from www.netsizer.com, while the 1995 estimate is from Matthew Gray of MIT. All figures represent the number of hosts listed with the Domain Name Service (DNS).

challenge due to the mismatch between the quality of service (QoS) required by the transport of compressed video and that offered by the network.

Originally designed for data traffic, the current Internet provides an unpredictable and time-varying channel in terms of delay, bandwidth, and packet loss rate. Nevertheless, by being tolerant to large delay and delay-jitter [2], data applications can be effectively supported using the ubiquitous Transmission Control Protocol (TCP), which essentially transforms time-varying bandwidth and loss into extra delay and delay-jitter. This transformation is achieved by adapting to the time-varying bandwidth, and using end-to-end retransmissions to recover from packet losses. While the TCP infrastructure has been the basis of many successful data applications, such as electronic mail, file transfer, web surfing, etc., the additional delay and delay-jitter is unacceptable for streaming audio and video applications.

Video compression techniques, on the other hand, are typically not designed with the transport channel characteristics in mind. Instead, a constant data-rate channel with low loss rates is assumed. For example, the popular MPEG-1 standard is designed for the compact disk media. As a result, such network-ignorant compression techniques cannot accommodate the dynamic nature of best-effort packet networks in terms of bursty losses and time-varying bandwidth and latency.

There are two approaches to bridge the gap between what the network offers and what the compressed media requires. The first approach is to rebuild or augment the

[2] Delay-jitter is the amount of variation in the end-to-end packet transit time.

current network to provide QoS guarantees necessary to transport network-ignorant multimedia. This is typically achieved through resource reservation and admission control to establish deterministic or statistical guarantees on delay, delay-jitter, and packet loss rate. Examples are Asynchronous Transfer Mode (ATM) networks and the Integrated Services (IntServ) model developed for the Internet. While this approach promises to provide high quality video, complexity and deployment issues have prevented its wide-spread adoption despite extensive research and development efforts. Even if such networks are eventually widely deployed, they are likely to have the undesirable features of (a) higher usage cost and (b) possibility of call blocking when reservations cannot be satisfied. A second approach is to develop *adaptive applications* that attempt to react to changing network conditions. The ability of an application to adapt to network conditions is limited on one side by the minimal network support of best-effort networks, and on the other side by limitations of existing compression technologies.

This dissertation explores methods to expand upon the adaptive applications approach. The dissertation is organized as follows. In Chapter 2, we present selected background materials relevant to video communication over best-effort packet networks and survey some existing approaches.

Chapter 3 presents a bandwidth-scalable and error-resilient video compression scheme for video streaming applications. Under the proposed scheme, compressed video can be packetized into individually decodable packets to effectively eliminate error propagation. In addition, each packet is of similar visual importance so as to reduce fluctuation in image quality under lossy conditions. Furthermore, the effects of packet loss are dispersed over the entire picture, resulting in spatially uniform degradation rather than the more objectionable missing blocks. The finely scalable nature of the proposed scheme further facilitates network transmission, as compressed video can be easily filtered to meet time-varying bandwidth constraints. Lastly, by using a low-complexity entropy coding technique, real-time encoding and decoding using the proposed scheme are possible on today's computers.

In Chapter 4, we introduce a unicast video transmission scheme over best-effort packet networks combining a low-delay TCP-friendly transport protocol in conjunction with the codec developed in Chapter 3. The goal of the transport protocol is to mimic the TCP rate control algorithm on a macroscopic scale so that fair sharing of resources for multiple instances of the protocol and TCP can be established. However, the protocol differs from TCP in that (a) no retransmission is performed to reduce end-to-end delay and delay-jitter, and (b) rate adaptation is less dynamic to prevent abrupt changes in communication rate. With the protocol being used to determine the desired communication rate as a function of network conditions, the proposed scalable compression scheme allows the media bit-rate to be easily adjusted to meet the communication rate. The error-resilience of the proposed compression scheme further improves visual quality under packet losses. Due to the reactive nature of typical rate control algorithms, error-resilience is often needed to maintain acceptable video quality, especially when low-latency is desired. Actual Internet experiments together with simulations are used to evaluate the performance of the combined scheme.

Chapter 5 explores the use of layered forward error correction (FEC) as an error control mechanism in a layered multicast framework. By organizing FEC into multiple layers, receivers can achieve different levels of protection commensurate with their respective channel conditions. Efficient network utilization is achieved as FEC streams are multicast only to receivers that need them. Furthermore, FEC is used without overall rate expansion by selectively dropping data layers to make room for FEC layers. Effects of bursty losses are amortized by staggering the FEC streams in time, giving rise to a trade-off between delay and quality. For rate control at the receivers, we propose an equation-based approach that computes network usage as a function of measured network characteristics. The main advantage of the equation-based multicast rate control scheme over existing schemes such as Receiver-driven Layered Multicast (RLM) and Receiver-driven Layered Congestion control (RLC) is the ability to achieve more inter-session fairness when multiple video sessions are simultaneously being transmitted. Fairness is achieved because competing sessions sharing a path will measure similar network characteristics. A series of simulation and actual MBONE experiments are performed using the compression technique presented in Chapter 3 to study the protocols and algorithms. We find that video quality is significantly improved at the same communication rate when layered FEC is used.

While the transmission of standard MPEG video over best-effort packet networks is decidedly difficult, future Internet service disciplines promise to provide enhanced support for multimedia communications. In Chapter 6 we present a framework for transmitting MPEG-compressed video over a Differentiated Services Internet by dividing the bit-stream into sub-streams of different delay and loss sensitivities. A Differentiated Services network allows applications to use different service models. For example, packet flows can be marked as real-time so that network routers can treat them differently than best-effort packet flows. The MPEG sub-streams can be transported using multiple traffic service classes with different bandwidth, transmission delay, and packet loss characteristics. This approach yields better network utilization as compared to using a single class due to the ability to transmit packets using traffic classes with commensurate QoS. We present specific methods for separating MPEG bit-stream into parts of different delay and also examine several existing methods of loss differentiation. Commercial DVD contents are used to evaluate the effectiveness of the different loss and delay differentiation methods. Simulations of transmitting a DVD movie over a DiffServ network are also presented. It is shown that a distortion reduction by 4 dB can be achieved by a packet classification scheme that is optimized for loss while another packet classification scheme that is optimized for delay can reduce end-to-end playback time by 33 ms when compared to packet classifiers that treat the MPEG stream as homogeneous in importance.

Finally, Chapter 7 presents a summary of the results in this dissertation along with suggestions for future work. Portions of Chapters 3 and 4 have been presented in [129, 126, 125, 7, 124, 123], while parts of Chapter 5 have been presented in [127, 128, 131]. The results in Chapter 6 have been presented in [130].

Chapter 2

Background

This Chapter reviews background on the relevant characteristics of the Internet to video communications. Several existing approaches to video communication over best-effort packet networks are surveyed.

2.1 Internet Service Models

The Internet was developed to connect a heterogeneous mix of networks that employ different packet switching technologies. While each network has a native mechanism of communication within itself, the Internet enables inter-network communication by providing a uniform scheme of *addressing*, *routing* and *packet interpretation*. Each computer on the Internet has a unique Internet address, which allows computers on different networks to identify each other. Internet routing, on the other hand, provides the mechanism to allow packets to move from one Internet address to another. Finally, the Internet Protocol (IP) serves as a "common language" to allow consistent interpretation of packets across networks. A common definition of the Internet is the set of networks connected by IP.

2.1.1 The Best-Effort Internet Protocol

One central issue in the design of the Internet is whether it is necessary for IP to provide guarantees in packet delivery. Historically, the protocol originally proposed for the Internet was conceived to provide both reliable and unreliable packet delivery services [16]. However, early implementations of the protocol only provided the reliable service [74]. It was not until the early 1980s that the current unreliable or *best-effort* IP service was established. There are two motivations for the best-effort design. First, the Internet is designed to connect existing networks. Thus, one of the most important guiding principles in its design is to include a wide variety of existing networks without requiring changes in their internal operation [101, 74]. To achieve this goal, minimal demands are to be placed on individual networks. Second, research efforts in packet audio during the 1970s had suggested that packet loss should sometimes be handled directly by the application rather than by the network [145, 74].

For applications such as file transfer and email, packet loss is unacceptable. A consequence of a best-effort IP is that reliability needs to be established on an end-to-end basis. To this end, *transport protocols* that utilize the services of IP and provide reliability through end-to-end retransmission are needed. This design of providing reliability on an end-to-end rather than on a hop-to-hop basis is arguably more cost-effective from a system design perspective [113]. The Transmission Control Protocol (TCP), in particular, provides a popular platform for building reliable applications over the intrinsically unreliable Internet. Today, most traffic on the Internet, such as web traffic and email transport, are carried using TCP.

TCP provides reliable transport by retransmitting packets lost in the network. One consequence of using retransmissions is the associated increase in delay and delay-jitter. For data applications, the trade-off is justified as the importance of reliability far exceeds low delay or delay-jitter. However, for transport of continuous media such as audio or video streams, an increase in either delay or delay-jitter may be highly objectionable. For this reason, TCP is typically not used for multimedia applications. Instead, the best-effort User Datagram Protocol (UDP) is often preferred. UDP is simply a user interface to the best-effort IP service. UDP also provides a packet checksum to verify data integrity, and a port-numbering scheme for demultiplexing traffic sent to the same destination. The important tasks of (a) *error control* or coping with packet losses, and (b) *rate control* or determining and maintaining the appropriate transmission rates are relegated to the application.

In Section 2.2, we survey several existing approaches to video communication over best-effort packet networks, and outline how the tasks of rate control and error control are performed.

2.1.2 The Multicast Extension to IP

IP is a unicast or one-to-one protocol that delivers a packet from one Internet address to another Internet address. For applications such as group video conferencing or Internet "radio" broadcasting, where a large audience is simultaneously present, unicast delivery may be too inefficient. For example, using unicast transport, an Internet "radio" station must keep track of the Internet address of every listener, and send a separate stream to each. This approach leads to two problems. First is the high complexity required for the station. Second is the inefficient use of network resources as a separate copy of the same audio stream needs to be served to each receiver – even if some receivers are located on the same network or possibly even the same destination host. The multicast extension to IP [31], or IP-multicast, addresses these problems by providing a service model in which a source can simultaneously communicate with multiple receivers by transmitting only a single copy of every packet. The network is then in charge of ensuring that only one copy flows through each link of the network that leads to a receiver.

The deployment of IP-multicast offers a best-effort multicast infrastructure, called the multicast backbone (MBone), over the existing unicast Internet. The MBone is sometimes referred to as a "virtual network" as it shares the same physical media as the unicast Internet. A special subset of IP-addresses are reserved as multicast or group addresses. Under IP-multicast, sources and receivers communicate using a multicast address similar to the way radio communication uses frequency bands: a source sends to a multicast address and receivers “tune” to the desired address. As a result, senders and receivers need not be aware of the existence of each other, and the difficult task of locating sources and destinations and of routing packets are left to the network.

Video multicast over the MBone is already happening. For example, the weekly “Berkeley Multimedia, Interfaces, and Graphics Seminar” has been broadcast on the MBone for a number of years [110]. Nevertheless, the wide-spread use of the MBone as a video distribution medium is limited by a number of deployment difficulties including scalability to a large network [2, 37]. Currently, a large amount of research effort has been focused on solving the scalability problem [136, 70, 34, 6, 40]. For instance, *sparse-mode* multicast routing [34, 6, 40] has been developed to improve the scalability of traditional *dense-mode* multicast routing protocols [142, 88, 33]. A dense-mode multicast routing protocol assumes interested receivers are densely populated over the network; thus it is practical to flood the entire network with packets, and rely on uninterested receivers to explicitly request pruning. Recent research has also investigated alternative service models that can supported more efficiently in the network [57, 18, 99].

When streaming video in a multicast environment, different receivers may be connected to the source via links of different bandwidth. As a result, it is impossible to use a single stream to optimize reception simultaneously for all receivers. This *bandwidth heterogeneity* problem can be solved by using scalable video. In a scheme known as *layered multicast* [32], video is compressed in a layered manner and different layers are transmitted on distinct multicast addresses. In such a way, different receivers with different available bandwidths to the source can subscribe to a different number of layers to effect rate control. Specific schemes for realizing rate control in a layered multicast framework will be discussed and compared later in Chapter 5.

2.1.3 Differentiated Services

The best-effort Internet has proven to be adequate for data applications. In an effort to provide enhanced quality of service (QoS) to more demanding applications such as video communications, the Internet Engineering Task Force (IETF) has considered a number of architectural extensions to the best-effort Internet. The Integrated Services (IntServ) model of the Internet [146], for instance, is an attempt to provide end-to-end QoS guarantees in terms of bandwidth, packet loss rate, and delay, on a per-flow basis. QoS guarantees are established using explicit resource allocation based on the Resource Reservation Protocol (RSVP) [155]. However, the high complexity and cost of deployment of the RSVP-based service architecture eventually led the IETF to consider other service differentiation mechanisms. The Differentiated Services (DiffServ) model, in particular, is specifically designed to achieve low complexity and easy deployment at the cost of less stringent QoS guarantee than IntServ [9]. Under DiffServ, service differentiation is no longer provided on a per-flow basis. Instead, it is based on the *code-point* or tag in each packet. Thus, packets having the same tags are given the same treatment under DiffServ regardless of where they originate.

The cost of easy deployment for DiffServ compared to IntServ is the reduced level of QoS support. The research challenge is to determine how to support multimedia content such as video in a DiffServ Internet. In Chapter 6, we present a packet tagging scheme for compressed MPEG-2 contents according to the relative loss and delay sensitivities of each packet. By sending packets using traffic classes of commensurate QoS within a DiffServ framework, better network utilization can be achieved.

2.2 Video Transport over Best-effort Networks

In this section, we categorize existing approaches to video communication over best-effort packet networks and compare their relative advantages and disadvantages. In particular, we focus on the ability of each approach to achieve rate control and error control. For rate control, we assume that a desired communication rate, called "available bandwidth," can be determined and that emphasis is placed on the ability of each scheme to adapt to the time-varying available bandwidth.

2.2.1 Fixed-Rate Single Layer

While both the MPEG and H.263 video compression standards have scalable extensions that produce a multi-rate video bit-stream, the majority of existing content is not coded using such extensions. Instead, non-scalable or fixed-rate compression techniques are typically employed, making it impossible to change the communication rate after compression has been completed. Inability to adjust communication rate causes (a) under-utilization of network resources when the encoding rate is lower than the available bandwidth, and (b) congestion when the encoding rate is higher [106]. This problem is illustrated in Fig. 2.1 where the solid line shows the time varying available bandwidth and the grey area shows the encoded video rate. Even though the encoding rate for a non-scalable compression scheme cannot be changed after compression, it can be time-varying in general.

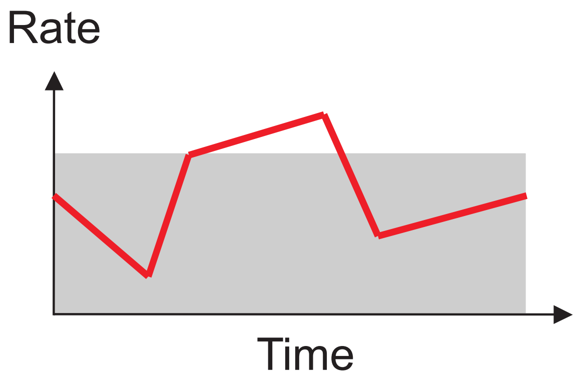


Figure 2.1: The Fixed-rate single-layer scheme produces compressed video whose rate cannot be modified to fit a time-varying channel.

Theoretically, the amount of bandwidth that can be supported by a time-varying channel is related to the notion of *effective bandwidth* [137, 65], and is generally close to the minimum channel rate if low latency is required, and close to the average rate if high latency can be tolerated. In practice, existing schemes such as those of Real Networks Inc. use a ten second or larger buffer, and transmit at a relatively low rate compared to the average channel rate. In such a way, the probability of congestion is reduced, and buffering at the receiver provides additional time for late arriving packets and retransmissions.

One advantage of standard compression techniques that are non-scalable is the high compression efficiency. However, due to the lower communication rate that needs to be assumed, it is questionable whether better end-to-end video quality is achieved. Furthermore, the additional latency associated with buffering has prevented the use of such techniques in an interactive setting where low delay is of utmost importance.

For a multicast environment, even when the available bandwidth does not change with time, the single-layer approach is still ineffective due to bandwidth heterogeneity amongst different receivers.

2.2.2 Stream Replication

To better exploit the available bandwidth using non-scalable compression, one possible enhancement is to provide multiple copies of the same video at different rates and switch between them as needed. This replicated stream or simulcast approach for unicast is illustrated in Fig. 2.2-(a). One example of this approach is the Real Network Inc. SureStream^{TM}. While stream replication provides better adaptability to bandwidth variations than a single non-scalable video stream, it also entails higher storage cost due to the need to store multiple copies of the same video. Generally, with more available copies at different rates, the Stream Replication scheme can better utilize the available bandwidth but the associated storage cost also increases.

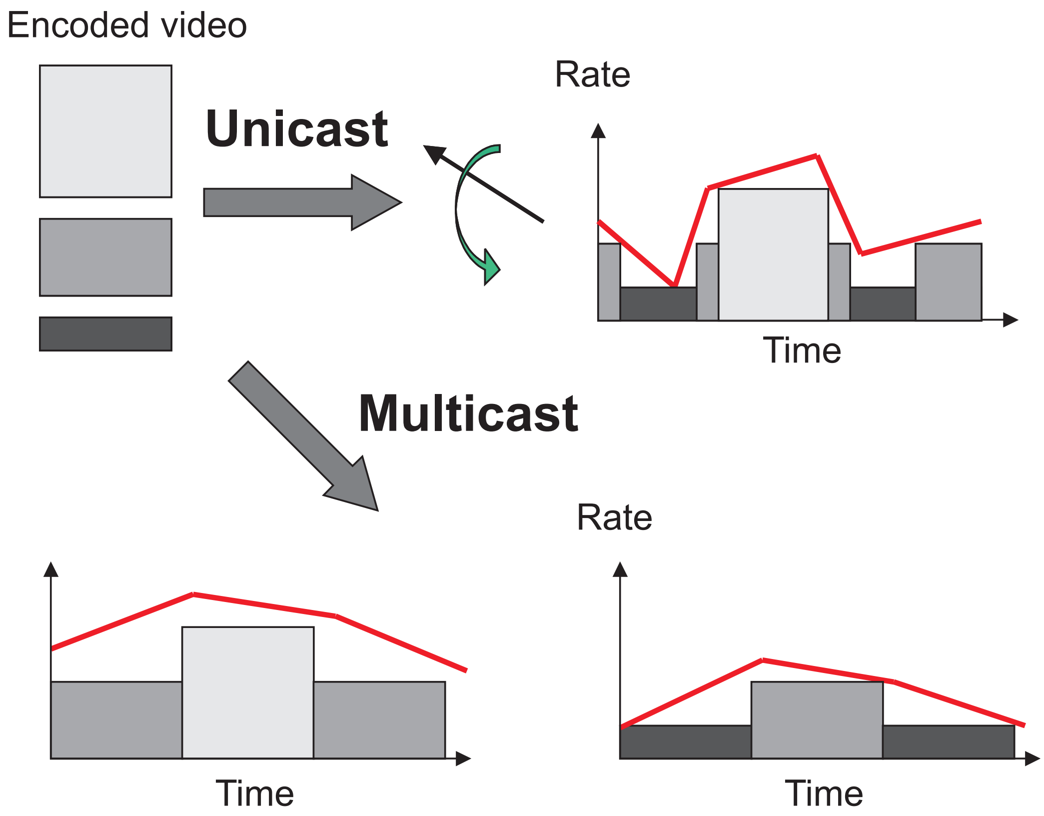


Figure 2.2: The Stream Replication scheme produces multiple copies at different rates for a video. Limited rate control is achieved by switching between the the different copies.

The stream replication approach has also been applied to multicast under the name of Destination Set Grouping (DSG) [19]. Under DSG, receivers are partitioned into groups of similar bandwidth requirement, and a separate stream is served to each group. Furthermore, rate adaptation within each group is possible [19]. In Fig. 2.2-(b), two groups catering to receivers of different bandwidths are shown with slowly time-varying bandwidth. Since data in all groups need to be transmitted simultaneously, the number of groups that can be practically supported is usually limited [75].

2.2.3 Adaptive Single Layer

For video conferencing or broadcasting of live events, video encoding has to be performed on a real-time basis. The availability of a live encoder provides the opportunity of modifying the video encoding rate "on-the-fly" to match available bandwidth [63, 138]. However, such a scheme is only applicable when a live encoder is available. Furthermore, it is a non-scalable approach that does not handle the bandwidth heterogeneity of a multicast environment.

For video conferencing, when there is typically a large stationary background, the use of conditional replenishment with intra-block coding has been proposed to reduce error propagation [85]. For non-conferencing contents however, the scheme generally yields poor compression performance due to the exclusive use of intra-block coding.

2.2.4 Transcoding

Transcoding can be considered an extension to the adaptive single layer approach to pre-compressed video. The idea is to fully or partially decode a video, and then recompress it to meet any specific bandwidth requirements [60, 115, 153]. However, besides quality degradation, transcoding adds latency due to the complex transcoding operation and the additional buffering that may be needed for certain mechanisms of rate reduction [60].

2.2.5 Scalable Video

Scalable video compression is a technique in which compressed video is organized into embedded layers. Typically, the use of more layers results in larger bit rate and also higher video quality in terms of spatial resolution, temporal resolution, or picture quality. By dynamically changing the number of layers used, rate control can be effectively achieved in both the unicast and multicast settings, as illustrated in Fig. 2.3. Multicast rate control is implemented by layered multicast described in Section 2.1.2 in combination with scalable video. In contrast, with the exception of destination set grouping of Section 2.2.2, none of the approaches using non-scalable video can effectively address the bandwidth heterogeneity problem in a multicast environment. The destination set grouping approach however, is hampered by excessive use of bandwidth [75].

Standard compression techniques such as MPEG, and H.263 have scalable extensions that can be used to produce a small number of layers, typically two. However, the small number of available layers limits the use of such scalable schemes for rate control purposes. Instead, schemes that offer fine grained scalability [134, 129, 103] are preferred for their ability to better match time-varying available bandwidth.

One problem with scalable video compression is the general lack of error resilience and thus, the reliance on the network to provide delivery guarantees. In Chapter 3, we present an error-resilient scalable video compression technique that is suitable for use in best-effort packet networks such as the Internet. Specific schemes for using the

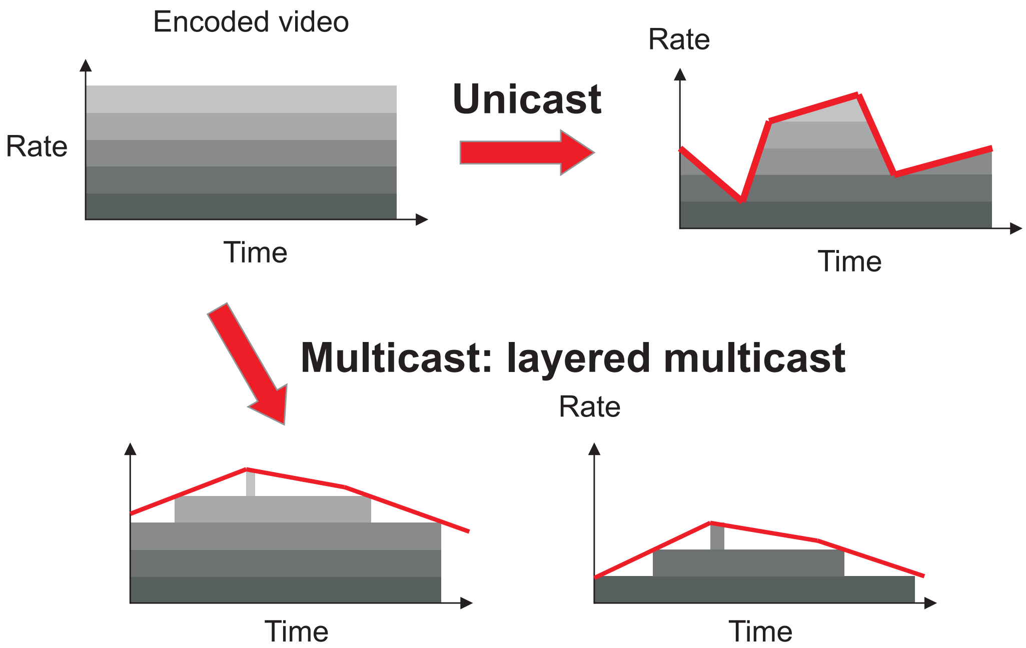


Figure 2.3: A scalable video compression scheme produces an embedded multi-rate stream that is malleable to fit a time-varying bandwidth under unicast and multicast.

error-resilient compression technique in both the unicast and multicast settings are presented in Chapters 4 and 5, respectively.

Chapter 3

Scalable Video for Internet Protocol (SVIP)

3.1 Introduction

Video compression algorithms are traditionally designed for relatively error-free environments. In particular, Shannon's joint source-channel coding theorem states that there is no loss in optimality by performing source and channel coding separately, assuming a stationary and memoryless channel with infinite delay and complexity [116]. However, given the real-time constraint of multimedia communications, the non-stationarity of best-effort packet networks, and the limited computing capabilities of real systems, compression schemes that tailor to the characteristics of the channel may provide better performance. This chapter describes a video compression algorithm, called *Scalable Video for Internet Protocol* (SVIP), which greatly facilitates the rate and error control goals of video communication over best-effort packet networks by being both finely bandwidth-scalable and resilient to random packet losses.

SVIP is based on three-dimensional subband decomposition and data partitioning in the subband coefficient domain, which provides error-resilience, and progressive or multi-rate quantization which provides bandwidth-scalability. A hierarchical block coding technique, described in Section 3.4, is then used to achieve speed and compression efficiency. A scheme based on 3D-subband decomposition is preferred to motion compensated residual coding (MCRC) because MCRC uses previously coded frames as reference for coding future frames which introduce data dependency and thus error propagation when reference frames are lost [85, 72]. In addition, schemes employing 3D subband decomposition have the potential of generating embedded representations with fine granularity of available bit rates [134]. In contrast, scalable compression schemes based on MCRC either employ multiple prediction loops that preclude fine-grain scalability due to the prohibitively high complexity or suffer from drift or low compression efficiency when motion vectors from a base layer are used for coding at higher rates. Furthermore, the computational requirements of 3D subband decomposition is considerably lower than motion estimation, thereby enabling real-time encoding.

Even though three-dimensional subband decomposition has been extensively used for video compression [134, 100, 124, 68], SVIP is unique in its ability to produce individually decodable packets of equal visual importance for transmission over best-effort packet networks. Furthermore, effects of packet loss are dispersed over a wide support and appear as blurring instead of unpleasant missing blocks with abrupt discontinuities. Consequently, relatively constant video quality can be achieved at the receivers even under lossy conditions.

We evaluate the performance of SVIP in several aspects. For error-resilience, we compare the behavior of SVIP against a number of other video compression algorithms under simulated packet losses. Results for actual Internet transmission require accompanying rate control algorithms and are presented later in Chapters 4 and 5 for unicast and multicast, respectively. Encoding and Decoding speed performance and compression performance at zero loss are also reported.

One property of an SNR-scalable video compression scheme such as SVIP is that the picture size and frame-rate remain constant while picture quality varies with varying bit rate. While this characteristic is desirable at high bit rates, it is sometimes preferable at low bit rates to have lower frame-rate but higher quality video frames. To this end, we present a frame-rate scalable extension to SVIP which forces only half of the input frames to be coded at low bit-rates, thereby achieving better picture quality.

The rest of this chapter is organized as follows. We describe how error resilience, bandwidth scalability, and low-complexity is achieved by SVIP in Sections 3.2, 3.3, and 3.4, respectively. Performance of SVIP in terms of compression efficiency, error resilience, and encoding/decoding speeds are presented in Section 3.5. Finally, a frame-rate scalable extension to SVIP is presented in Section 3.6 followed by a conclusion.

3.2 Error Resilience

Bandwidth-scalable video compression schemes are often designed with transport prioritization in mind and produce packets that are highly inter-dependent [47, 28, 45, 134, 124, 135]. In particular, it has been shown [47, 28, 45] that by separating the source material into a base layer and an enhancement layer and by applying priority queuing [28, 45] or different loss probabilities [47], results superior to the one layer approach are obtained for statistically multiplexed networks. However, due to the lack of prioritized transport in best-effort packet networks, packet losses may appear random, undermining the potential benefits of the layered scheme [5]. For example, SNR-scalable video compression algorithms [47, 28, 45, 134, 124, 135] produce packets that are *linearly* dependent, i.e., N packets are produced for every K video frames so that if packet i is lost, error propagation would prevent decoding of packets $i+1,...,N$. While this approach works well in ATM-type networks where end-to-end quality of service guarantees can be enforced, it is unsuitable for transmissions over best-effort packet networks because the lack of prioritized transport causes packet losses to appear random, resulting in large variations in received video quality.

To see why linearly dependent packets yield large variations under random loss, consider a channel with an independent packet loss rate of p. The probability that we can decode exactly i packets out of a total of N transmitted packets is then $(1-p)p^i$ for $i \neq N$, and $(1-p)^N$ for $i = N$, a bimodal distribution that is geometric but with a tall spike at $i = N$. The distribution for $N = 20$ and $p = 0.1$ is shown in Fig. 3.1. Over 70% of the time we can either decode all 20 or at most 6 packets, resulting in large variation in the quality of the received video. Furthermore, despite a packet loss rate of only 10%, Fig. 3.1 indicates that there are only 10 decodable packets on average out of 20 transmitted packets. This is due to the linear dependency of packets since correctly received packets are not decodable if their dependent packets are lost.

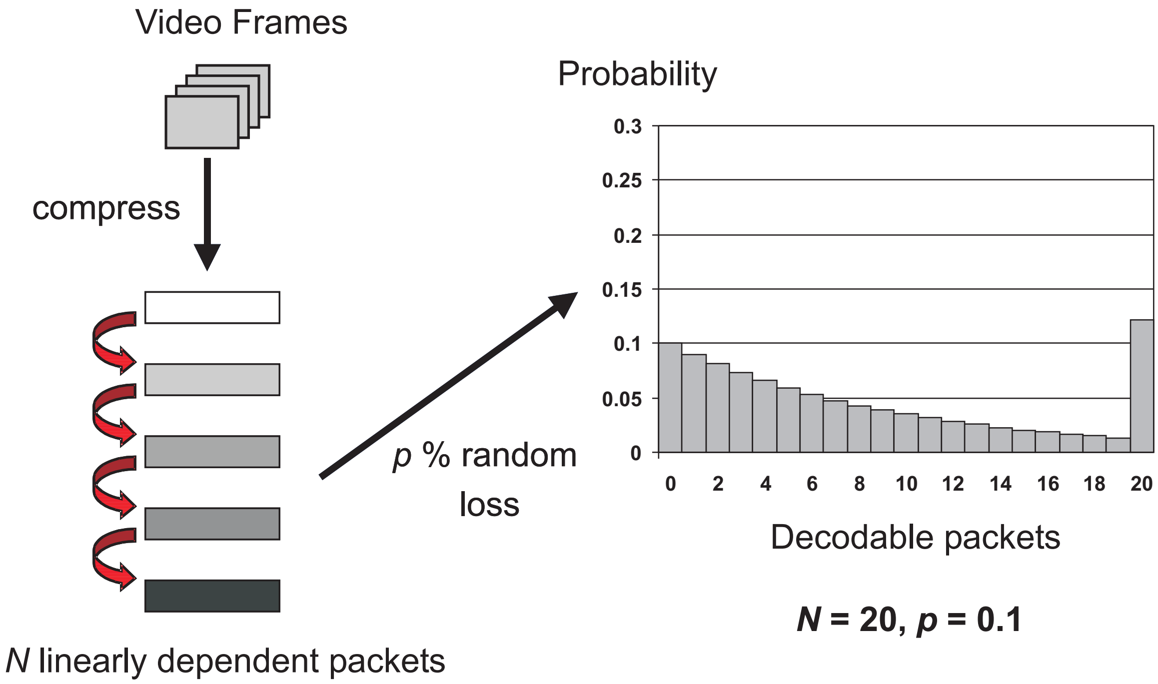


Figure 3.1: Distribution of the number of decodable packets when 20 linearly dependent packets are subjected to 10% random loss.

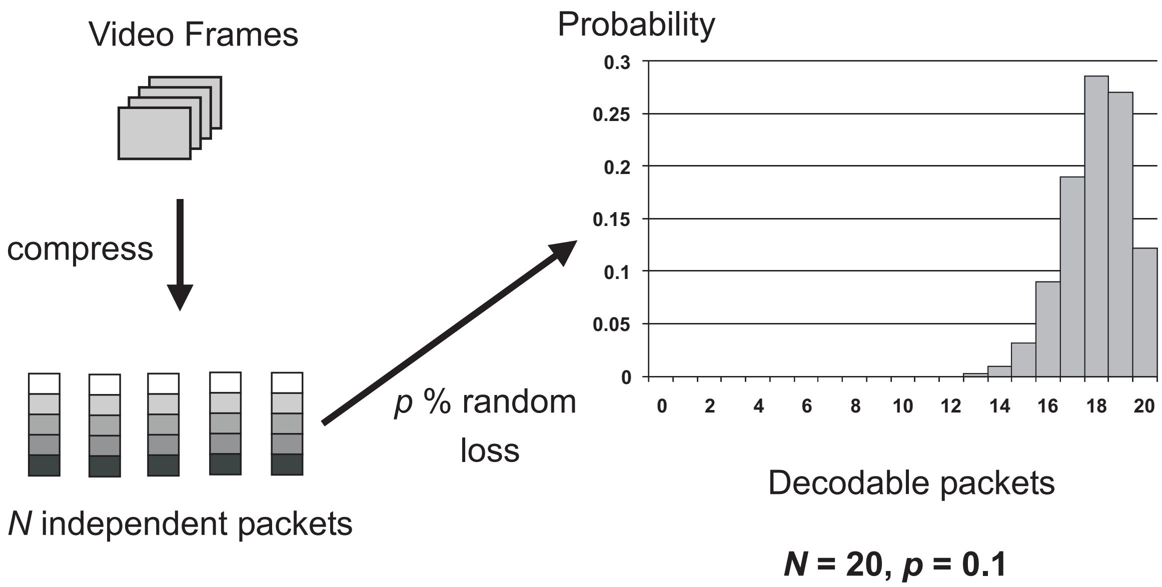


Figure 3.2: Distribution of the number of decodable packets when 20 individually decodable packets are subjected to 10% random loss.

Every packet must be individually decodable or independent to eliminate error propagation. Fig. 3.2 shows the distribution of decodable packets when 20 independent packets are subjected to a random 10% loss. A binomial distribution with an average of 18 decodable packets and a small variance of 1.8 is obtained. One way to achieve individually decodable packets is to employ a forward decomposition of the source material into M *components* and then compress each component independently to form a packet. Each packet can then be decoded to a co-image where the sum of all co-images form the original image. The use of this concept in a networking application is shown in Fig. 3.3. However, the requirement to produce independently decodable packets generally results in lower compression efficiency.

There are a number of such decompositions for still images. One example is the

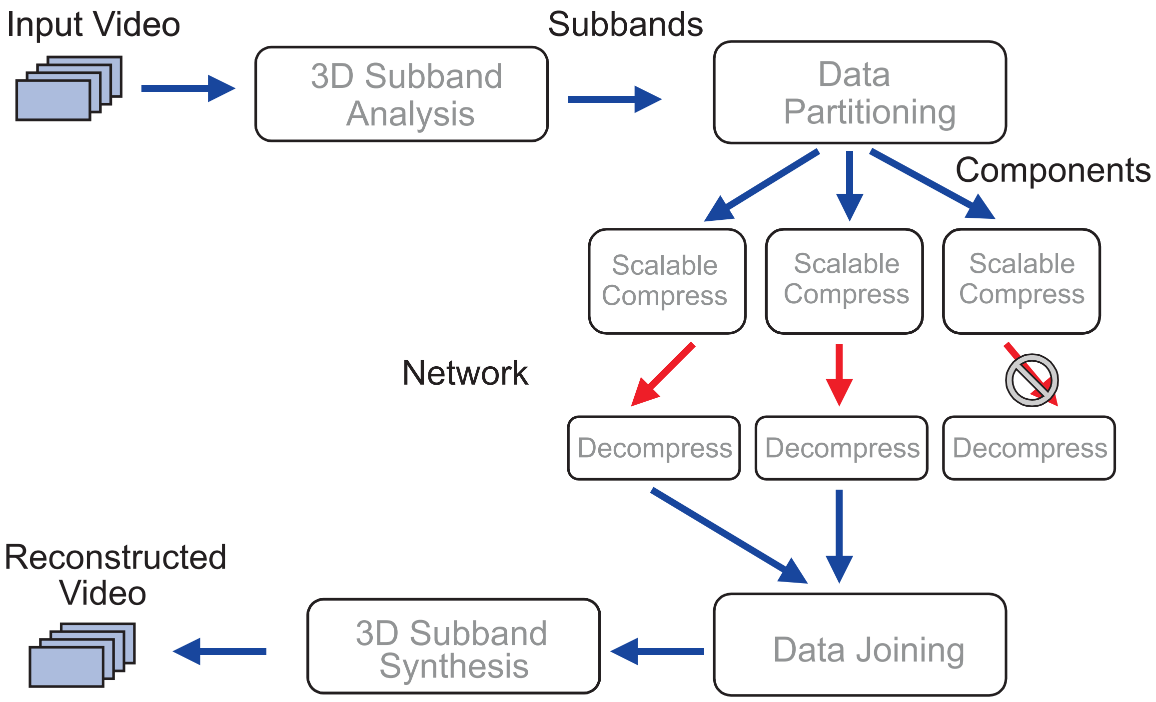


Figure 3.3: Using individually decodable packets in lossy networks.

polyphase decomposition which takes every M consecutive pixels and distributes one pixel to every component. Each component then would be individually decodable and approximately of equal importance. This scheme suffers from low compression efficiency. Another approach is to use block-based coding in the pixel domain. However, when one packet contains all information about a spatial location, its loss will cause all information in that location to be lost. Yet another approach is to use subband decomposition to divide the source into subbands that can be compressed independently. However, the low-frequency subbands contain most of the energy for natural images. If each subband goes into a packet, this skewness would cause large variation in decoded picture quality under lossy conditions.

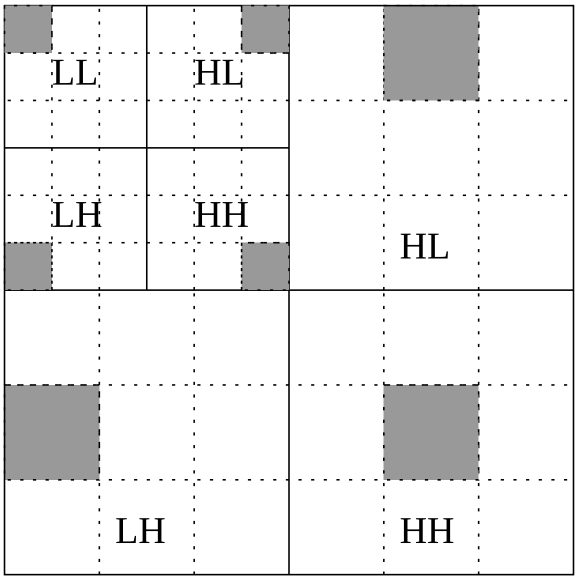


Figure 3.4: Grouping coefficient blocks from different subbands to form a component.

To overcome the problems of the above approaches, we use an alternative data partitioning scheme for subband-based compression: instead of making each subband a component, we partition each subband into an equal number of coefficient blocks. Each coefficient block in a subband carries information about some localized region in the original frames. The components are then formed by grouping from each subband, an equal number of coefficient blocks that correspond to different spatial regions of the source. As a still image example, Fig. 3.4 shows the formation of one component out of a total of nine. Since there are seven subbands, it would take at least seven packet losses to completely eradicate a particular spatial region. Conversely, a single packet loss contributes to at most one frequency band in any spatial location.

To extend the framework from still image to video, one possible way is to use 2D subband decomposition with motion compensation. However, since motion compensation and prediction is unsuitable for finely scalable video, and also introduces error propagation, a scheme based on 3D subband coding is used [134, 124]. A component then consists of coefficient blocks of different spatial locations from the set of spatio-temporal subbands.

3.3 Bandwidth Scalability

In this section, we will describe packetization of compressed information into a rate-scalable structure. Rate-scalability is desirable in real applications not only because of its usefulness in rate control, but also because it provides complexity scalability whereby receivers with limited decoding power can reduce data that is being decoded.

Instead of achieving rate-scalability by skipping components, we compress each component into a multi-rate representation. Rate-scalability is then obtained by modulating the size of the compressed representation. The scheme is depicted in Fig. 3.5 where $R_3 > R_2 > R_1$. This scheme is in contrast to traditional packetization of layered video where the inter-dependency and varying importance of the packets often require transport prioritization [45, 135].

Each component contains coefficient blocks that are independently compressed using progressive quantization followed by hierarchical block coding [124] described in more detail in Section 3.4. When subband coefficients are represented using signed-magnitude format, progressive quantization reduces to bit-planing with an appropri-

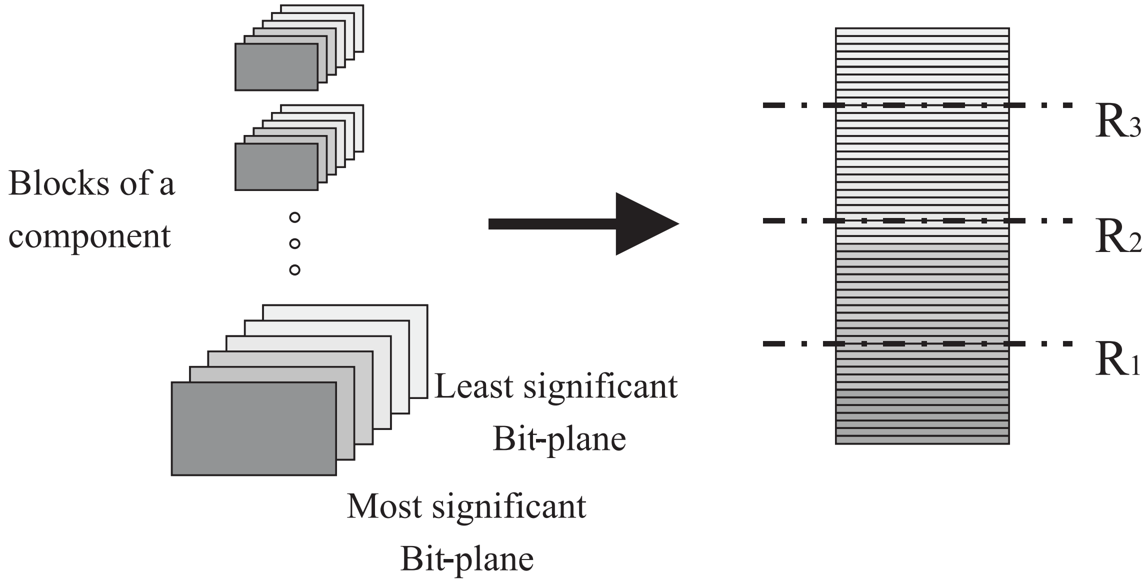


Figure 3.5: Achieving fine grained rate-scalability by stacking small compressed codewords of bit-planes.

ate choice of quantizer. Successive bit-planes of a coefficient block are then inter-coded to produce small codewords which provides fine granularity for layered packetization.

In order to produce good quality for all the rates R_1, R_2 and R_3, we require packetization to be carried out so that data is roughly packed in the order of importance, as illustrated in Fig. 3.5. Because hierarchical block coding exploits correlations between bit-planes of the same coefficient block, the more significant bit-planes must be packed before the less significant ones. Given that constraint, our packetization strategy then forms an ordering of all the bit-planes in all coefficient blocks in a component. However, there is in general no bit-plane packing scheme that simultaneously minimizes distortion at all rates. One practical heuristic is the greedy approach: at any one time, we examine every coefficient block in a component to find the most

Description	No. bits	min value	max value
T_{LL}/T_{LH}	5	1	100
T_{LH}/T_H	3	0.5	7
S_{LH}/S_{HH}	5	1	32
S_{HL}/S_{HH}	5	1	32
S_{LL}/S_{HH}	6	0.5	2048
$S_{LL}/(S_{LH}+S_{HL}+S_{HH})$	2	1	8
δ	3	1/8	24
$\log_2(\alpha)$	3	9	16

Table 3.1: Parameters for the energy distribution of subbands. T_{LL} and S_{HL} stand for the energies of the temporally low frequency bands and the horizontally high but vertically low frequency bands respectively. δ is the decay factor for the spatially high-frequency subbands (e.g., the ratio of energies in the two HL subbands in Fig. 2) and α is the energy density of the DC subband.

significant bit-plane that has not already been packed. We then choose from the set of most significant bit-planes the one that offers the most expected energy reduction per bit, and pack the entire bit-plane. However, coding the decision of the packing order as described above incurs large overhead since one decision is needed per bit-plane per coefficient block. Instead, we take an approximate approach as follows. Every time the encoder finishes subband analysis, it computes and parametrizes the energy distribution of the subbands into 8 parameters totaling 32 bits which can then be compactly transmitted per packet. The parameters and their explanations are given in Table. 3.1 and its caption. Both the decoder and the encoder then reconstruct from the parameters an approximate energy distribution which is used for estimating the next bit-plane that would yield the best rate-distortion performance. After a bit-plane is compressed, its actual energy contribution is used to update the approximate energy distribution both at the decoder and the encoder. The parameters described above indicate the relative importance of different subbands averaged over all spatial locations, and serve as an initial estimate for both the encoder and the decoder. Since each component contains only one coefficient block corresponding to only one spatial location in each subband, the energy distribution of the coefficient blocks within a component may be different from that of the subbands. As a result, the update is necessary to exploit the spatial variation of energy within a subband.

A schematic diagram for the procedures at the decoders and encoders are shown in Figs. 3.6 and 3.7 respectively.

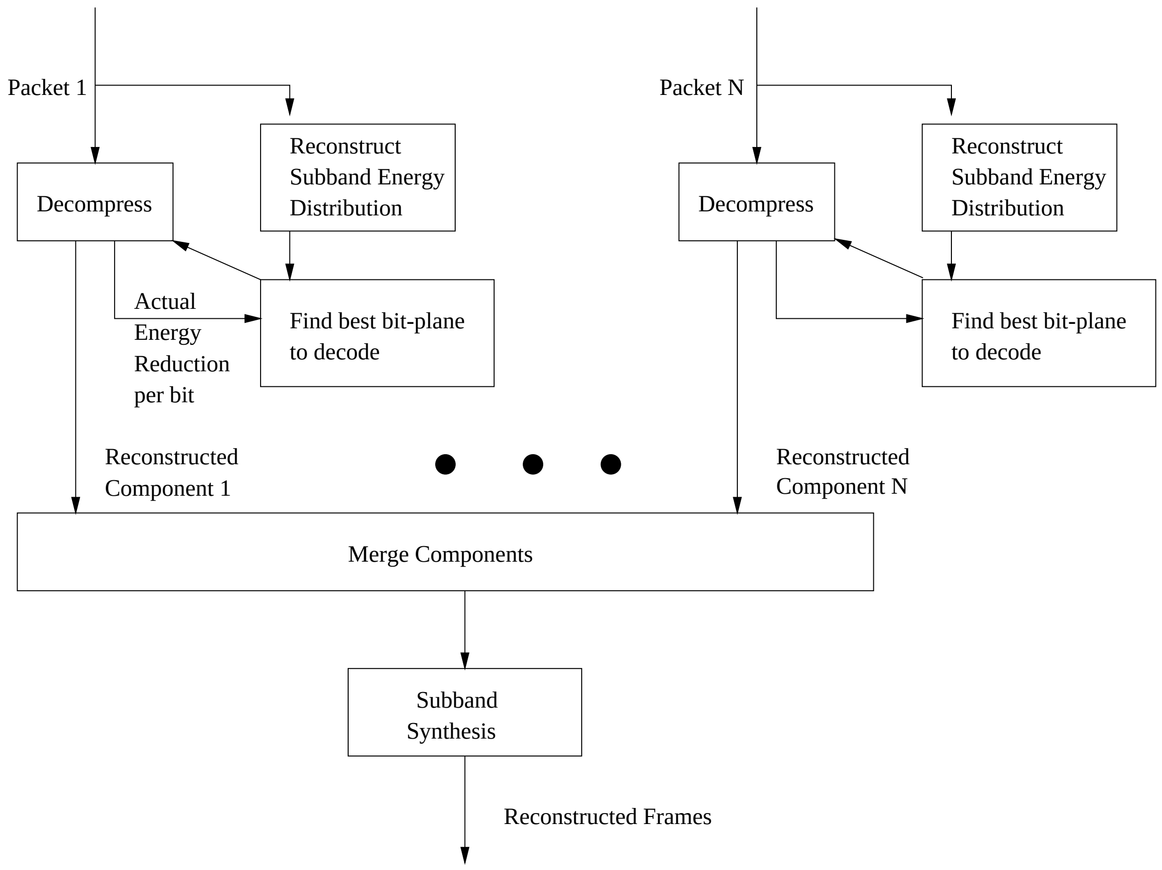


Figure 3.6: Schematic for SVIP decoding.

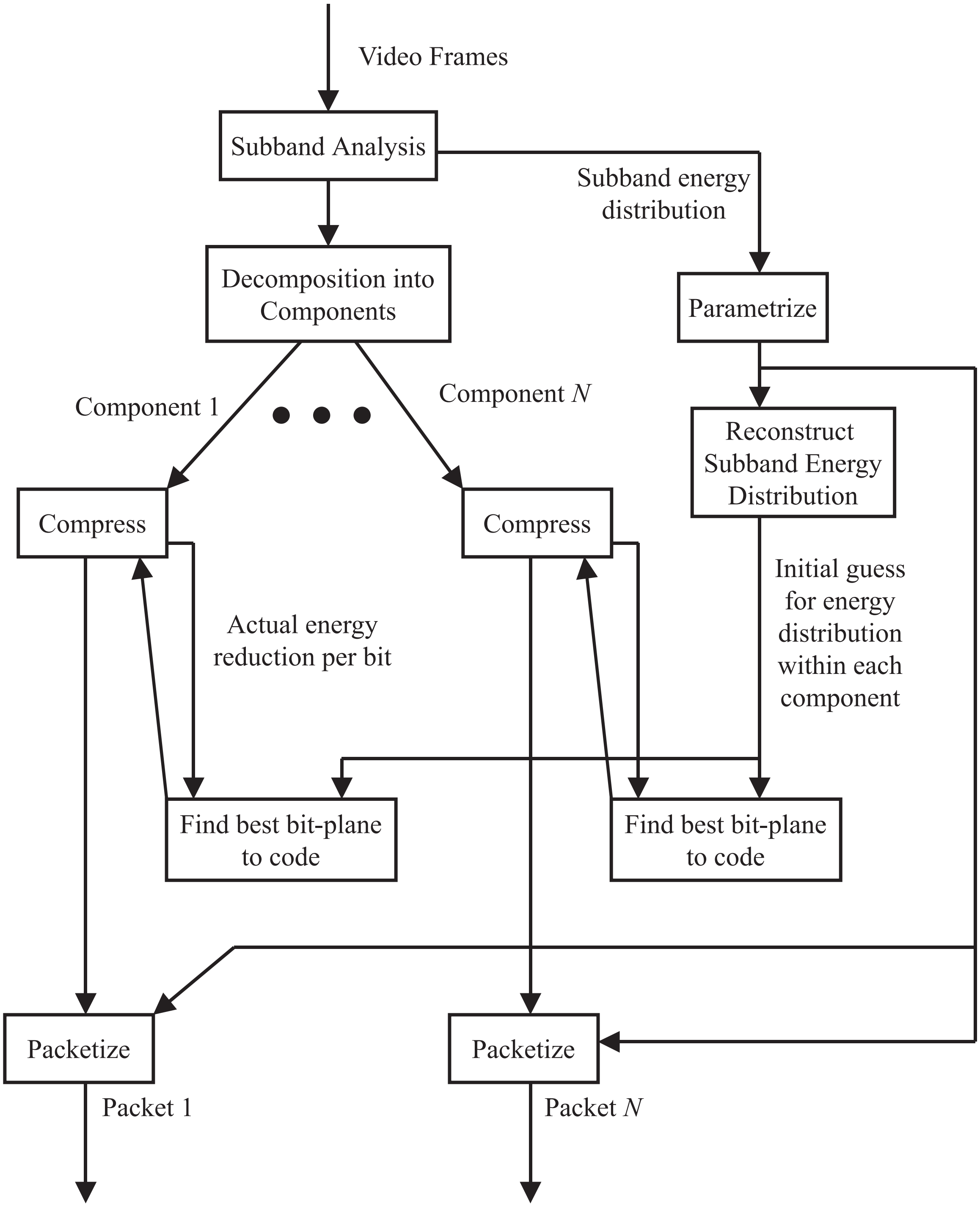


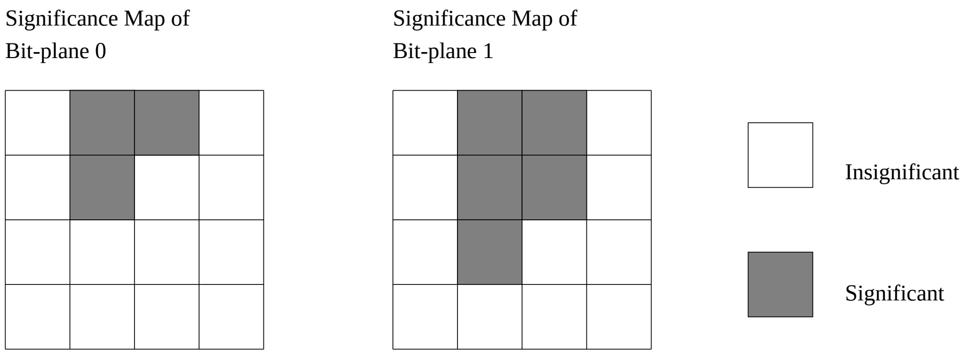
Figure 3.7: Schematic for SVIP encoding.

3.4 Real-time Implementation using Hierarchical Block Coding

This section describes the application of hierarchical block coding (HBC) [124] to real-time scalable video based on 3D subband coding. HBC is a fast and efficient way to exploit correlations between bit-planes of subband coefficients. This approach was chosen at the expense of coding efficiency because arithmetic coding based schemes previously used for scalable video coding [134] are often too compute-intensive for real-time applications. For example, in decoding video of size 320×240 coded at 1.5 *Mbps* using the scheme of [134], it is found that over 70% of the overall complexity can be attributed to arithmetic coding.

In coding every bit-plane, we maintain a *significance map* which is a binary map indicating whether each coefficient is significant or non-zero if we stop at the current bit-plane. When a coefficient is significant in a certain bit-plane, we will transmit its binary value uncoded. They are not coded to reduce both the encoding and decoding complexity because little compression gain is achievable by coding them. The significance map with the values are sufficient to reconstruct the bit-planes.

We code the significance map of the first bit-plane with significant elements using a quad-tree based approach [71]. Because significance maps for subsequent bit-planes can only grow, we skip information that can be inferred from previous significance maps. An example for a 4×4 block is illustrated in Fig. 3.8. The codeword for the first significance map starts with an “1” indicating that the 4×4 block has significant values. The next 4 bits “1100” shows whether each of the 2×2 sub-blocks contain significant values, starting with the top-left sub-block and counting clockwise. For each 2×2 sub-block containing significant values, 4 bits are added to indicate which 1×1 sub-blocks are significant. To code the significance map for the next bit-plane, we observe from the previous significance map that the 4×4 block as well as the first two 2×2 sub-blocks contains significant values. This information can therefore be skipped, improving compression performance. The effectiveness of HBC is based on the fact that coefficients in the high frequency subbands are often quantized to zero so that quad-tree based approaches which assign short codewords to blocks with many zeroes are efficient.




```
Layer 0 Bits: 1 1100 0101 1000
Layer 1 Bits: - --10 0-0- -010 0100
```

Figure 3.8: An example of coding significance maps using hierarchical block coding. “-” indicates values that can be deduced.

A frequently used operation for quad-tree coding is testing whether a block contains significant values. Since the significance map is binary, it takes only 16 bits to represent the significance map of a 4×4 block. Thus, it takes only one 16-bit compare to check if a 4×4 block is significant and only two 32-bit compares for an 8×8 block. As a result, HBC admits a low complexity implementation.

3.5 Performance of SVIP

In this section, we evaluate the performance of SVIP in terms of its compression efficiency and encoding/decoding speed performances. We will also compare its performance against several other compression methods when subjected to random packet loss.

3.5.1 Compression Performance

Since SVIP is required to produce independently decodable packets and be bandwidth scalable, it is necessarily more restrictive than compression schemes that do not satisfy those criteria. In particular, it lacks the motion models of MPEG and H.263 [59] and does not exploit correlation between subbands, as is done in SPIHT [112] or Shapiro's embedded zero-tree scheme [117]. Furthermore, correlation between coefficient blocks within the same subband is also deliberately ignored. As a result, there is in general a decrease in compression efficiency when there is no packet loss.

We compare the Peak Signal-to-Noise Ratio (PSNR)[1] for two sequences: "Raider of the Lost Ark," and "Mother and Daughter," with 600 and 300 frames respectively.

[1]The PSNR of a video frame is a function of the Mean Square Error (MSE) between the original

The *Raider* sequence is a high motion fighting scene at 24 fps whereas *Mother* is a low motion head and shoulder sequence at 30 fps. The image size for both sequences is 320×224. We perform PSNR comparisons of our scheme with H.263, MPEG-1, and MPEG-4 fine grained scalability (FGS) [103][2] using group of picture (GOP) sizes of 4, 8 and 16. We do not consider longer GOP patterns because our intended application is transmission over lossy networks where long GOP patterns are more susceptible to error propagation due to packet losses.

An important issue to deal with when comparing codecs is bit allocation. Our approach is to ensure that all frames are coded by each codec and that the overall bit rates are identical. Specifically, we generate the H.263[3] results using fixed quantization parameters (QP) across all frames. The resulting bit-rate is then controlled by changing QP in the range [1, 31]. When the desired bit-rate cannot be reached by changing QP alone, the input sequences are subsampled to result in lower frame rates. This input subsampling occurs when we try to compress *Raider* using GOP sizes of 4 and 8 and *Mother* using GOP size of 4 at a rate below 128 *kbps*. The MPEG results are generated using the Berkeley MPEG-1 encoder [49] and decoder [96].[4] We used

frame and the reconstructed frame:

$$PSNR = 10 \log \frac{255^2}{MSE}$$

[2]MPEG-4 FGS uses a non-scalable MPEG-4 base layer and many fine granularity enhancement layers.

[3]The software from the University of British Columbia is used.

[4]Further information about the Berkeley MPEG encoder/decoder and other MPEG tools is available from http://bmrc.berkeley.edu/frame/research/mpeg

the Berkeley encoder with ten slices per frame and exhaustive search to encode the video. Target bit-rates are set so as to match those produced by H.263. The MPEG-1 encoder dynamically adjusts the compression quality of each frame using its own rate control mechanism, which is essentially based on a leaky bucket algorithm described in [14].

For SVIP and MPEG-4 FGS, we use the same number of bits per GOP instead of more elaborate bit allocation algorithms. This is because for our intended networking applications, the number of bits used per GOP is dictated by the rate control algorithm to be described later in Chapter 4 for unicast and Section 5.6 for multicast. Thus, the number of bits used by SVIP and MPEG-4 FGS is identical on a per GOP basis, while that of SVIP, MPEG-1 and H.263 are identical only on a per sequence basis.[5] Given a fixed bit budget for a GOP, we propose two ways to distribute bits among the components. The simpler approach is the constant bit rate (CBR) allocation in which each component receives the same number of bits. With variable bit rate (VBR) allocation, the compressed bits of each component are divided into fixed size cells and distortion tags representing the energy reduction of each cell are stored. A greedy algorithm for energy reduction is then used to allocate bits among the components. Since the distortion tags are not needed to decode the components but are only used to perform bit allocation, they do not need to be transmitted for applications such as video playback over networks where the video server is perform-

[5] It is not possible to have identical bit usage for all schemes on a per GOP basis due to the different rate control mechanisms implemented in the H.263 and MPEG-1 software.

ing the rate allocation and filtering. We use 40 bytes for a cell and 2 bytes for a distortion tag, which are found experimentally to work well. This yields a 5% storage overhead in using VBR allocation.

The results of comparing the four codecs using the *Mother* and *Raider* sequences are shown in Figs. 3.9 and 3.10 respectively. When using SVIP and MPEG-4 FGS, one embedded bit-stream is produced for every frame rate. Subsets of the bit-stream are then extracted to decode a video at the same frame rate but at a different bit rate. For H.263 and MPEG-1, a separate bit-stream is generated for every frame rate and every bit rate. The PSNR curves for SVIP are shown in red, with solid and dotted lines for CBR and VBR bit-allocation respectively. PSNR curves for non-scalable compression schemes are shown in blue, with solid and dotted lines for H.263 and MPEG-1 respectively. Finally, the PSNR curves for MPEG-4 FGS are shown in black. A particular meaningful comparison is that between SVIP with CBR bit allocation and MPEG-4 FGS since both are finely scalable and with similar bit allocation strategies. To highlight the comparison between the two schemes, their PSNR curves are shown in thicker lines in Figs. 3.9 and 3.10. We see that the performance of SVIP is more or less identical in the CBR and VBR cases with the largest differences being at the lower bit rates. The gain in compression performance of using SVIP with VBR over SVIP with CBR is shown in Fig. 3.11. We see that at bit-rates higher than 500 *kbps*, the difference is typically very close to zero. Generally, H.263 outperforms MPEG-1, MPEG-4 FGS and SVIP for almost all rates to which

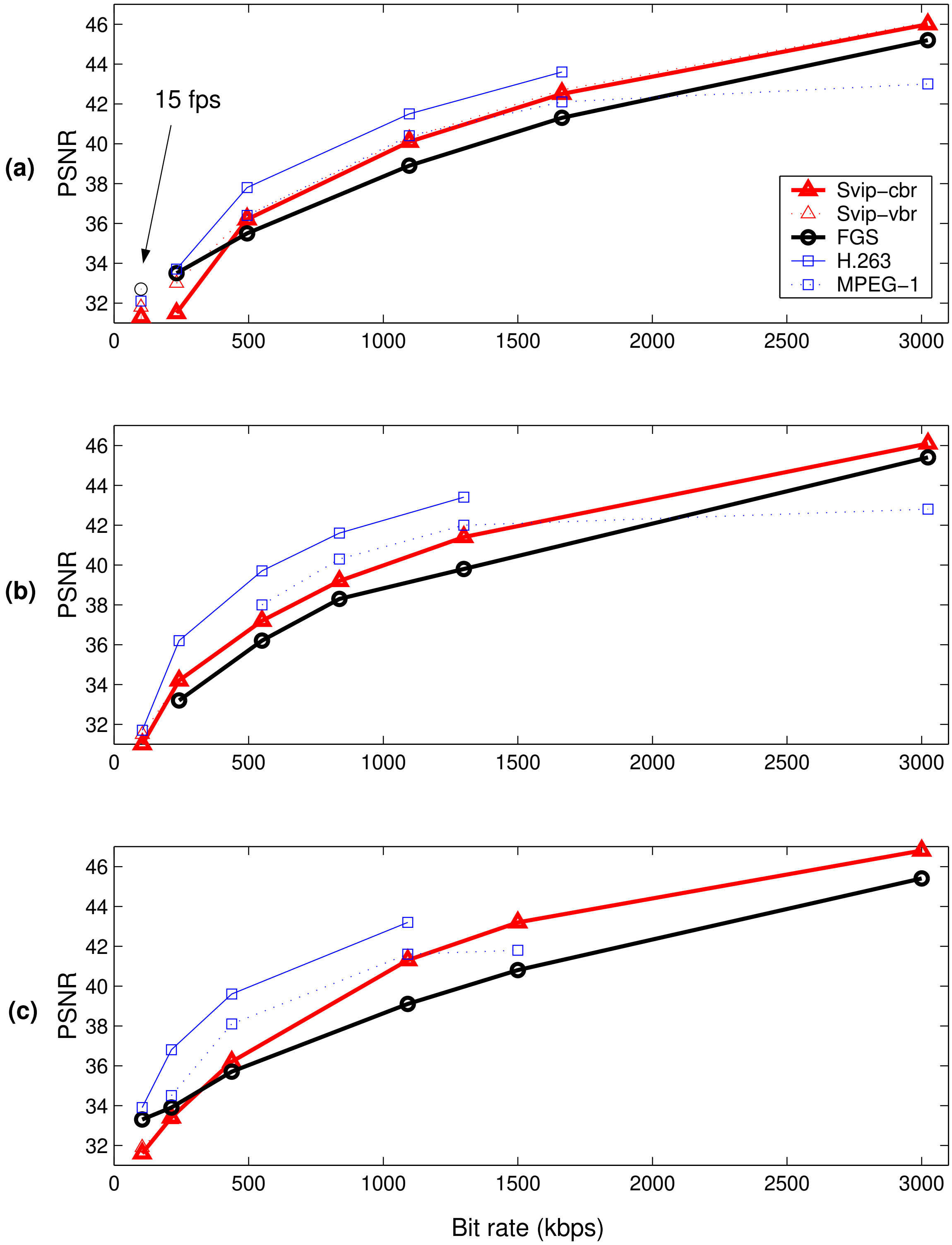


Figure 3.9: PSNR comparison for the *Mother* sequence using GOP sizes of (a) 4, (b) 8, and (c) 16. Frame rate is 30 fps unless otherwise noted.

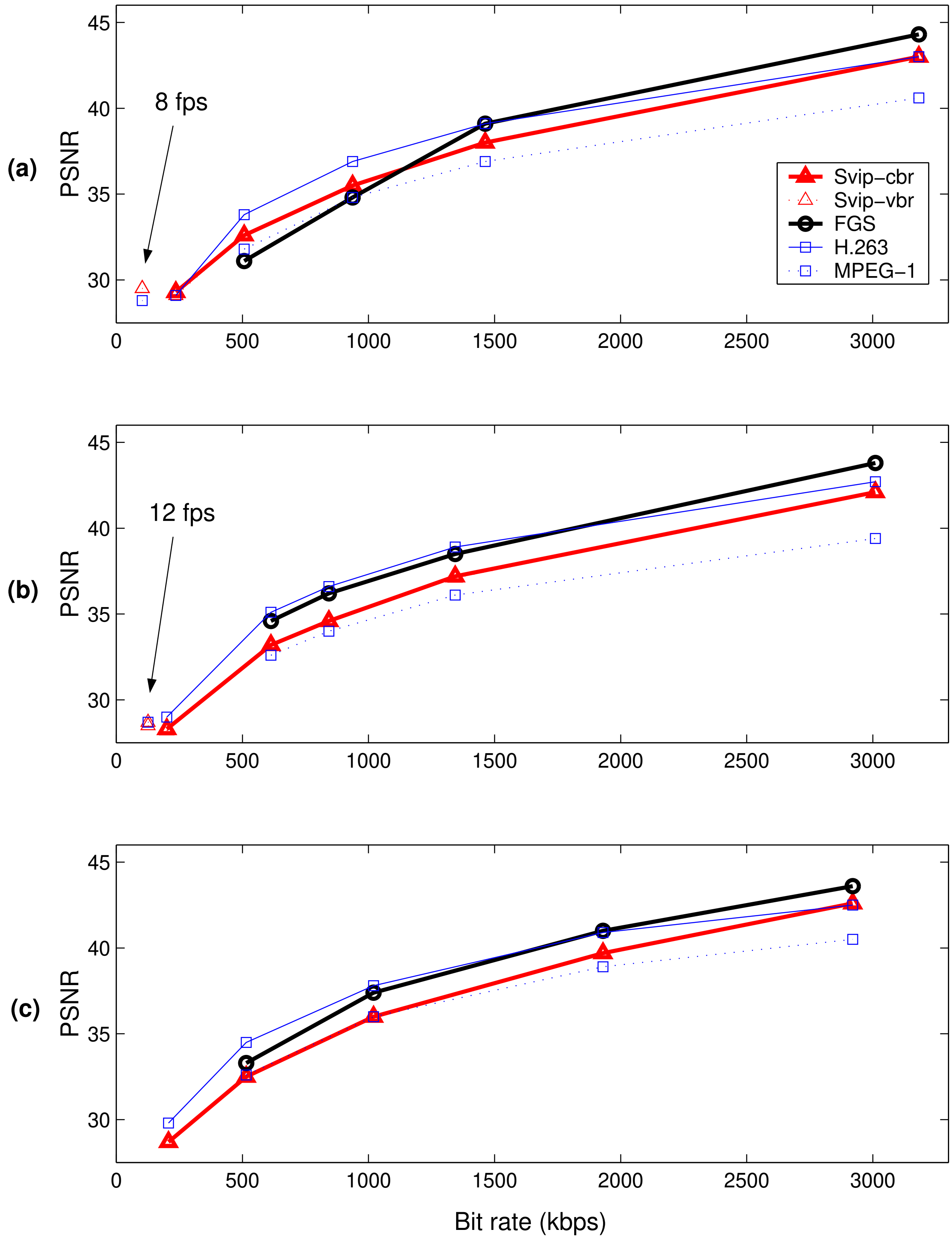


Figure 3.10: PSNR comparison for the *Raider* sequence using GOP sizes of (a) 4, (b) 8, and (c) 16. Frame rate is 24 fps unless otherwise noted.

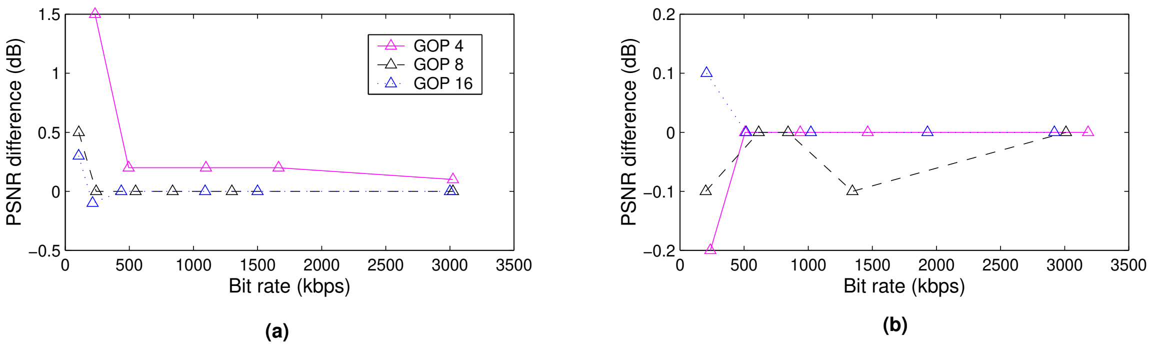


Figure 3.11: Gain in compression performance of SVIP with VBR over SVIP with CBR bit allocation for (a) *Mother* and (b) *Raider*.

it can be applied.[6] The gain in compression efficiency of H.263 over SVIP is shown in Fig. 3.12. We observe that the gap between H.263 and SVIP generally increases with

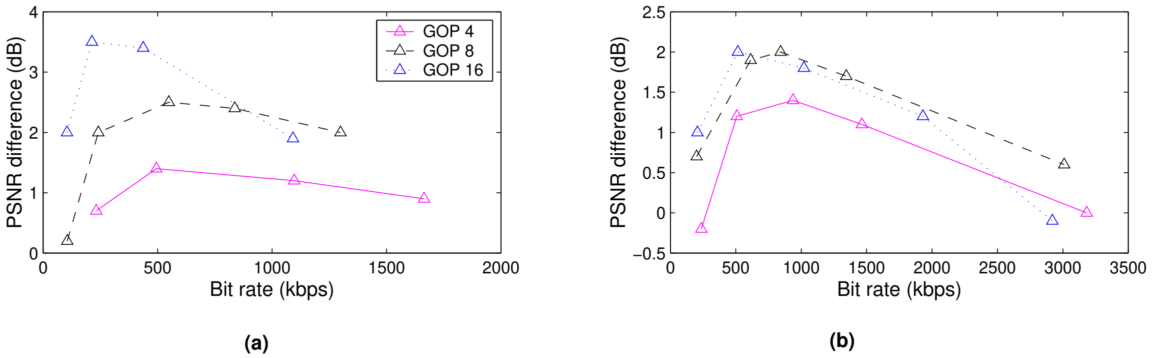
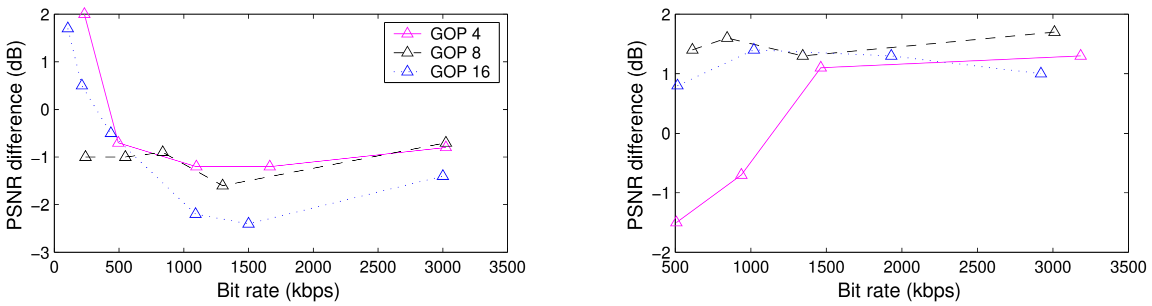


Figure 3.12: Gain in compression performance of H.263 over SVIP for (a) *Mother* and (b) *Raider*.

increasing GOP size due to the lack of a motion model in SVIP. Furthermore, the gap increases at first as bit-rate increases, but eventually decreases as bit-rate further increase beyond about 1 $Mbps$. For GOP sizes of 4, 8 and 16, the maximum difference is 1.2, 2.4 and 3.4 dB respectively for *Mother*, and 1.4, 2.0 and 1.9 dB respectively for *Raider*. The mild performance penalty is expected due to the requirements to

[6]For *Mother*, no quantization step size yields bit rates between 2-4 $Mbps$



(a) (b)

Figure 3.13: Gain in compression performance of MPEG-4 FGS over SVIP for (a) *Mother* and (b) *Raider*.

produce bandwidth scalability, independently decodable packets for error resilience, and a real-time implementation.

We next compare the compression performance of the two scalable compression schemes SVIP with CBR bit allocation and MPEG-4 FGS. Both schemes are finely scalable and have similar bit allocation strategies. The PSNR gain of MPEG-4 FGS over SVIP with CBR bit allocation is shown in Fig. 3.13 for the *Mother* and *Raider* sequences. We observe that they have comparable compression efficiency, with SVIP performing better than MPEG-4 FGS for the low-motion *Mother* sequence but worse for the high-motion *Raider* sequence. Specifically, for bit-rates above 1 $Mbps$, SVIP is about 1 to 2 dB better than MPEG-4 FGS for the *Mother* sequence, but is about 1 to 1.5 dB worse in the *Raider* sequence. We further compare the compression performance of SVIP and MPEG-4 FGS using five additional MPEG-4 test sequences described in Table. 3.2. The results for the PSNR comparisons for the five sequences using GOP sizes of 4, 8, and 16 are shown in Figs. 3.14 to 3.18. Again, for low-motion sequences such as *Container* and *Silent*, we see from Figs. 3.14 and 3.15 that SVIP generally performs better than MPEG-4 FGS. Specifically, the performance

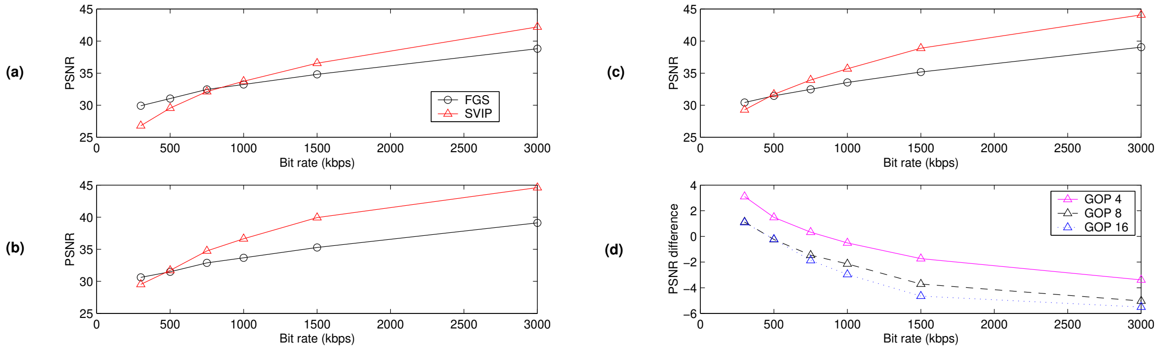


Figure 3.14: Compression performance of MPEG-4 FGS and SVIP for *Container* using GOP size of (a) 4, (b), 8 and (c) 16. Compression gain of FGS over SVIP is shown in (d). Base layer sizes for FGS are 212, 125 and 82 *kbps* for GOP size of 4, 8 and 16 respectively.

gain of SVIP over MPEG-4 FGS is between 0 to 3 dB at 1 $Mbps$ and between 3 to 6 dB at 3 $Mbps$. The *News* sequence consists of two low-motion news reporters and a large background screen showing high-motion contents. As shown in Fig. 3.16, the compression performance achieved by SVIP and MPEG-4 FGS is similar, with the difference less than 2 dB in most cases. Finally, for the high motion sequences

Name	Description
Container	A slowly moving container ship.
Silent	A person communicating in sign language with a fixed background.
News	News report with a large screen showing high motion contents.
Foreman	A scene with complex motion.
Mobile	A highly complex scene in both the amount of motion and texture.

Table 3.2: Descriptions of addition test sequences for comparing the compression performance of SVIP and MPEG-4 FGS. All sequences are of size 352×288 except *Mobile*, which is of size 320×224. The frame rate for all sequences is 30 fps.

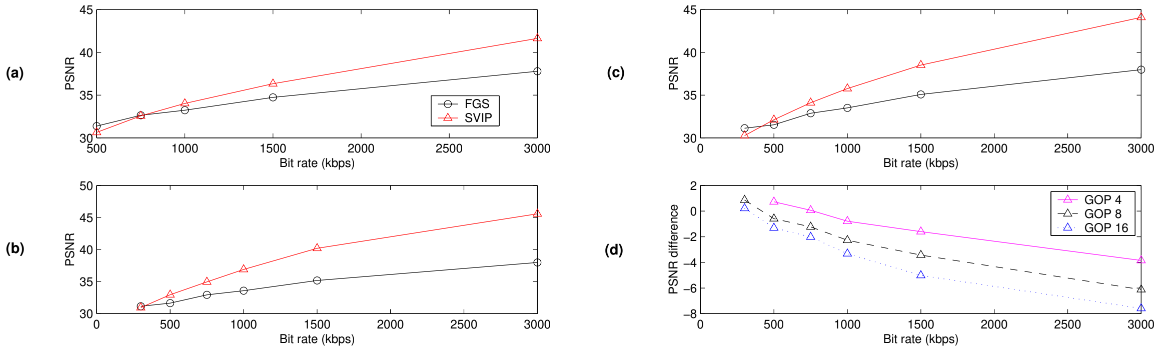


Figure 3.15: Compression performance of MPEG-4 FGS and SVIP for *Silent* using GOP size of (a) 4, (b), 8 and (c) 16. Compression gain of FGS over SVIP is shown in (d). Base layer sizes for FGS are 250, 175 and 137 *kbps* for GOP size of 4, 8 and 16 respectively.

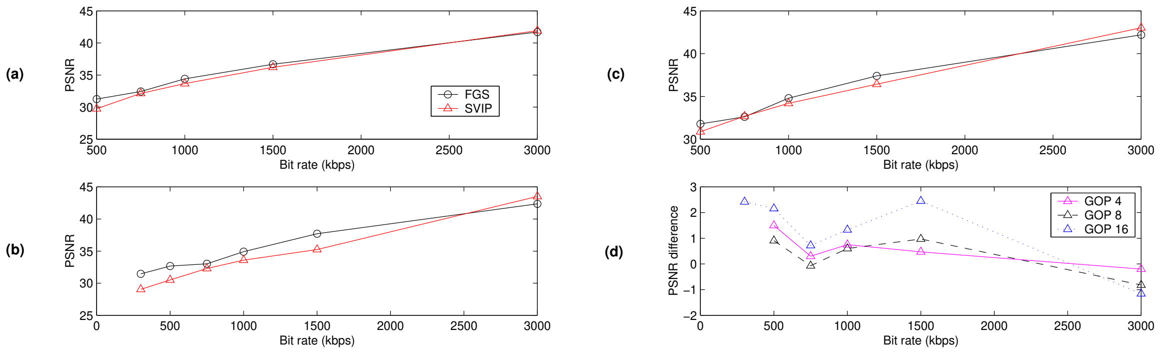


Figure 3.16: Compression performance of MPEG-4 FGS and SVIP for *News* using GOP size of (a) 4, (b), 8 and (c) 16. Compression gain of FGS over SVIP is shown in (d). Base layer sizes for FGS are 252, 161 and 112 *kbps* for GOP size of 4, 8 and 16 respectively.

such as *Foreman* and *Mobile*, we see from Figs. 3.17 and 3.18 that MPEG-4 FGS performs better than SVIP. Specifically, the compression gain of MPEG-4 FGS over SVIP is about 2 dB for bit-rates about 1.5 $Mbps$ for *Foreman* and 2 $Mbps$ for *Mobile*, and decreases as bit-rate increases. For MPEG-4 FGS, inter-frame correlation is only exploited at the base layer, but not at the refinement layers. For video sequences with a large stationary background such as the *Mother*, *Container* and *Silent* sequences, it

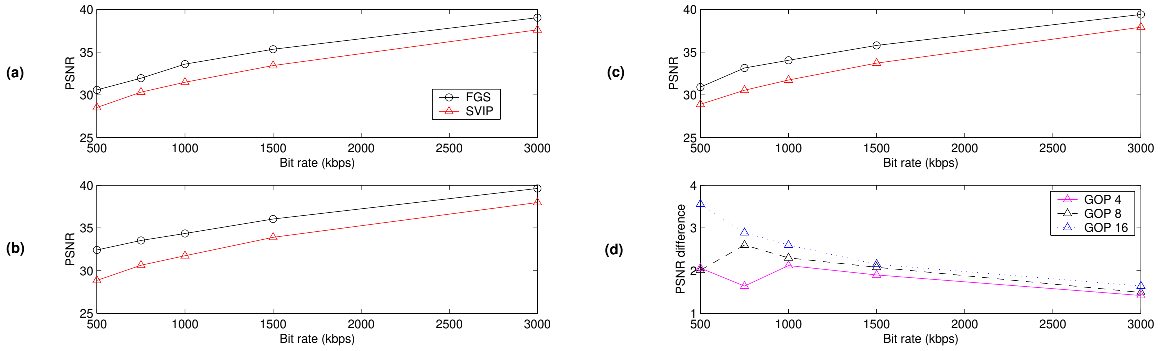


Figure 3.17: Compression performance of MPEG-4 FGS and SVIP for *Foreman* using GOP size of (a) 4, (b), 8 and (c) 16. Compression gain of FGS over SVIP is shown in (d). Base layer sizes for FGS are 342, 274 and 238 *kbps* for GOP size of 4, 8 and 16 respectively.

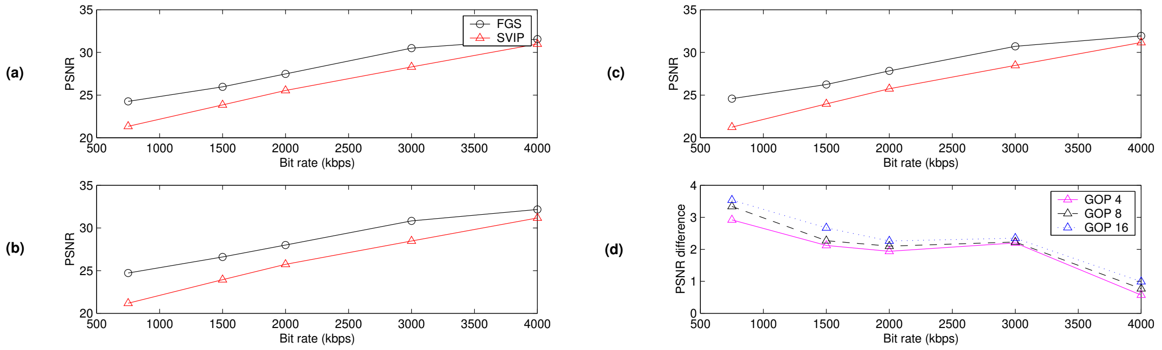


Figure 3.18: Compression performance of MPEG-4 FGS and SVIP for *Mobile* using GOP size of (a) 4, (b), 8 and (c) 16. Compression gain of FGS over SVIP is shown in (d). Base layer sizes for FGS are 538, 403 and 333 *kbps* for GOP size of 4, 8 and 16 respectively.

is expected that the failure to exploit any temporal correlation in the FGS refinement layers results in significant loss of compression efficiency. On the other hand, for relatively high motion sequences such as the *Raider*, *Foreman* and *Mobile* sequences, the loss of compression efficiency is expected to be relatively small. When generating MPEG-4 FGS bit-streams, we always choose the lowest quantization parameter of 31 for the base layer to obtain the biggest range of available bandwidth. The base layer size for the *Mother* sequence at 30 fps is 134, 90, and 69 *kbps* for GOP size of 4, 8, and 16 respectively. The *Mother* at 15 fps using a GOP size of 4, the base layer size is 75 *kbps*. For the *Raider* sequence, the base layer size is 265, 257, and 251 *kbps* for GOP size of 4, 8, and 16 respectively. The base layer size for the sequences in Table 3.2 is shown in the captions of Figs. 3.14 to 3.18.

3.5.2 Performance under Loss

We next compare the performance of SVIP under 5% simulated packet loss. Four other schemes are considered: (F) MPEG-4 fine-grain scalability, (M) MPEG-1, (S) 3D subband coding where every packet contains one subband and, (T) the scalable compression scheme of [124] which produces linearly dependent packet. For MPEG-4, the base-layer is encoded using an error-resilience option known as video packet [104] to avoid loss of synchronization due to packet losses. Reversible variable length codes (RVLC) are not applied since it is intended for environments with bit-errors rather than congestion losses. Packetization of the MPEG-1 bit-stream is performed so that no slice is split across packets unless the slice size exceeds packet size [56]. A GOP size of 4 is used for all cases except MPEG-1, where GOP sizes of 2 and 15 have been used. The results are shown in Fig. 3.19. We see that only SVIP enjoys a uniform high quality of received video. Even though packets under scheme (S) are independent, the skewness in their energy causes large variations in received video quality. Schemes (T), (M), and the base layer of scheme (F) suffer from error propagation and show

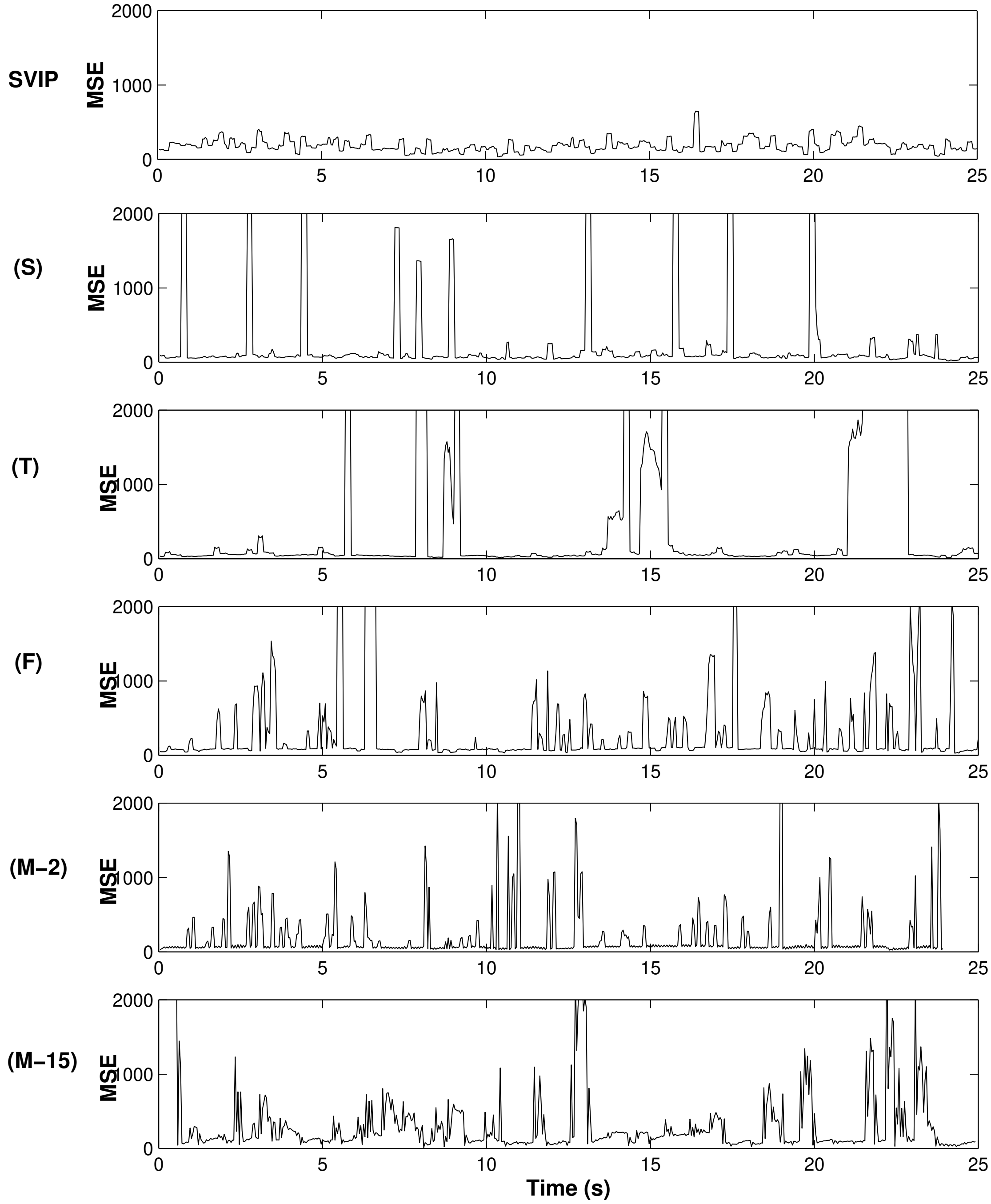


Figure 3.19: Variation of quality at 5% simulated random packet loss for different video compression schemes. (M-2) and (M-15) denote MPEG-1 with GOP size of 2 and 15, respectively.

even larger variations. In particular, we can see from Fig. 3.19 that MPEG errors have a longer "tail" with a longer GOP pattern because of motion compensation. For SVIP, simple error concealment is performed on the DC-subband where every missing coefficient is estimated by the average of its surviving neighbors. Similar concealment techniques are not applicable to scheme (T) in which all spatial locations in the DC subband are compressed and transmitted together.

3.5.3 Speed Performance

For the 600 frame *Raider* sequence of size 320 $\times$ 224 at 24 fps, the encoding speeds on a 400 MHz Pentium-II are 34 to 21 frames per second in the bit-rate range of 200 *kbps* to 2 *Mbps* respectively. The decoding speeds in the same bit-rate range varies from 60 to 25 frames per second. The reported times exclude disk access and display time [124].

Fig. 3.20 shows the speed comparison of SVIP and that of the Berkeley MPEG-1 decoder [96] on an 170 MHz UltraSparc-1. For the *Raider* sequence at bit rates ranging from 500 *kbps* to 3 *Mbps*, the SVIP encoding rate is 18.2 to 11.2 fps and the decoding rate is 17.5 to 12.5 fps, respectively. On the same machine the MPEG encoding rate is 0.4 to 1.6 fps using exhaustive and logarithmic search, respectively. The reported speed excludes disk access and display times. We see that real-time software-only video encoding and decoding of SVIP is possible on today's computers.

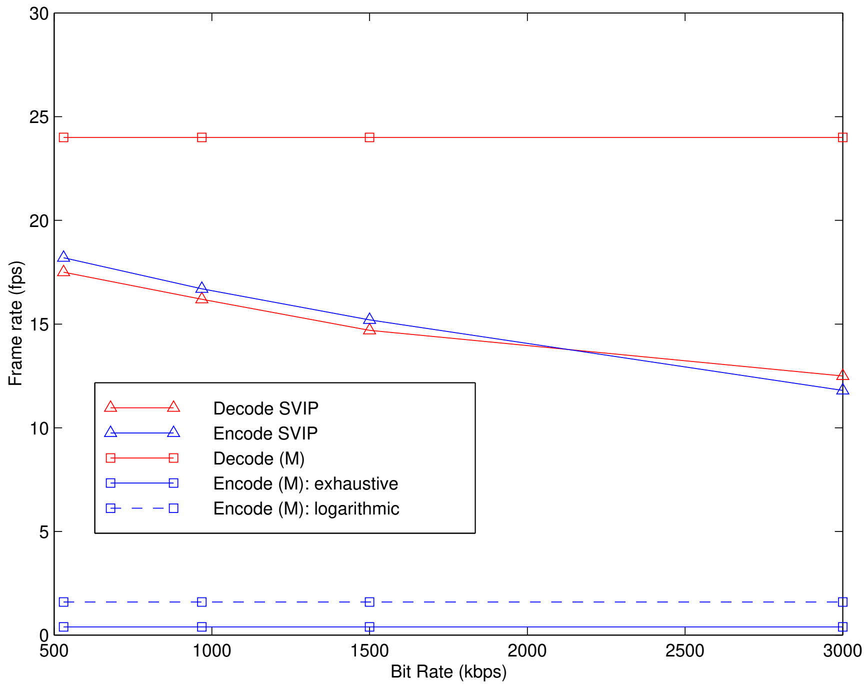


Figure 3.20: Speed comparisons of SVIP and MPEG-1 (M) on an 170 MHz Ultra-sparc. “Exhaustive” and “logarithmic” refer to the different strategies in motion estimation of MPEG.

3.5.4 Visual Quality

To evaluate the visual quality of SVIP, we compress every four frames of the *Raider* sequence into 70 independently decodable packets of size 450 bytes using SVIP. This yields a total compressed bit-rate of 1.5 $Mbps$ for the sequence. The packets are then subjected to 20% random loss. The original video frames 5, 25, and 55 are shown in Fig. 3.21-(a) and the reconstructed frames after 20% loss are shown in Fig. 3.21-(b). Simple error concealment is performed on the DC-subband where every missing coefficient is estimated by the average of its surviving neighbors. We see that errors are dispersed over a wide support and while the image is uniformly

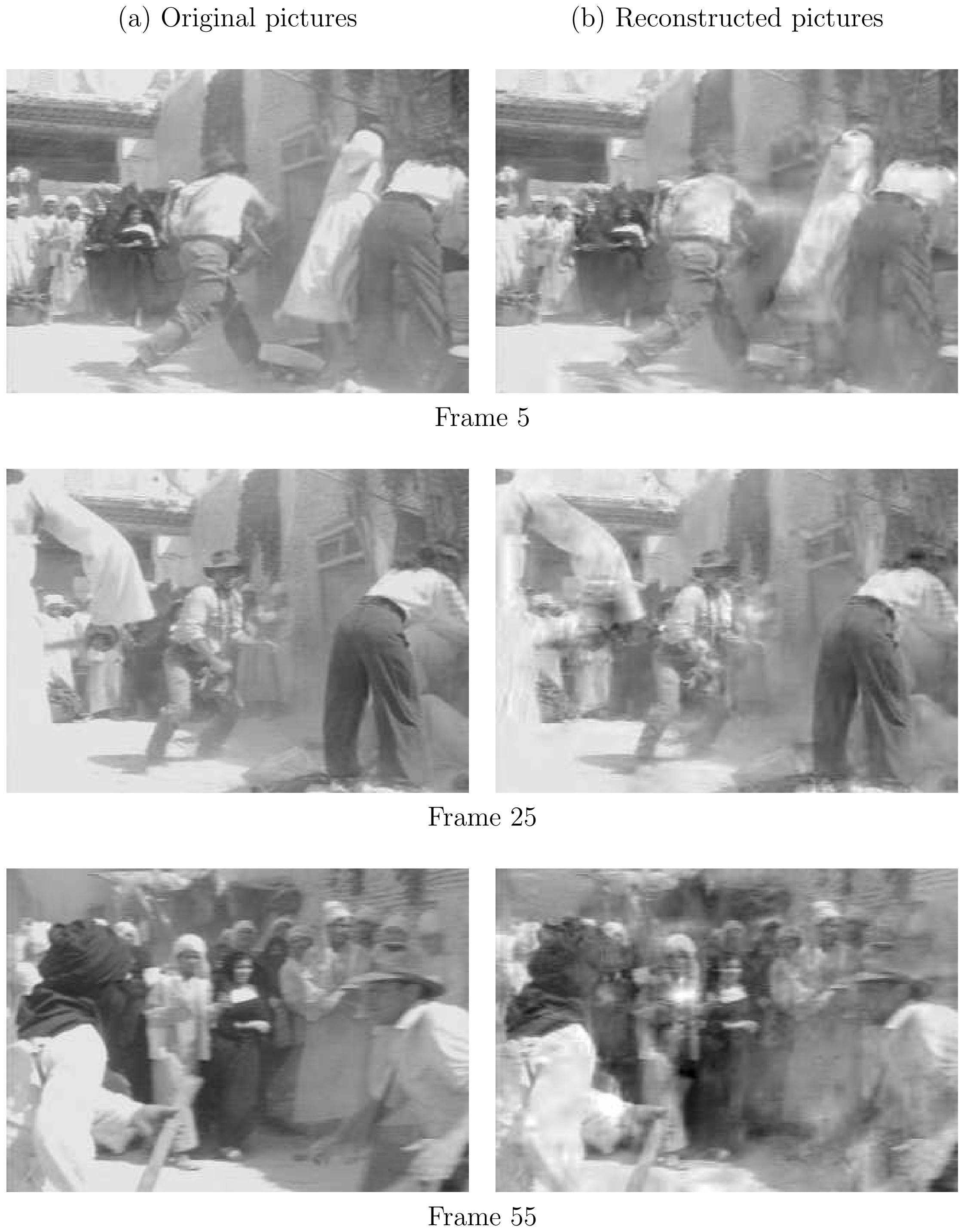
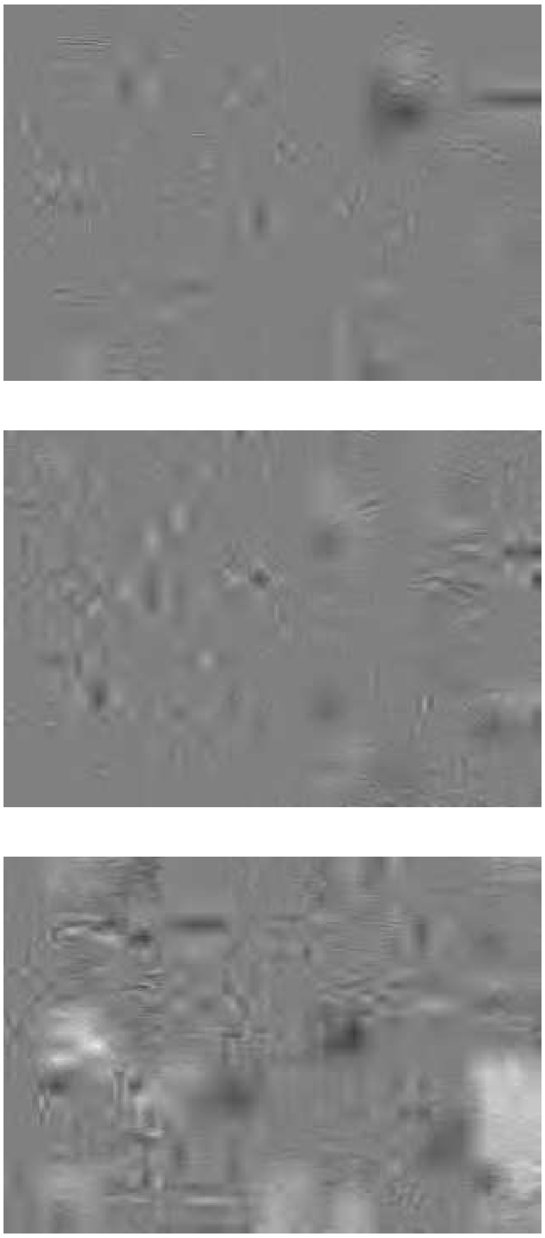


Figure 3.21: Frames 5, 25 and 55 of (a) the original *Raider* sequence, and (b) *Raider* sequence compressed at 1.5 *Mbps* and then subjected to 20% random loss.



Frame 5

Frame 25

Frame 55

Figure 3.22: Difference images between video frames reconstructed with no packet loss and with 20% loss.

blurred and the total energy is diminished, all features of the original image are still visible. Furthermore, even though block based coding is employed, there are no sharp discontinuities because data partitioning is performed in the frequency domain instead of the pixel domain. Fig. 3.22 shows the difference images between the video frames reconstructed with no packet loss and with 20% packet loss. On a gray-scale between 0 to 255, a gray level of 128 represents no difference, while lighter and darker gray levels correspond to larger errors. We can see from Fig. 3.22 that there is most error in frame 55 and least error in frame 5 but in all cases the error is dispersed throughout the image to yield smooth looking pictures.

3.6 Extension to Frame-rate Scalability

In this section, we present a frame-rate scalable extension to SVIP to obtain higher picture quality at low bit-rates at the cost of reduced frame rate. In a 3-D subband framework, a lower frame-rate approximation of the original sequence can be obtained by the temporally low-pass subbands. However, one drawback to such an approximation is that, even at very high bit rates, the reconstructed pictures will still be different from the original frames [26]. For example, when the Haar filter is applied to two video frame F_1 and F_2, the low frame rate video reconstructed from the temporally low-pass subbands only will asymptotically approach the averaged frame $(F_1 + F_2)/2$, rather than frames F_1 or F_2, as bit rate increases.

One approach to achieve frame rate scalability is as follows. First, we sub-sample the input frames by 2 and then apply regular 3-D subband coding to compress the odd frames of the original sequence up to a predetermined rate R. When the bit-rate reaches R, we perform 3-D subband filtering on the even frames as well. Instead of coding the odd and even frames as two independent threads, better compression performance can be obtained by predictively coding the temporally low subbands of even frames from the reconstructed temporally low subbands of the odd frames at rate R. At bit rates above R, bits can be allocated to any subbands. In such a way, videos at full and half frame rates are free from temporal subband filtering distortion. A schematic diagram for the encoding procedures is shown in Fig. 3.23. The scheme can

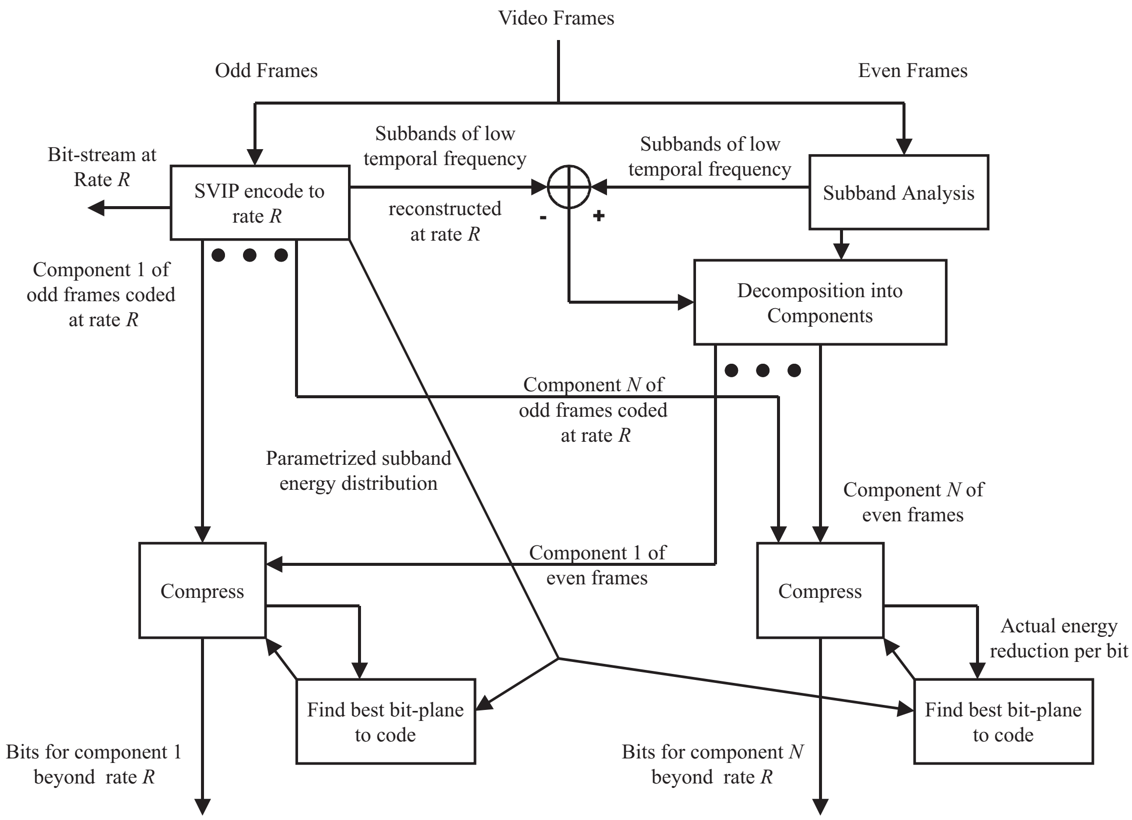


Figure 3.23: Schematic for SVIP encoding using the frame-rate scalability extension.

be straight-forwardly extended to obtain more frame rates at the cost of increasing complexity. Fig. 3.24 shows a PSNR comparison for the *Raider* sequence at size 320×224 using SVIP and the temporal scalability extension of SVIP. For SVIP, the full frame-rate of 24 fps is maintained at all bit-rates. For SVIP with frame-rate scalability extension, an R of 250 $kbps$ is used so that only 12 fps is compressed at rates below 250 $kbps$. We see that significantly better picture quality, often exceeding 2 dB, can be obtained at rates below 250 $kbps$ when frame-rate scalability extension is used. However, the penalty is the reduced frame-rate at low-bit rates, and the slightly lower compression efficiency, usually within 1 dB, at rates higher than 250 $kbps$.

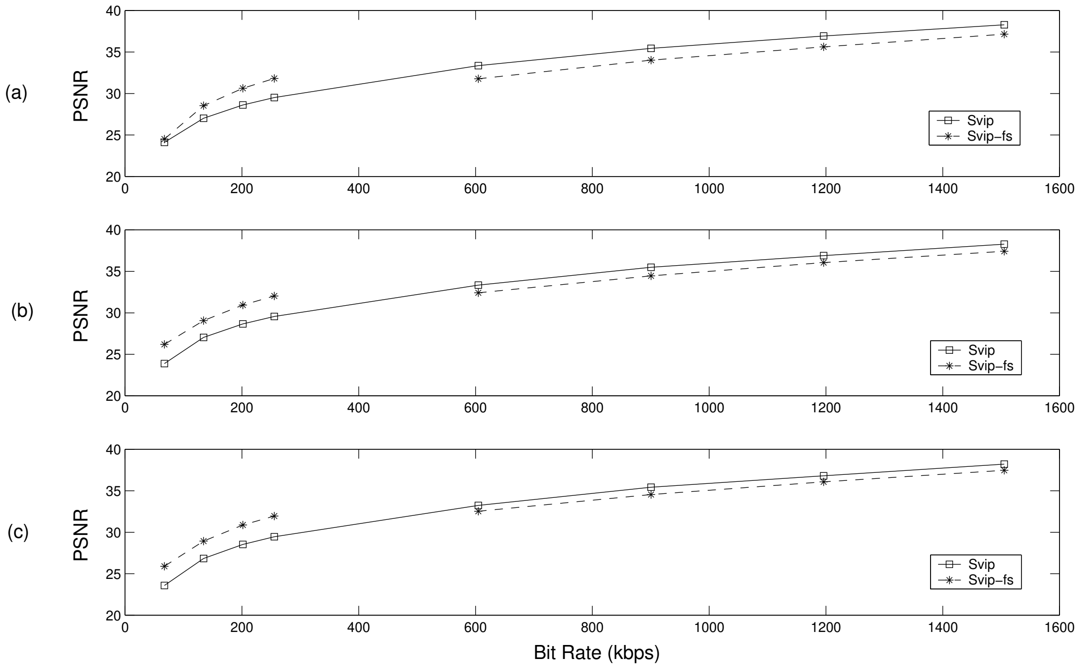


Figure 3.24: PSNR comparisons for *Raider* sequence using SVIP and the frame-rate scalable extension of SVIP (Svip-fs) using GOP size of (a) 4, (b) 8, and (c) 16 respectively.

3.7 Conclusions

In this Chapter, we have presented an error-resilient and bandwidth-scalable compression method called SVIP. We have evaluated SVIP in terms of compression performance with and without loss, and encoding/decoding speed. SVIP is found to yield significantly better video in terms of lower distortion and smaller variation in quality when there is packet loss. The cost of this error-resilience property is a slight loss in compression efficiency when packet losses are absent. We have also demonstrated that a real-time SVIP encoder and decoder can be implemented on current workstations. Finally, we presented a frame-rate scalable extension to SVIP that produces higher quality pictures at low bit-rates at a small cost in reconstructed image quality at higher bit rate.

Chapter 4

Unicast Streaming using SVIP and TCP-friendly Rate Control

This Chapter introduces a unicast video transmission scheme over best-effort packet-switched networks combining a low-delay TCP-friendly transport protocol and the error-resilient and bandwidth-scalable SVIP compression method described in the previous Chapter. Under SVIP, video is packetized into individually decodable packets of equal visual importance. Consequently, relatively constant video quality can be achieved at the receiver under lossy conditions. Furthermore, packets can be truncated to instantaneously meet the time-varying bandwidth imposed by a TCP-friendly transport protocol. As a result, adaptive flows that are friendly to other TCP traffic are produced. Actual Internet experiments together with simulations are used to evaluate the performance of the combined scheme.

4.1 Introduction

There are two main requirements for video communications over best-effort packet networks: (1) bandwidth adaptability and (2) error-resilience. Lack of bandwidth adaptability results in non-adaptive streams with two sets of disadvantages. First, it leads to congestion collapse when the aggregate bandwidth of the video traffic exceeds network capacity [61]. Second, it competes unfairly with other adaptive traffic, such as TCP, which reduces transmission rate in face of network congestion. Existing approaches to rate control for video traffic over best-effort packet networks often require online encoding. By dynamically changing the parameters of the encoder, it is then possible to limit the data rate [138, 63]. For pre-encoded video, Jacobs and Eleftheriadis [60] have proposed a scheme in which selective DCT coefficients from MPEG streams are dropped to yield lower rates. However, both schemes are typically unable to match the desired rate instantaneously so that output buffering becomes necessary, adding to total latency. Another possible approach to rate control is transcoding, but it results in increased computational complexity and delay.

Lack of error resilience results in error propagation, and hence widely varying video quality as a function of time. Current approaches to mitigate the effects of error propagation include error control mechanisms at the transport level. This typically takes the form of retransmissions or forward error correction (FEC). Retransmission based error control methods often fail to be real-time, particularly when round-trip propagation delay is large. FEC schemes on the other hand, are often ineffective when losses are bursty [22]. In terms of error control, existing real-time video applications either only employ intra-block coding [85] which yields low compression efficiency or prohibit the use of lost regions as reference for motion prediction [138] which does not extend to pre-encoded videos.

To address the above problems of rate and error control, one possible solution is to re-engineer the network to provide the necessary quality of service (QoS) guarantees via reservations and admission control [155]. Besides requiring changes in the current infrastructure, this approach also introduces call blocking when resources become temporarily over-subscribed. Furthermore, even when such QoS guarantees are widely available, it is likely to be more costly than the plain old best-effort service.

An attractive alternative is to use bandwidth scalable video with feedback rate control whereby transmission sources adjust their rates in response to changing network conditions. Bandwidth adaptation can then be easily and instantaneously achieved for pre-encoded sources by simple filtering operations. This approach solves the rate control problem while the error control problem can be solved by using an error-resilient compression scheme such as SVIP.

4.2 TCP-Friendly Rate Control (TFRC)

We next consider the problem of determining the available bandwidth, or appropriate communication rate for unicast video communication. One important criterion in determining the available bandwidth is the notion of fairness, which is well-studied in networking and can assume several definitions [83, 65]. Several research groups have focused on developing packet switching disciplines that achieve fairness by requiring routers to store per-flow information [35, 94, 95]. Problems with applying such solutions to the current best-effort Internet include high complexity and the required global infrastructure change. More recently, it has been shown in [66] that a special form of fairness can be achieved without additional router support. By performing end-to-end rate adaptations based on *shadow prices* supplied explicitly by the network, it is proven mathematically that a stable and fair rate allocation can be achieved. Unfortunately, while these results of [66] are interesting, it cannot be applied directly to best-effort packet networks without explicit feedback.

On the other hand, best-effort networks such as IP-networks make no attempt to establish fairness. Because most Internet traffic uses TCP, rate control for the Internet is essentially determined by the behavior of TCP. An outline of TCP rate control is as follows. When there is network congestion and packets are lost, the TCP receiver will notify the sender of the packet loss, and the sender will subsequently slow down its sending rate. In the absence of congestion, the sender slowly increases its sending rate until congestion hits again [61]. This cycle of rate increase and decrease is used to detect the dynamic available bandwidth. While TCP rate adaptation does not explicitly enforce fairness, and is known to be unfair under certain conditions [42, 55], it appears to work well in practice. Thus, rather than attempting to achieve global fairness goals, this Chapter focuses on constructing a TCP-friendly transport protocol that mimics TCP rate control behavior.

One advantage of a TCP-friendly protocol is that for every source/destination pair, fair sharing of resources with multiple instances of itself and other TCP flows, the dominant source of Internet traffic, can be established. TCP cannot be used directly for video transport because reliability is achieved at the expense of time varying delay and throughput. Fig. 4.1 shows the end-to-end delay in seconds when 300 seconds of video material generated at constant rates of 600 and 630 *kbps* respectively are transmitted back to back from Toronto to Berkeley using TCP at noon time. In both cases, since the delay experienced at the end of the experiment is close to zero, the long term average throughput of the TCP connections must exceed the data rates. However, we observe that the instantaneous end-to-end delay can still be significant, up to 6 seconds in our experiments.

There are two sources of latency in TCP: (1) backlog of data when throughput temporarily drops below the data generation rate and (2) retransmission latency. To achieve the low latency objective, we eliminate backlog of data by (a) filtering a scalable video bit-stream to meet the instantaneous throughput and (b) not performing retransmissions. One way to ensure fair competition with TCP for bandwidth is to match exactly the traffic pattern of the TCP rate control algorithm (i.e., an adaptive window algorithm [61]. However the sending window size of TCP is usually halved on determination of a single packet loss, resulting in traffic patterns with abrupt changes. Instead of matching the TCP traffic pattern exactly and instantaneously,

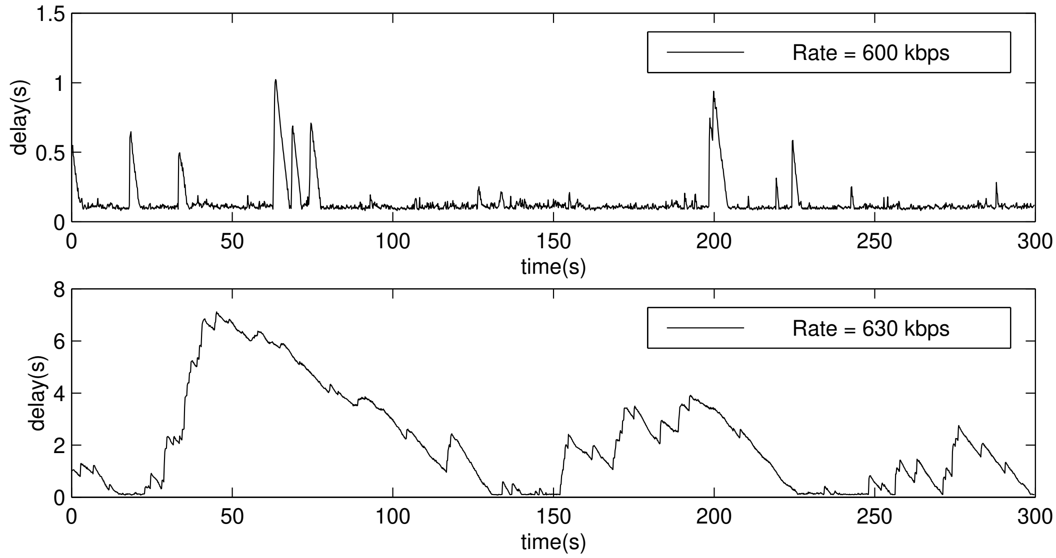


Figure 4.1: Delay between data generation and data reception for video transport using TCP from Toronto to Berkeley (noon, May 11, 1998).

a more relaxed form of fairness can be obtained by matching the TCP throughput on a macroscopic scale. On a lightly loaded network, the TCP sending window has the same size as the amount of buffer space (B) available in the receiver, yielding an average throughput (T) of:

$$T = \frac{B}{RTT} \tag{4.1}$$

where RTT is the dynamic estimate of the round-trip time. When the window size is changing due to the congestion avoidance algorithm [61, 121], Mahdavi and Floyd [79], Mathis *et.al.* [81] have derived and verified expressions relating TCP throughput to the packet loss rate (p):

$$T = k \, \frac{MSS}{RTT \times \sqrt{p}} \tag{4.2}$$

where MSS, the maximum segment size, is the amount that the TCP sending window increases per RTT when there is no packet loss. k is a constant between 0.7 [81] and 1.3 [79] depending on the particular derivation of Equation 4.2.

A non-TCP flow sharing the same path as a TCP flow will experience and measure similar network conditions. By choosing B and MSS, and measuring p and RTT appropriately, it is possible for the non-TCP flow to estimate the throughput of the TCP flow and subsequently control its transmission rate accordingly. Even though B and MSS are implementation dependent, they are constants and do not play an important role in the dynamic evolution of throughput. The dynamic estimates RTT and p however, must be calculated in accordance with actual TCP implementation.

In TCP, a new estimate of RTT is obtained per RTT using a coarse clock that typically has a resolution as low as 500 ms [121]. These coarse estimates are then used to update the average round trip time ($srtt$) as well as the mean deviation using first order autoregressive estimators [61]. These quantities are in turn used to determine the timeout in which an unacknowledged packet is declared lost [121]. To achieve close correspondence with TCP, the same estimator is kept to determine timeout. Packet losses within one RTT are considered a single congestion event, and p is determined by [81]:

$$p = c/D \tag{4.3}$$

where c is the number of congestion events in an observation window while D is the amount of data in units of MSS transmitted in the same observation window. Since the coarse granularity of the TCP clock inadvertently causes much fluctuation in the measurement of RTT, the throughput estimated by Equations 4.1 and 4.2 will show large variation. Instead, the throughput is calculated using an additional RTT estimate that is measured using an accurate clock.

Given estimates of TCP throughput, we construct TCP-friendly rate-based transport protocols (TFRP) by first choosing a desired transport packet size that is reasonable for the network being used. This involves a trade-off between large protocol overhead for small packets and higher chance of fragmentation for large packets. Specific numbers will be given when we describe our network experiments in Section 4.3. A video transmitter attempts to make a fixed number of transmissions per second, which is determined by the characteristics of the video source. For the SVIP sources we used for experimentation in Section 4.3, the number is chosen so that over a period of one GOP, the number of transmissions equals the number of components used in the SVIP source. The size of each transmission is modulated by the rate controller so as not to exceed throughputs given by Equations 4.1 and 4.2. We reduce transmission overhead by combining consecutive small transmission attempts if the combined size does not exceed the desired transport packet size. Larger packets are transmitted as is and without fragmentation.

4.3 Performance of SVIP with TFRC

This section, describes experiments using SVIP in conjunction with TFRP over the best-effort Internet. We will first show that uncontrolled flows may lead to severe congestion so that rate control algorithms should always be used. We then demonstrate the fairness of TFRP when competing with TCP for bandwidth. The results of simultaneous transmissions of SVIP and its non-resilient counterpart (T)[1], both using TFRP will then be presented. Finally, we will compare SVIP against scheme (T) with adaptive FEC employed.

Because video traffic consumes considerable network resources, non-adaptive flows can cause severe congestion. Fig. 4.2 shows the results of a non-adaptive flow from Hong Kong to Berkeley carrying video compressed using Scheme (T) at 1 Mbps. The video source is a 600 frames *Raiders* sequence encoded at 12 fps and is being transmitted in a loop. Every 4 frames is packetized into linearly dependent packets to provide low latency and limit the effects of error propagation. We see that the large transmission rate induces much congestion with a packet loss rate average of about 33%. A very large average Mean Square Error (MSE) of 777 is sustained despite the high data rate.

Since scheme (T) is bandwidth-scalable, we can apply TFRP to avoid transmission at excessive rates. The result is shown in Fig. 4.3. Comparing the results in Fig. 4.3 with those of Fig. 4.2, we see that much of the congestion is eliminated even though occasional peaks are still observed due to the linear dependency of packets under scheme (T). The average MSE for the experiment is 67.

To achieve rate control in a TCP friendly manner, we propose the use of TFRP.

[1]Described in Section 3.5.2

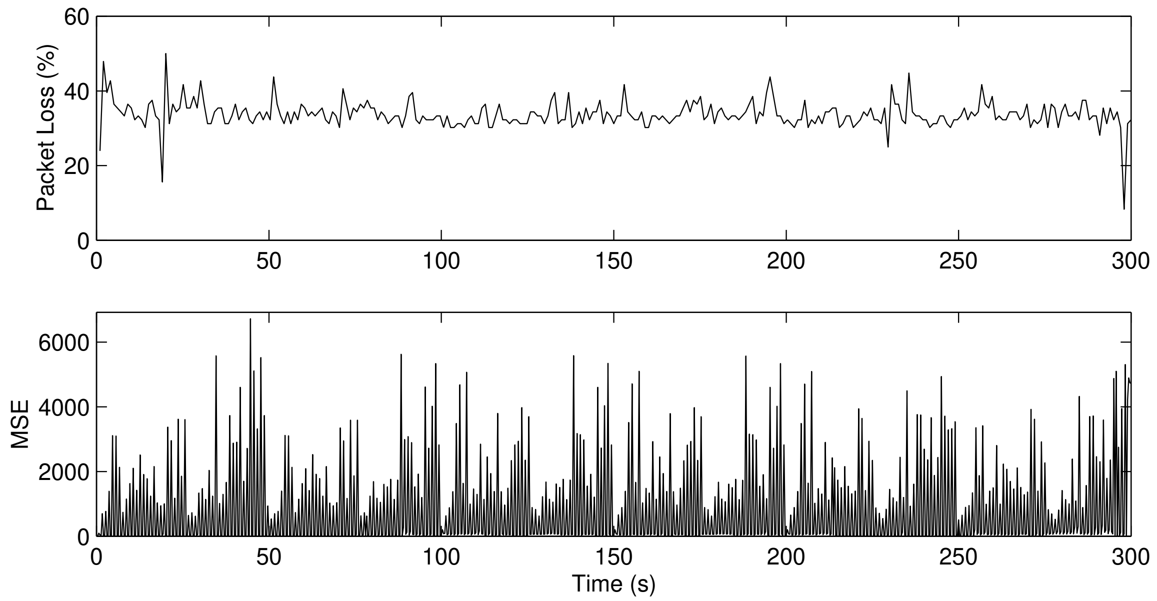


Figure 4.2: Transmission of non-adaptive flow at 1 Mbps from Hong Kong to Berkeley using scheme (T) (2 pm, 9/8/98).

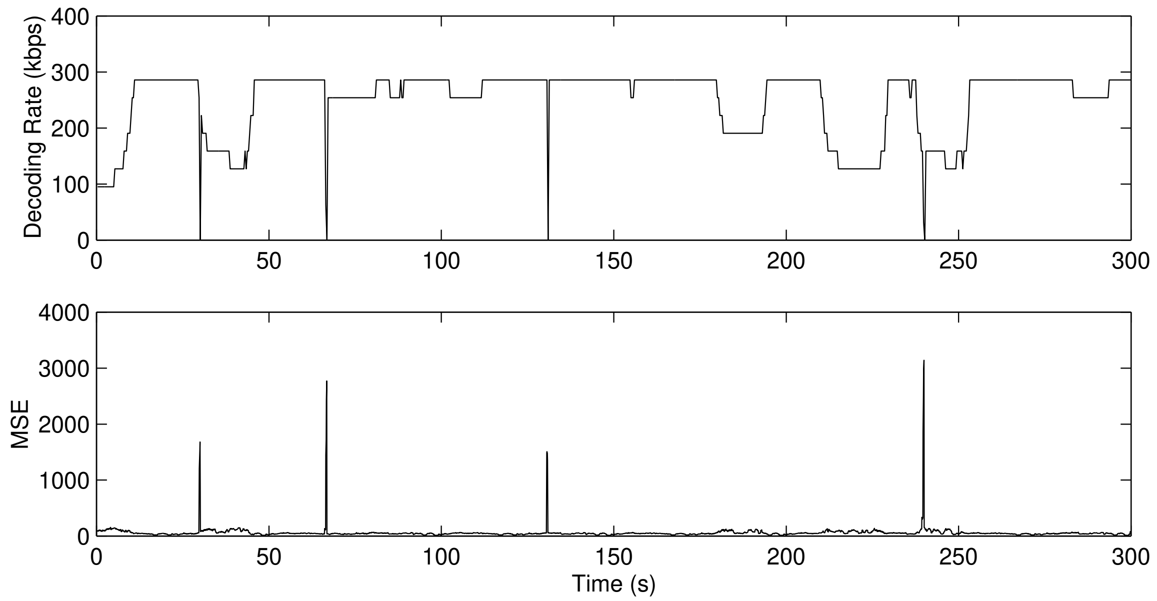


Figure 4.3: Transmission of adaptive flow from Hong Kong to Berkeley using scheme (T) and TFRP (3 pm, 9/8/98).

Figs. 4.4 and 4.5 show simultaneous transmission of TCP and TFRP from Hong Kong to Berkeley and Toronto to Berkeley respectively. A transport packet size of 500 bytes is chosen, which parallels the transport packet size of some TCP implementations. Seventy packet transmissions are attempted per GOP time with an observation window of 32 $\times$ RTT to measure p. The selection of measurement window is discussed in more details in the Appendix of this Chapter. The size of each packet is modulated by the rate controller so as not to exceed throughputs given by Equations 4.1

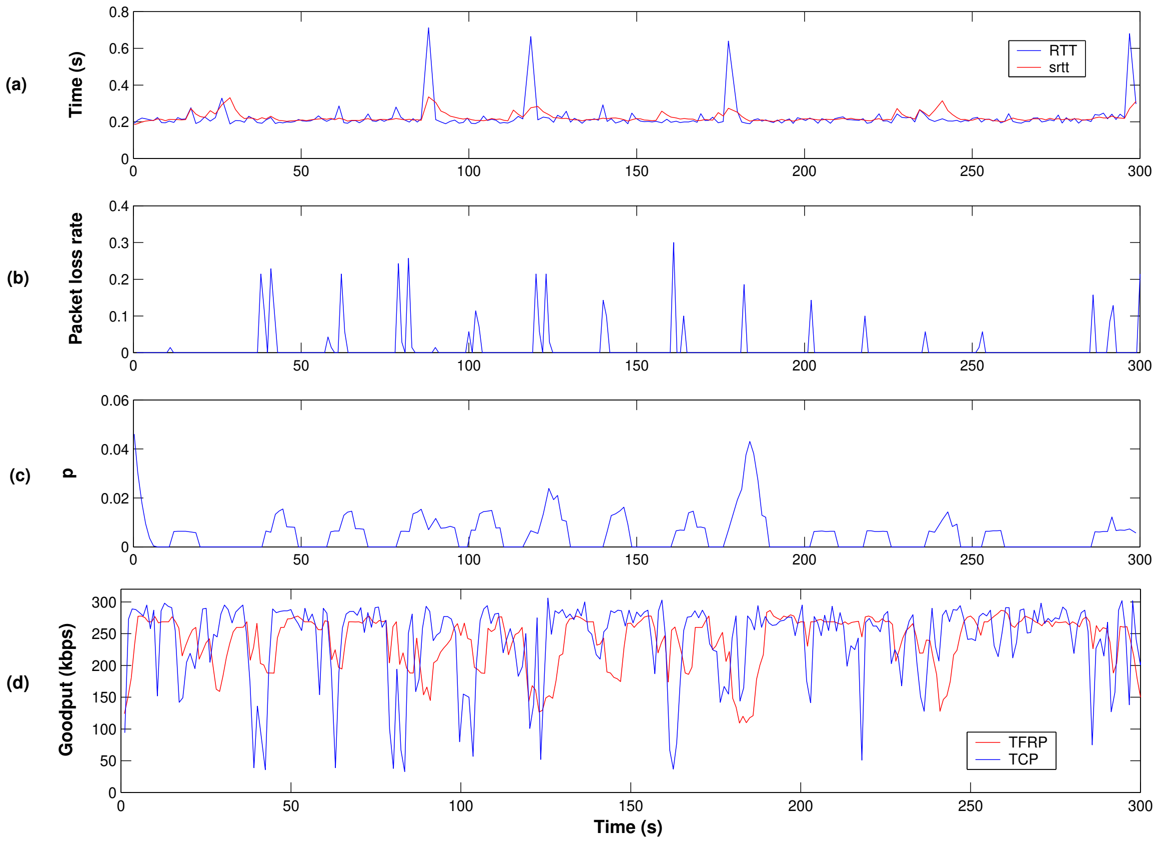


Figure 4.4: Simultaneous transmission using TCP and TFRP between Hong Kong and Berkeley. (a) shows instantaneous RTT and the average RTT ($srtt$). (b) shows packet loss rate observed at the receiver. (c) shows p as estimated at the sender. (d) shows the bandwidth consumes by TCP and TFRP.

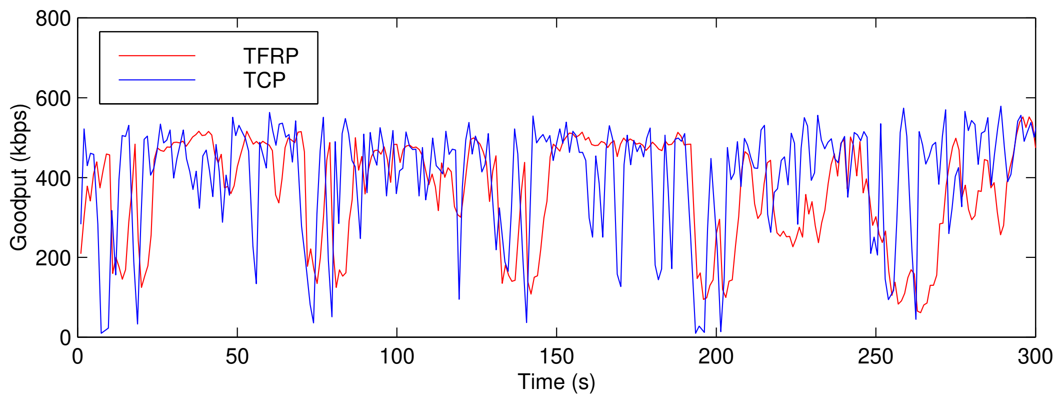


Figure 4.5: Throughput of simultaneous transmission of TCP and TFRP from Toronto to Berkeley.

and 4.2. Successive small packets may be combined to reduce transmission overhead if the combined size does not exceed 500 bytes. An initial estimate of 5% is used for p which is eventually replaced by the estimate of Equation 4.3 after $32 \times RTT$. Both channels are congested, showing an average loss rates of 1.4% and 0.5% respectively, with occasional bursty losses as high as 30%. The average throughputs for TCP is 241 and 414 *kbps* from Berkeley to Hong Kong and Toronto respectively while that of TFRP is 239 and 370 *kbps* respectively. This demonstrates that TFRP can co-exist with TCP. We see that while the protocols show similar throughput in both cases, TFRP shows significantly less variation than TCP due to the use of an averaging window to measure p. This result is particularly useful for real-time multimedia transport where large and frequent quality fluctuations are undesirable. The use of a larger window provides further smoothing, but at the expense of longer reaction time to changes in network conditions. We see that TFRP shows less throughput variation for the flow between Hong Kong and Berkeley than the flow between Toronto and Berkeley. This is due to the longer congestion times for the Toronto to Berkeley connection which TFRP considers as a signal for persistent congestion and therefore reduce its transmission rate.

Fig. 4.6 shows the bit-rate trace of TFRP from Toronto to Berkeley. Initially the throughput stabilizes at around 700 *kbps* until two more TFRP instances are started 80 seconds into the experiment. This causes temporary congestion before each instance of the protocol adjusts its transmission rate to around 480 *kbps*. There is little actual packet loss except when the protocol is reacting to the change in network conditions. This experiment shows that TFRP is successful in sharing the available bandwidth to avoid excessive loss.

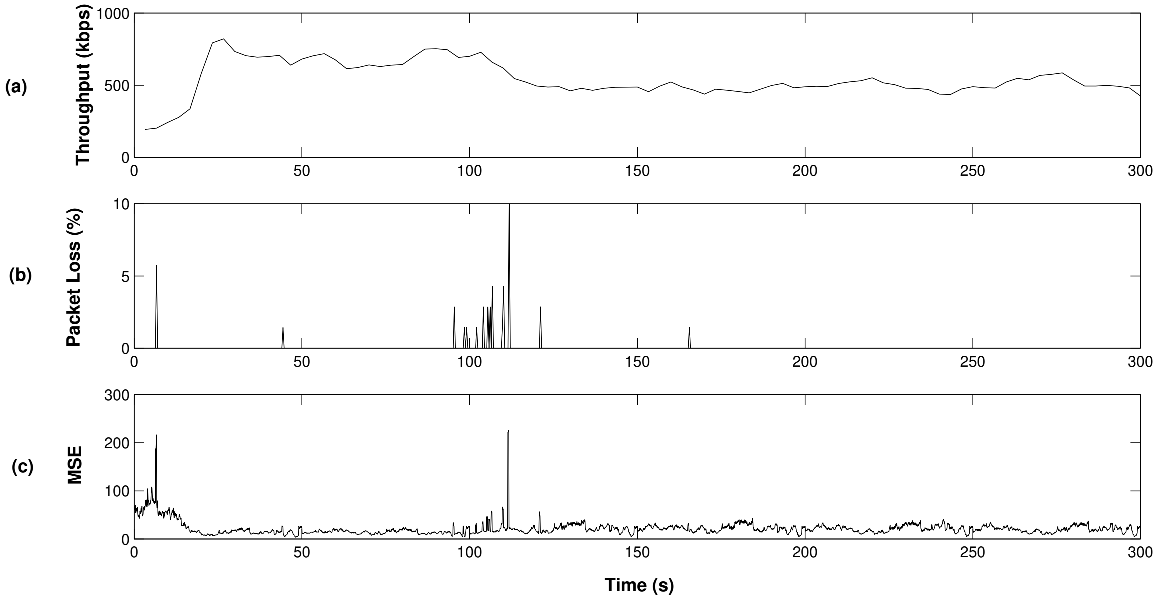


Figure 4.6: Results of transmitting SVIP using TFRP from Toronto to Berkeley (6 pm, June 23, 1998). (a) shows throughput, (b) shows packet loss rate at receiver, and (c) shows the MSE trace of the received video.

4.3.1 Streaming without the use of FEC

We next compare the simultaneous Internet transmission of scalable video compressed using SVIP schemes (T). TFRP is used to carry both streams. The packet loss rate is measured in 1/3 second intervals and is shown in the top graph of Fig. 4.7. There are moderate and bursty packet losses, with an average around 3.8% and variation between 0 to 23%. We see that SVIP out-performs (T) significantly both in having a lower average MSE of 105 versus 244, and in having smaller standard deviation of 69.9 versus 460. Visually, packet losses under SVIP appear as blurred patches in different locations in different frames whereas in (T) we observe uniformly blurred frames. However, the data dependency between packets in (T) causes large temporal variation in quality, resulting in oscillations between high quality and very poor quality pictures. Such wild oscillations do not exist for SVIP since the packets do not depend on each other. As a result, SVIP produces much more visually pleasing decoded video than scheme (T).

4.3.2 Streaming with the use of FEC

We next compare the simultaneous Internet transmission of SVIP and scheme (T) video using adaptive FEC. We restrict our discussion to the comparison between SVIP and scheme (T) because both are finely scalable and can be easily used in conjunction with TFRP and are similar in that both employ hierarchical block coding in addition to 3D subband analysis.

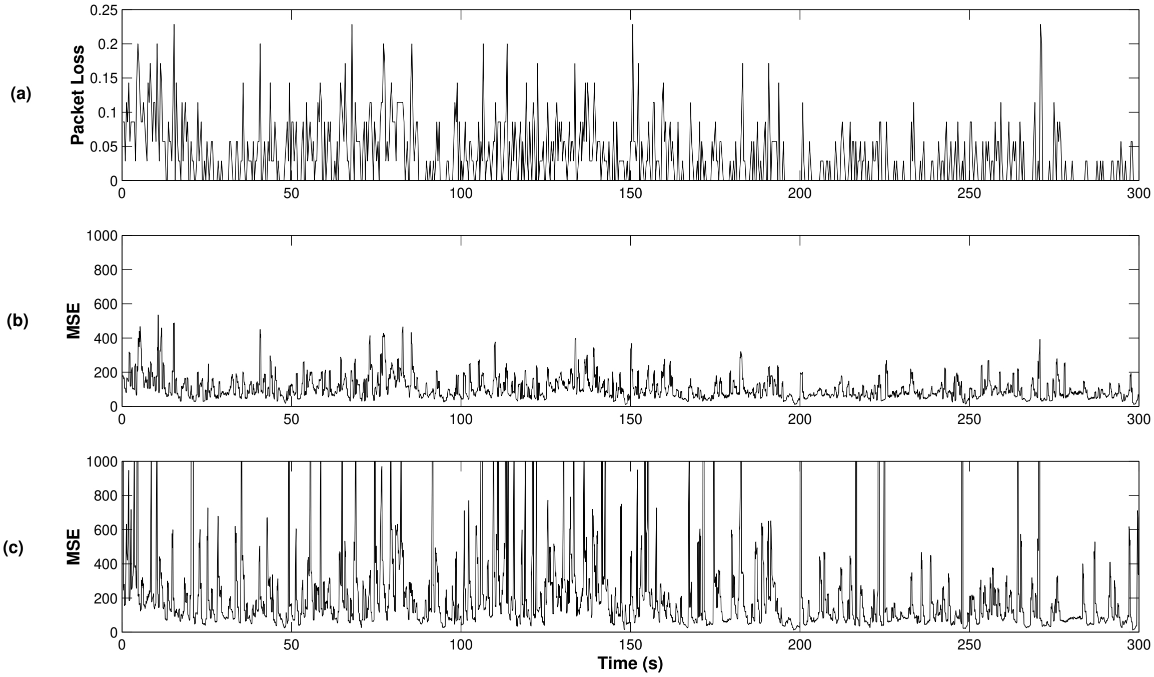


Figure 4.7: Results of video transmission from Hong Kong to Berkeley using TFRP (3 pm on 10/20/98). (a) shows loss rate, (b) and (c) shows the MSE trace for SVIP and scheme (T) respectively.

Fig. 4.8 shows the distortion-rate characteristics when the *Raider* sequence under SVIP and scheme (T) are subjected to different simulated losses and with no FEC added. We see that the two schemes have comparable performance when there is no loss. However, the performance of scheme (T) deteriorates much faster than that of SVIP as loss rate increases. If we were to know the packet loss rates *a priori*, one possible way to improve the quality of scheme (T) transmission is to apply unequal error protection (UEP) whereby more important packets are protected more. Furthermore, since scheme (T) is bandwidth scalable, the UEP scheme can be performed without overall data expansion by reducing the number of informational packets to make room for redundancy or parity packet.

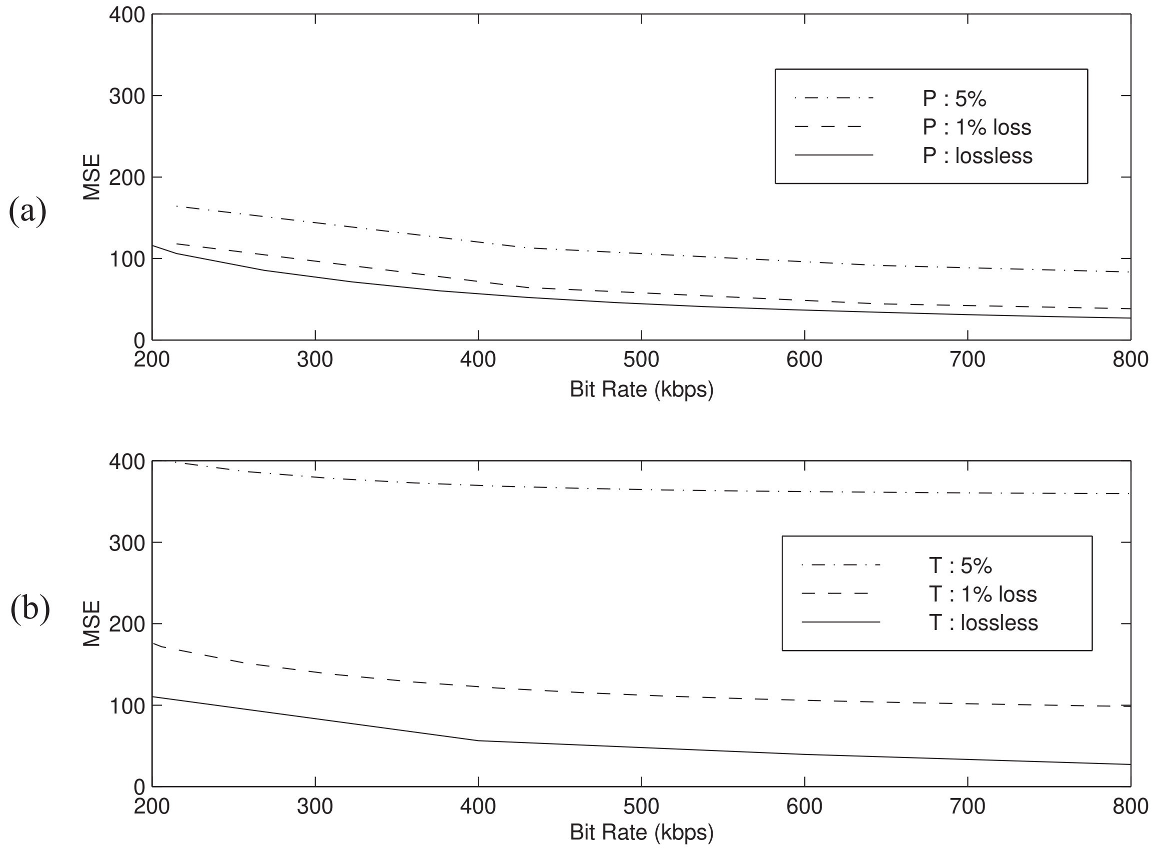


Figure 4.8: Distortion-rate characteristics of (a) SVIP and (b) scheme (T) at 0, 1 and 5% random packet loss.

One common approach to achieving unequal error protection for wireless environments is to use Rate-Compatible Punctured Convolutional (RCPC) codes [51] under which it is possible to assign different levels of protection to different blocks of bits. However, in order to use RCPC to adaptively generate FEC, the complex task of computing the optimal code on a real-time basis under time-varying bandwidth constraint and packet loss rate has to be addressed. Furthermore, the choice of the code has to be reliably communicated to the receiver before decoding can proceed.

Such schemes however, are relatively computation intensive. Instead, we will achieve unequal error protection for scheme (T) using packet replication [144].

Given a bandwidth constraint C, and a known packet loss rate p_{loss}, our proposed UEP scheme minimizes the expected distortion of scheme (T) video, D, given by:

$$D = \sum_{i=1}^{M} p_i D_i \tag{4.4}$$

where p_i is the probability that all packets up to packet i are received and packet $i+1$ is lost. D_i is the distortion when the first i packets are decoded and is approximated by the zero loss curve in Fig. 4.8. M is the least important packet that is being transmitted and is given by the bandwidth constraint:

$$\sum_{i=1}^{M} n_i R_i \leq C \tag{4.5}$$

where n_i is the number of copies that packet i is transmitted and R_i is the size of packet i. To reduce the number of variables, and hence computational complexity, we restrict all packets to have equal size; i.e. $R_i = R \ \ \forall i$. This is a reasonable assumption because unequal packet size introduces considerable difficulties in actual implementation. A further assumption is that packet losses are independent. Under that assumption, p_i can be calculated from probability of packet loss (p_{loss}) according to the following formula:

$$ps_0 = 1.0$$

$$p_i = ps_{i-1} \times p_{loss}^{n_i}$$

$$ps_{i+1} = ps_i - p_i$$

where ps_i denotes the probability of receiving the first i layers correctly.

Given the above assumptions, the only remaining variables for the constrained optimization problem are M and n_i. The optimization is performed using exhaustive search over all possible M and n_i that satisfy Equations 4.4 and 4.5. The size of the search space however, is prohibitive; specifically, it is equal to the number of ways to distribute $N = \lfloor \frac{C}{R} \rfloor$ balls into M bins where M can range from 1 to N. We can reduce the size of the search space by enforcing the constraint that more important packets have at least the same level of protection as less important packets. This is an intuitively pleasing assumption for scheme (T) since loss of packet i would render packets $i+1$, $i+2$, ... useless. Mathematically, this translates into $n_i \geq n_j$ for $i < j$.

For our experiments, the packet size R is chosen to be 300 bytes to yield fine grained bandwidth-scalability. Decreasing R further will provide finer granularity at the expense of larger transmission overhead. The packet loss rate p_{loss} is approximated using a running estimate that is updated at least 3 times per second. The n_i are pre-computed using exhaustive search for 16 fixed loss rates ranging from 0.25% to 20%. Table 4.1 shows the size of the search space for various values of N, the total number of packets to be sent as dictated by the flow control algorithm of TFRP, as well as the optimal values for n_i when $p_{loss} = 0.05$. When p_{loss} is increased to 0.1, the optimal values of n_i change according to Table 4.2. As expected, at larger loss probabilities, the optimization algorithm opts to send fewer packets, but protects those few packets with more duplicates than at lower loss probabilities.

Similarly for SVIP, given a fixed rate at which to code a component, it is possible to reduce that rate to make room for piggy-backing low rate versions of other components as FEC. Following a formulation similar to that of Equations. 4.4 and 4.5, an optimal FEC scheme can be obtained for every assumed packet loss rate.

We carried out an actual Internet experiment again between Hong Kong and Berkeley. The results are shown in Fig. 4.9. The average MSE for SVIP and scheme (T) with FEC are 100.53 and 201 respectively. Both show an improvement over the case when FEC is not used. A comparison of Figs. 4.7 and 4.9 indicates that FEC reduces the variation in the decoded video for scheme (T). However, scheme (T) with FEC still shows larger average distortion and larger variation compared to SVIP with or without FEC. The better performance of the error-resilient scheme compared to that of the non-resilient scheme with FEC is due to the mismatch between the assumed and actual packet loss rates. Since only delayed channel estimates are available for

	search space	n_1	n_2	n_3	n_4	n_5	n_6	n_7
N = 5	7	2	1	1	1			
N = 10	42	3	2	1	1	1	1	1

Table 4.1: Optimal distribution of duplicate packets for 5% packet loss and total transmitted packet of N.

	n_1	n_2	n_3	n_4	n_5	n_6	n_7
N = 5	3	1	1				
N = 10	4	2	2	1	1		

Table 4.2: Optimal distribution of duplicate packets for 10% packet loss and total transmitted packet of N.

the best-effort Internet, such a mismatch is unavoidable. However, FEC schemes are typically optimized only for a fixed assumed loss rate and are sensitive to such mismatches. SVIP, on the other hand, is designed without assuming a packet loss rate and therefore tends to perform better when the channel states are not precisely known.

4.4 Conclusions

In this Chapter, we described a low latency video transmission scheme consisting of a TCP-friendly transport protocol with our proposed bandwidth-scalable compression scheme SVIP. Low latency is established by the elimination of buffering in

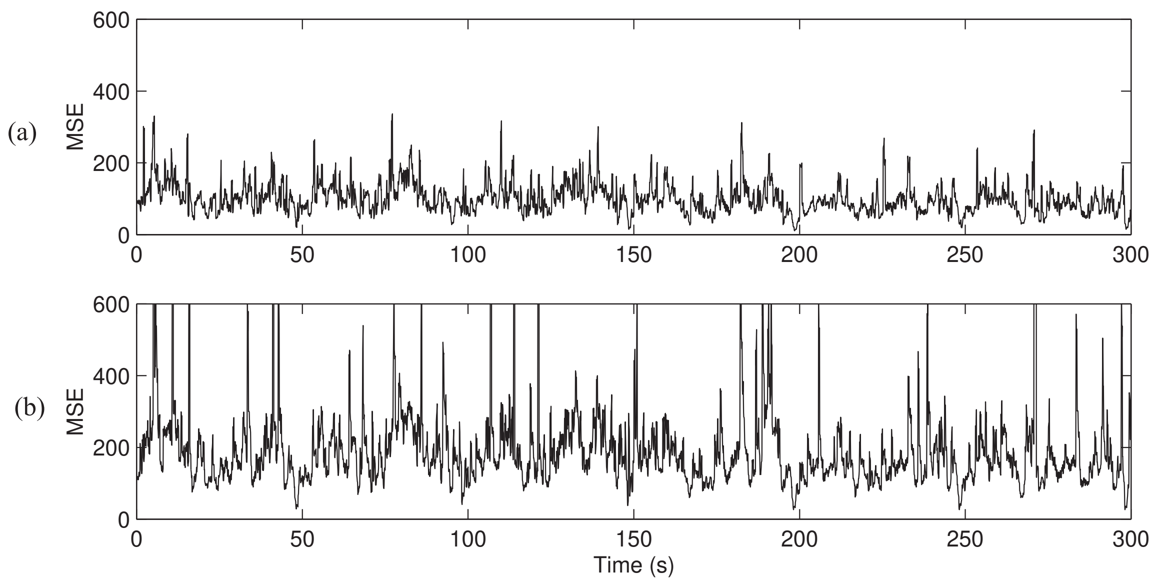


Figure 4.9: Distortion of a simultaneous video transmission of (a) SVIP, and (b) scheme (T) from Hong Kong to Berkeley using TFRP and adaptive FEC.

rate control so that the overall latency is propagation delay plus decoding time. We We perform actual Internet experiments comparing another bandwidth-scalable compression scheme (T) which is similar to the proposed compression scheme except with dependencies among packets. We find that the ability to independently decode packets reduces the resulting distortion significantly even after an adaptive FEC scheme is applied to protect the non-resilient bit-stream.

4.A Appendix – TFRC Parameter Measurement

One central issue in the design of TCP-friendly rate control (TFRC) is the measurement method of the parameter p in Equation 4.2. In this Appendix, we will compare two methods of calculating p, which represents the number of congestion event[2] per unit time. In the *window-based* approach, p is calculated as n_W/W, where W is a time window corresponding to a fixed multiple of round-trip time, and n_W is the variable number of congestion events in W. In the *event-based* approach, p is calculated as K/W_K, where K is a fixed parameter, and W_K is the variable time covering the last K congestion events. To motivate the use of the window-based approach in Chapter 4, we will first show via simulations that the event-based approach may result in excessively long reaction time when the network transitions from a congested state to a less congested state. We will then investigate the effects of changing the window size W on the estimated throughput.

[2]Packet losses within one round-trip time are considered as a single congestion event.

4.A.1 Event-based and Window-based Loss Measurement

Theoretically, when the occurrence of loss event is frequent, a small observation window is sufficient to measure p to a given accuracy. Conversely, when loss event occurs infrequently, a large window is needed to achieve the same accuracy. Under event-based measurement, the observation window is a constant multiple of the period of congestion event, and therefore is small when loss event occurs frequently, and large when loss event occurs infrequently. Thus, an event-based measurement method for p would appear to be more suitable than the window-based method. However, the use of event-based measurement suffers from another drawback in practice. Consider the use of event-based measurement with $K = 8$ [43], and with congestion events happening at the times marked with crosses in Fig. 4.A. At time t_1, the measurement window is given by W_8, and p is estimated by $8/W_8$. Similarly, at time t_2, the measurement window is given by W_8', and p is estimated by $8/W_8'$. However, between t_2 to t_3, there is no new congestion event, and at t_3, the measurement window is still given by W_8'. The lack of a congestion event between times t_2 and t_3 will therefore prevent a sender at time t_3 to realize that the network conditions have changed unless some additional time-out mechanisms are incorporated [43]. Despite the network transitions to a less congested state at time t_2, it is not until time t_4 that the transition is recognized.

We next present *ns* simulations to compare the reaction times to sudden increase and decrease of network load using the window-based scheme described in Section 4.2, and an event-based scheme with $K = 8$. The choice of $K = 8$ has been shown in [43] to

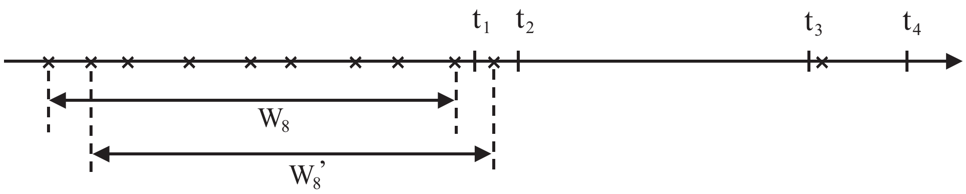


Figure 4.A: Timing diagram illustrating the long reaction time of an event-based measurement scheme. The crosses indicate congestion events.

be reasonable for many circumstances. The simulation topology is shown in Fig. 4.B where a TFRC flow is sharing a bottleneck link of 24 $Mbps$ with 49 TCP flows. In

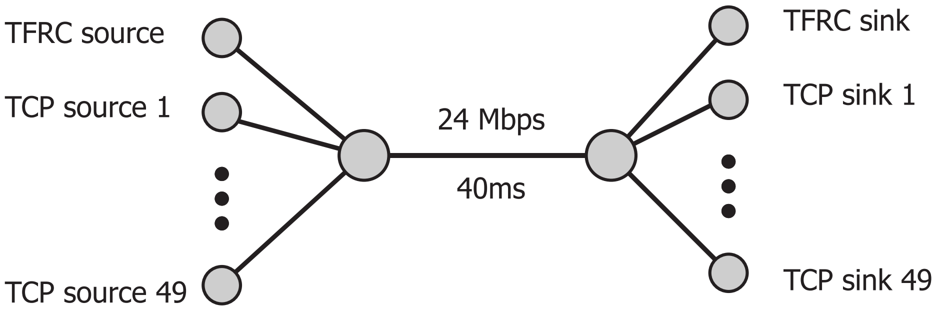


Figure 4.B: One TFRC flow sharing bandwidth with 49 other TCP flows in a network with the dumbbell topology.

two independent runs of the experiment, a different implementation of TFRC using event-based and window-based measurement will be used respectively. All sources are started at $t = 0$. At $t = 100s$, 36 of the 49 TCP connections are stopped to give a sudden decrease in network load. At $t = 300s$, the 36 stopped TCP connections are resumed to give a sudden increase in network load. The response to the decrease and increase of network load at times 100 and 300 seconds are shown in Figs. 4.C-(a) and (b) for the event-based and window-based measurement schemes respectively. Our main observation is that under the event-based scheme, it takes about 30 seconds for the transition at $t = 100s$ to a less congested state to complete. The same transition takes about 4 seconds under the window-based scheme. The transition time at $t = 300s$ is shorter under the event-based scheme than the window-based scheme. However, the transition time under the window-based scheme is reasonably small at 4 seconds. The potentially large reaction time of the event-based scheme is the primary reason for using window-based measurement in Section 4.2.

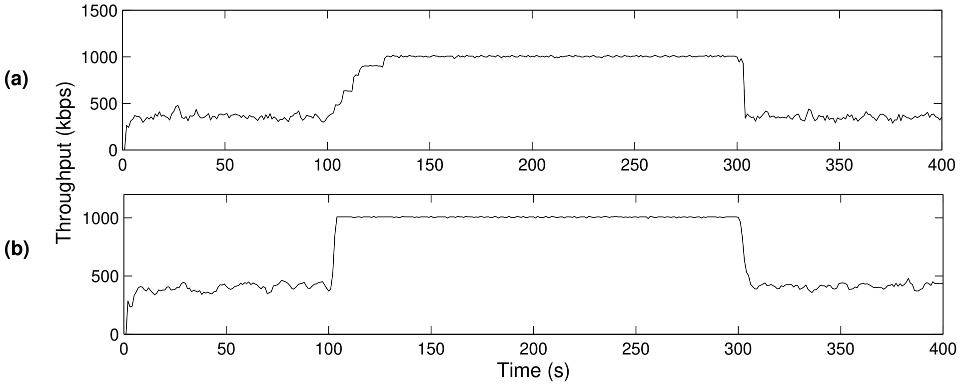


Figure 4.C: Response of TFRC to sudden decrease of network load at $t = 100s$ and sudden increase at $t = 300s$ under (a) event-based measurement and (b) window-based measurement.

4.A.2 Measurement Window

In this Section, we will examine the effects of changing the window size under the window-based measurement scheme. Specifically, we will consider (a) the smoothness of the throughput trace, and (b) the response time to changing network conditions. The same simulation setup shown in Fig. 4.B is used, with the same transitions to low-congestion and high-congestion at $t = 100s$ and $t = 300s$ respectively. In six independent runs of the experiment, a different implementation of TFRC will be used with window size 8, 16, 32, 64, 128 and 256 times the round-trip time. The results are shown in Fig. 4.D. As expected, the use of a larger window size results

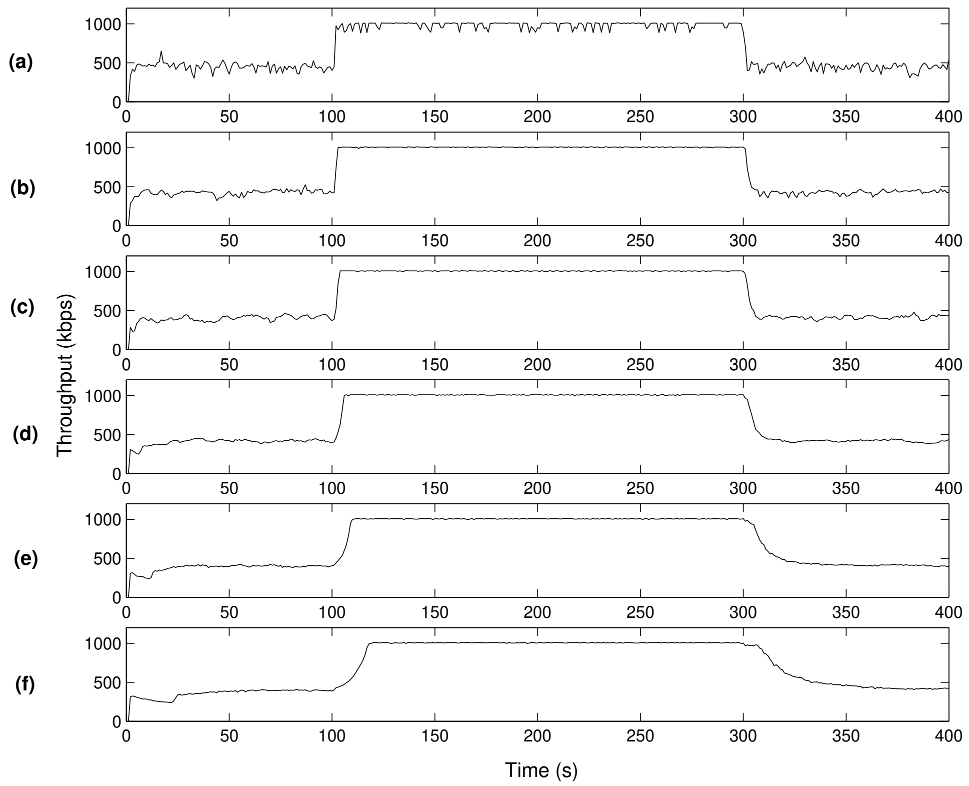


Figure 4.D: Throughput trace of TFRC using different window sizes of (a) 8, (b) 16, (c) 32, (d) 64, (e) 128 and (d) 256 times round-trip time.

in smoother throughput when network condition does not change. However, larger window size also results in longer reaction time to changes in network condition. For example, when the measurement window is 128 and 256 times round-trip time, the transition at $t = 100s$ takes 10 and 20 seconds to finish respectively. Using a smaller window of 32 times round-trip time will reduce the transition time to about 4 seconds. Generally, the selection of the window size involves a trade-off between smoothness and responsiveness. In our TFRC implementation in Section 4.2, a window of 32 times round-trip time is chosen for reasonable responsiveness and smoothness.

Chapter 5

Multicast Streaming using SVIP and layered FEC

The use of scalable video with layered multicast has been shown to be an effective method to achieve rate control in heterogeneous networks. In this chapter, we describe the use of layered FEC as an error control mechanism in a layered multicast framework. By organizing FEC into multiple layers, receivers can obtain different levels of protection commensurate with their respective channel conditions. Efficient network utilization is achieved as FEC streams are multicast only to receivers that need them. Furthermore, FEC is used without overall rate expansion by selectively dropping data layers to make room for FEC layers. Effects of bursty losses are amortized by staggering the FEC streams in time, giving rise to a trade-off between delay and quality. For rate control at the receivers, we propose an equation-based approach that computes network usage as a function of measured network characteristics. We show that equation-based rate control achieves fairer bandwidth sharing amongst competing sessions as compared to existing multicast rate control schemes such as Receiver-driven Layered Multicast (RLM) and Receiver-driven Layered Congestion control (RLC). Fairness is achieved because competing sessions sharing a path will measure similar network characteristics. Simulations and actual MBONE experiments are performed using the error-resilient, scalable video compression scheme SVIP. We find that video quality is significantly improved at the same communication rate when layered FEC is used.

5.1 Introduction

The deployment of IP multicast in the Internet Multicast Backbone (MBONE) has provided a convenient and efficient way of disseminating video information to multiple recipients over the Internet. Similar to the unicast Internet, the MBONE is a best-effort network that requires rate and error control mechanisms for video communications. However, the implementation of such rate and error control functions are considerably more challenging for multicast than unicast. This is because in a unicast setting, the needs of only a single receiver have to be addressed whereas in a multicast setting, it becomes necessary to simultaneously cater to the potentially conflicting needs of multiple recipients.

In best-effort unicast networks, rate control is typically implemented by adapting the communication rate at the source in reaction to feedback information from the receiver [61]. This source-based adaptation approach is widely used for unicast data communications, and is the basis of many unicast [63, 138, 129] and some multicast [12, 3] video transmission schemes. However, by sending the same single stream to all receivers, multicast source-based adaptation schemes cannot accommodate the heterogeneity of channel conditions and user preferences in multicast. Furthermore, by requiring feedback information from all receivers, source-based adaptation schemes also suffer from the feedback implosion problem, which occurs when the source is overwhelmed by feedback messages from the receivers in a large multicast.

One possible approach to overcome the short-comings of the source-based adaptation scheme is to transmit replicated copies of the same video corresponding to different communication rates using different multicast groups [19]. In such a way, the bandwidth heterogeneity problem is better addressed since each receiver can choose, from the available multicast groups, the single copy whose rate best matches its experienced channel conditions. However, since transmitting multiple copies of the same source consumes more network bandwidth, the actual number of copies that can be effectively sent is usually limited. Further improvements of the replicated stream approach includes adaptively modifying the rates of each stream [19] at the cost of additional complexity due to the need for feedback, and thus the possibility of feedback implosion [75].

A more flexible alternative to the multicast rate control problem is offered by the layered multicast framework for scalable video [114, 32, 84]. Scalable video compression produces an embedded bit-stream that allows decoding at multiple rates [134, 85, 68, 129]. Under layered multicast, different layers of a scalable video are carried in different multicast groups so that receivers can individually subscribe or unsubscribe to the appropriate multicast groups to achieve rate control [84, 114, 135, 140, 139, 76]. Thus, the more layers to which a receiver subscribes, the higher the quality of the received video. Generally, producing an embedded bit-stream for scalable compression results in lower compression efficiency than its non-scalable counterpart at the same bit rates [27]. Nevertheless, layered multicast is often preferred to schemes based on source adaptation or stream replication due to its ability to effectively address bandwidth heterogeneity.

On the error control side, a large body of work has focused on providing partial or full reliability to all receivers using retransmissions [44, 97] or type-II hybrid-ARQ [147] schemes employing both error control codes across packets and retransmissions [90, 98, 67, 109]. Since source-based retransmission schemes are well known to be non-scalable to large groups of receivers, elaborate schemes have been employed to organize the receivers into repair trees to effect hierarchical retransmission [97, 67, 150], or local groups to facilitate local repair [44]. However, the effectiveness of such schemes is undermined by the lack of explicit topological information under IP-multicast. As a result, there is no explicit basis to determine how receivers should be organized into repair trees or groups [91]. Consequently, complex and inaccurate methods have been used to infer the network topology [44, 150]. Alternatively, manual building of repair trees [97, 78] and manual configuration of networks [67] have been proposed, with the drawback of requiring *a priori* knowledge of the set of receivers, the network topology, and requiring administrative privileges to all participating networks.

Due to high cost and complexity, multicast retransmissions are often justified only for low bandwidth applications that require reliability, such as the shared whiteboard in the MBONE tools [44] or financial data distribution. Retransmissions are inappropriate for video multicast applications due to the higher communication rates and packet loss tolerance of video. In contrast, pure forward error correction (FEC) based approaches to error control can be easily realized under IP multicast. For instance, the Digital Fountain approach for reliable, non real-time multicast of data is based on transmitting the original and redundant data in a "carousel" manner [15].

This chapter investigates the use of an FEC based error control mechanism for video multicast which we will refer to as *layered FEC*. An overview of the basic layered FEC scheme is presented in Section 5.2. The optimization procedure for each receiver is discussed in Sections 5.3. The use of SVIP described in Chapter 3 in a layered FEC setting is discussed in Section 5.4. In Section 5.5 some possible approaches to organize FEC in a layered manner are discussed. To evaluate the layered FEC scheme in a setting where multiple concurrent sessions are present, a rate control mechanism that address inter-session fairness is needed. To this end, Section 5.6 discusses the use of an equation-based rate controller and evaluates its fairness properties via simulations. Simulation and experimental results for video transmissions are presented in Section 5.7, followed by conclusions in Section 5.8.

5.2 Overview of the Layered FEC Scheme

There are three objectives for the layered FEC scheme. First, it has to be compatible with the layered multicast model to allow effective rate control. Second, to account for the heterogeneity of packet loss rates in multicast, it needs to provide different levels of protection to different receivers. Third, it needs to cater to the different latency preferences of different users. For example, in a live multicast of a lecture, participants who want to ask questions and interact with the lecturer have a stronger preference for low latency rather than high picture quality. On the other hand, passive viewers may be willing to sacrifice latency for higher video quality. We shall classify the viewers according to their tolerance to latency and refer to the former as *low-latency viewers*, and the latter as *high-latency viewers*.

The basic mechanism of the layered FEC scheme is illustrated in Fig.5.1. Video frames are blocked into groups of pictures (GOP) that are compressed together using scalable video compression techniques [129, 134, 85, 68]. In Fig. 5.1, GOP 0 is compressed into 3 video or data layers, GOP-0-1 to GOP-0-3, with more layers corresponding to higher picture quality. In addition, FEC layers are constructed to protect against packet losses, with more FEC layers corresponding to higher level

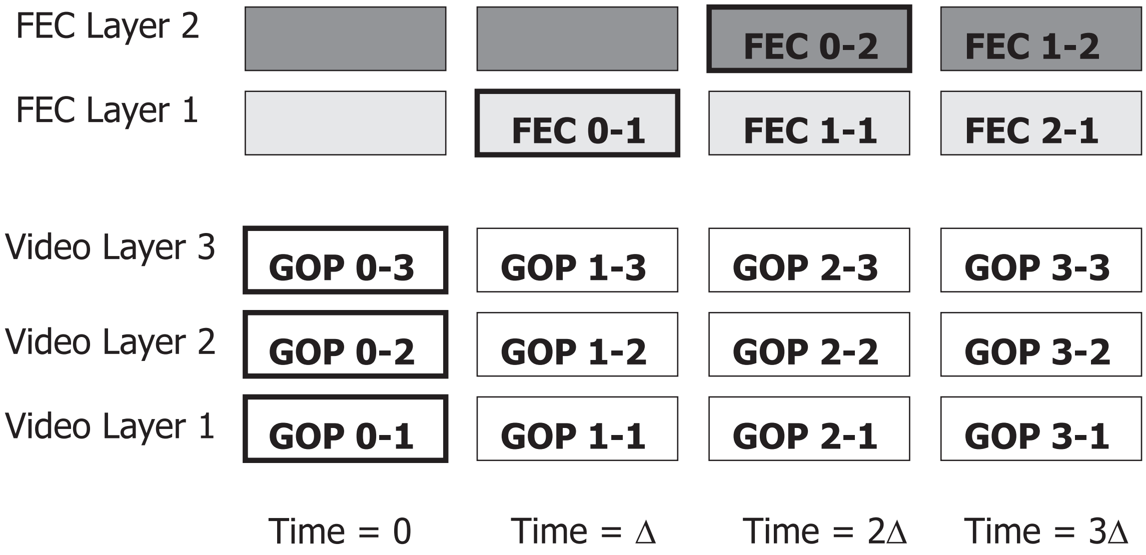


Figure 5.1: Timing diagram of the layered FEC scheme.

of protection. In Fig. 5.1, two FEC layers, FEC-0-1 and FEC-0-2 are created for GOP 0. Each of the data and FEC layers is then transmitted in a different multicast group. In such a way, a *menu* of layers is created, and each receiver can individually choose from it the best set of layers that satisfies a prescribed rate constraint. The data layers corresponding to the same GOP are transmitted in the same time slot to provide low latency while the transmission of FEC layers are delayed with respect to the data layers as a counter measure for burstiness in packet losses. A related scheme has previously been suggested in [11] for packet audio where coarsely quantized copies of audio segments are staggered in time and transmitted using one multicast group to combat loss burstiness. There are two consequences to the delaying of FEC layers. First, low-latency viewers who desire real-time video cannot utilize the FEC layers. Second, subscribing to larger number of FEC layers results in larger latency, thereby constituting a trade-off between the level of protection and latency. Note that in some special cases, as in the scheme of [107] where special dependence structure exists between video frames, it is possible to use FEC for low-latency applications.

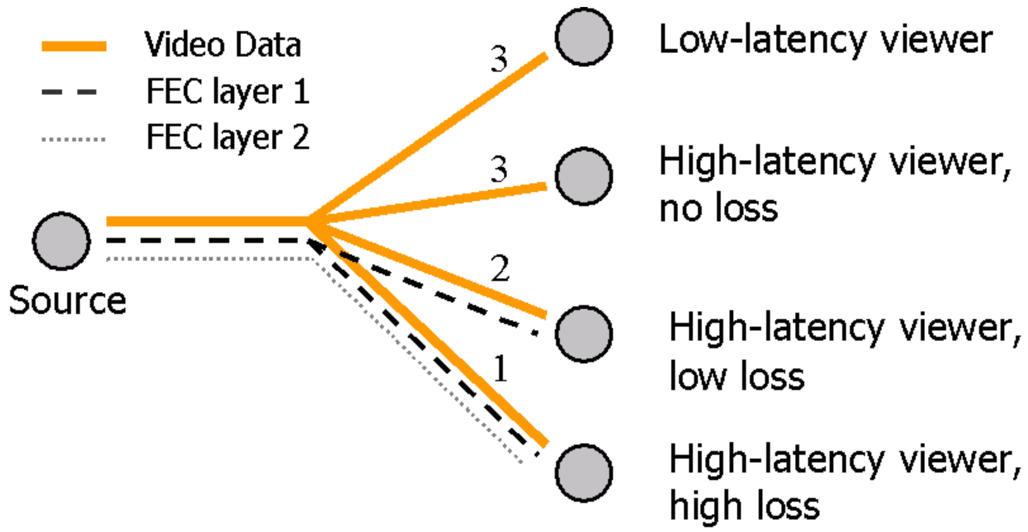


Figure 5.2: Tailoring layered FEC to different receivers having the same available bandwidth. The numbers indicate the number of data layers transmitted along each link.

Fig. 5.2 illustrates how the layered FEC scheme can be used to address the heterogeneity in packet loss rates and latency preferences of different receivers. Assume three data and two FEC layers of equal sizes are used and that all receivers have a bandwidth constraint of three layers. Regardless of packet loss rate, a low-latency viewer decides against the use of FEC layers to avoid the associated additional latency. In our example, it will subscribe to three data layers. A high-latency viewer, on the other hand, can take advantage of the FEC layers to obtain better video quality at the cost of higher latency. For example, while a high-latency viewer with no loss has no need for FEC and can afford to subscribe to only data layers at no extra latency, a high-latency viewer with low losses may choose to replace one data layer with an FEC layer to get better picture quality at the extra cost of one GOP delay. Similarly, a high-latency viewer with high losses can subscribe to one data and two FEC layers to compensate for the high packet losses at the extra cost of two GOP delay. We see that in all cases, the bandwidth constraints can be satisfied even when FEC is used. Furthermore, FEC packets are only transmitted to branches that lead to subscribers requiring FEC, thereby reducing overall network load. In contrast, traditional FEC schemes for multicast often compute a set of FEC packets which are multicast to all receivers regardless of whether they are needed [90, 98].

Given a menu of data and FEC layers, each receiver in a layered FEC implementation needs to perform two basic functions. First, it needs to determine the amount of available bandwidth; this is discussed in Section 5.6. Second, it needs to choose from the menu the optimal set of data and FEC layers that satisfies the bandwidth constraint; this is discussed in Section 5.3. The related problem of actually generating the data and FEC layers is discussed in Section 5.5.

5.3 Selection of Data and FEC Layers

In this section, we consider the problem of selecting from the menu an optimal set of data and FEC layers under a prescribed packet loss rate and bandwidth constraint.

Specifically, the chosen criterion for optimality is minimum distortion.

Under our layered FEC scheme, each receiver has an associated subscription level $\mathbf{s}$, which is uniquely specified by the number of subscribed data layers L, and the number of FEC packets used to protect each layer k_i, $0 \leq i < L$. An expression for distortion D can be computed as a function of the packet loss rate p and subscription $\mathbf{s}$. The exact dependence of D on p and $\mathbf{s}$ is characteristics of the specific compression technique being used. In this Section, we derive the distortion function for the error-resilient video compression technique used in our experiments.

Let $\mathbb{M}$ be the set of all possible subscriptions given by a fixed menu, and $R(\mathbf{s})$ the rate associated with subscription $\mathbf{s}$, we seek to find the optimal subscription $\mathbf{s}^* \in \mathbb{M}$ that minimizes the distortion $D(\mathbf{s}, p)$ subject to a bandwidth constraint of B:

$$\mathbf{s}^* = \underset{\mathbf{s} \in \mathbb{M},\ R(\mathbf{s}) \leq B}{\arg\min}\ D(\mathbf{s}, p) \tag{5.1}$$

When the search space $|\mathbb{M}|$ is relatively small, $\mathbf{s}^*$ can be effectively found by exhaustive search over $\mathbb{M}$. The computational cost is then roughly the search space times the cost to evaluate $D(\mathbf{s}, p)$. Because data layer i is more important than layer j for $i < j$, in general, we can further reduce the search space by considering only the set $\mathbb{M}' \subseteq \mathbb{M}$ of subscriptions such that $k_i \geq k_j$ whenever $i < j$. For large search space, $\mathbf{s}^*$ can be found iteratively using Lagrangian type techniques [20] if some convexity assumptions are satisfied. Even though the cost of an iteration is high compared to the cost of evaluating D, such schemes may be justifiable when the number of iterations needed is small and the search space is large.

5.3.1 Error-Resilient Video Compression

One important factor affecting distortion is the error-resilience properties of the video compression scheme. Generally, more error-resilient compression schemes have lower compression efficiency in loss-free environments. This means at the same communication rate, a less resilient compression scheme will have more bits available for FEC to compensate for the lack of error-resilience. While it is beyond the scope of this Chapter to compare various compression schemes under fixed and known loss rate and available bandwidth, it is generally agreed upon that error-resilient compression tends to work better when the channel is bursty and the loss rate is not known *a priori* [129]. There are three reasons for this. First, even with the use of FEC, video data may still be lost due to traffic burstiness, and as such, error-resilient compression can potentially result in graceful degradation. Second, it is not possible to determine the instantaneous channel conditions of networks. In practice, only delayed estimates of measurable quantities such as packet loss rates and round-trip times are available to video streaming applications. Those constraints can cause transient losses when network conditions change and before the application can adapt to the change. Without error-resilient compression, such losses may translate to catastrophic video quality degradations. Third, in a layered FEC framework, low-latency users do not utilize any FEC due to delay constraints, and as such, error-resilience compression offers better video quality for them under lossy conditions. The error-resilient but non-scalable scheme of [107] allows the use of FEC for low-latency applications, but is not applicable in a layered multicast environment due to the lack of scalability.

5.3.2 Computation of Distortion Function

In this section, we will derive the distortion function $D(\mathbf{s}, p)$ for the error-resilient compression method SVIP described in Chapter 3. One characteristic of SVIP is that the bit-stream for every group of pictures that is compressed together can be separated into M components that are of similar importance and can be decoded independently. The left hand side of Fig. 5.3 shows the M independent, equally important components that correspond to a group of picture, and also some possible bit-rates, such as R_1 to R_4 that are available by truncating the bit-stream. One possible way to use the codec in a layered multicast setting using four layers is illustrated in the right hand side of Fig. 5.3. In our example, the first layer contains the bits from the bit-stream that are necessary to decode the original video at rate R_1. The bits from each component will be sent as a separate packet to give a total of M packets for the layer per GOP. The second layer contains the additional bits that are necessary to decode the original video at rate R_2. Again, one packet is generated for each component. In such a way, subscribing to one, two, three, and four layers results in total rates of R_1, R_2, R_3, and R_4, respectively.

Fig. 5.4-(a) shows the packet loss dependence for the scheme described in Fig. 5.3 with four data layers, six components per GOP and a total of three GOP. Because the components can be decoded independently, a packet loss in one layer will cause error

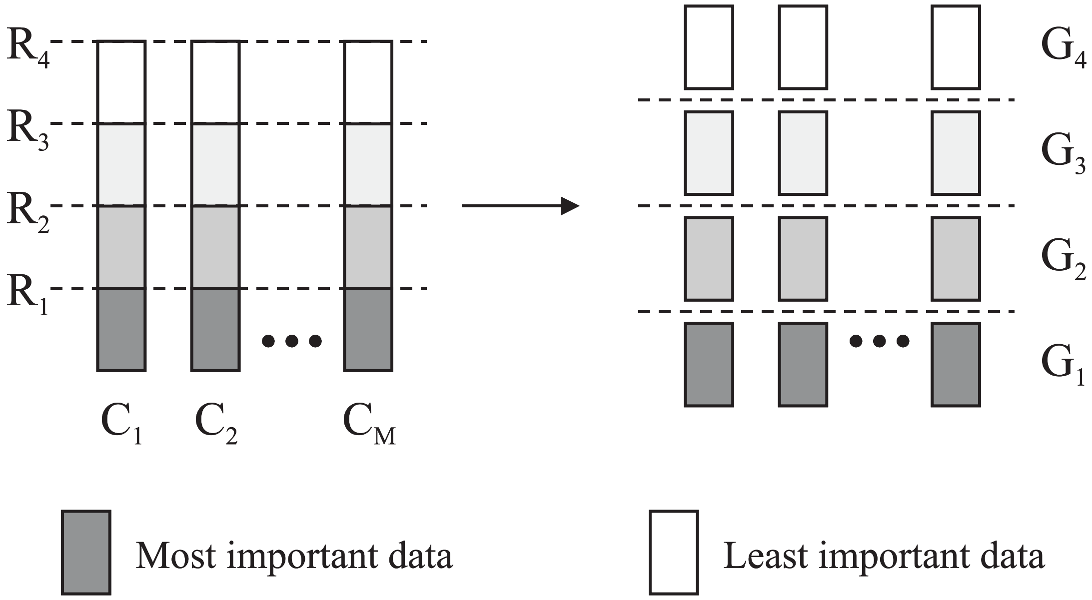
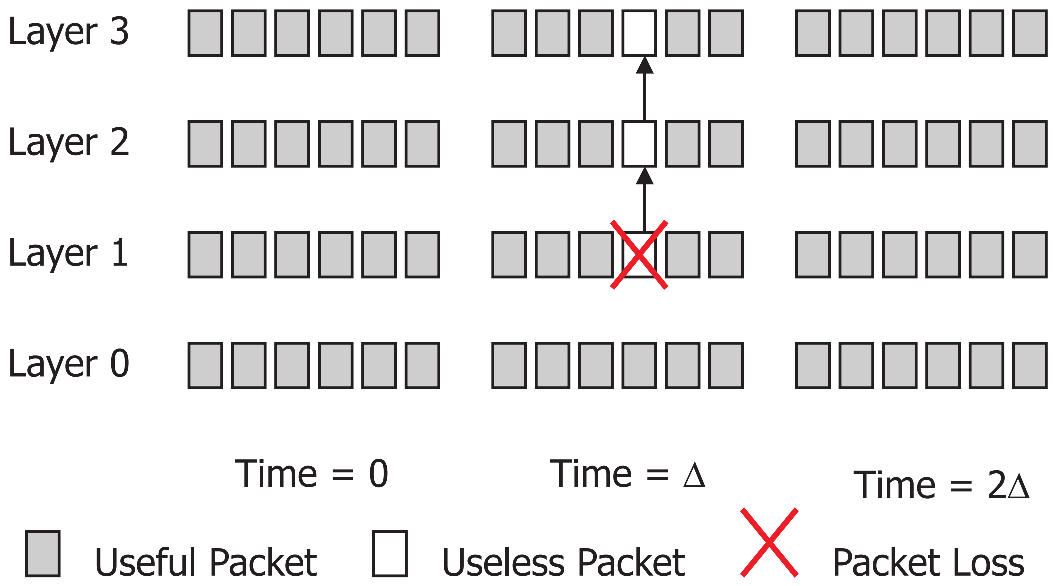


Figure 5.3: The mapping of SVIP bit-stream into multiple multicast groups (G_1 - G_4).

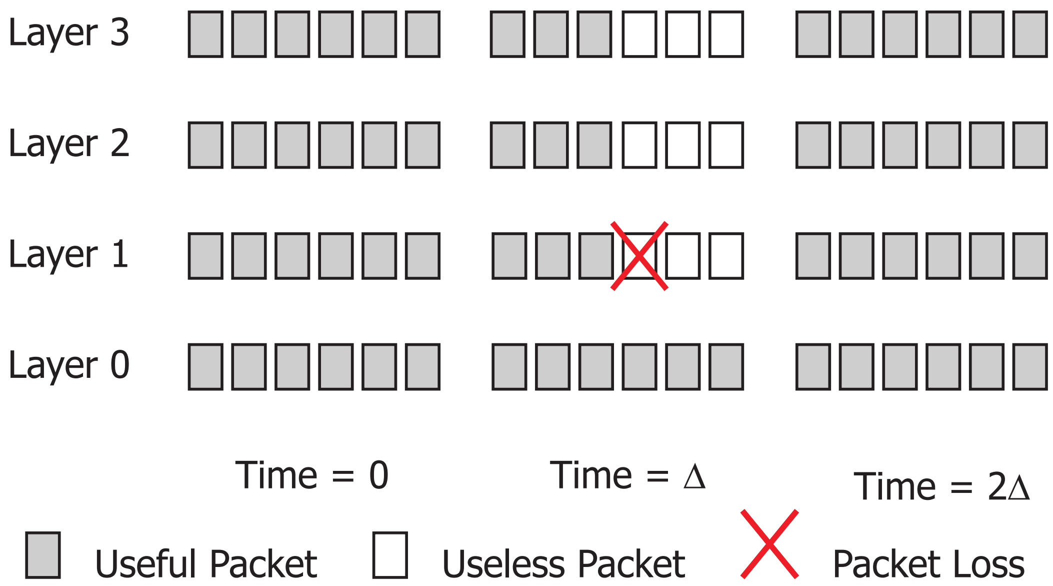
propagation into only one packet in each subsequent layers. In contrast, finely scalable but non-resilient compression schemes often fail after the first point of loss [134, 86, 68]. As a result, when a packet in one layer is lost, all subsequent packets, including some or all higher layers are rendered useless, as illustrated in Fig. 5.4-(b).

5.4 Performance of SVIP

Given that every group of video frames are compressed together to give M independent, equally important components as shown in Fig. 5.3, the total distortion for a GOP, D, can be approximated by the sum of the distortions due to the individual components $D^{(c)}$. Let $p_i^{(c)}$ be the probability that out of a total of L subscribed layers, only the first i layers of component c are decodable and with associated distortion



(a): *Error-resilient scalable video.*



(b): *Non-resilient scalable video.*

Figure 5.4: Data dependence for scalable video in a layered multicast framework.

$D_i^{(c)}$. We have,

$$\begin{aligned} D &\approx \sum_{c=1}^{M} D^{(c)} \\ &= \sum_{c=1}^{M} \sum_{i=0}^{L-1} p_i^{(c)} \cdot D_i^{(c)} \end{aligned}$$

Since all components are assumed to be of equal importance, $D_i^{(c)}$ and $p_i^{(c)}$ are independent of c, and are equal to $D_i^{(1)}$ and $p_i^{(1)}$ respectively. Hence, we get:

$$D \approx M \sum_{i=0}^{L-1} p_i^{(1)} \cdot D_i^{(1)}$$

Defining $D_i = MD_i^{(c)}$, we get:

$$D \approx \sum_{i=0}^{L-1} p_i^{(1)} \cdot D_i \tag{5.2}$$

Thus, D_i is the distortion over a GOP when i layers are used to decode all components, and can be approximated using the operational rate-distortion curve.

Let $q_i^{(c)}$ be the probability that the packet in layer i of component c is lost and cannot be recovered using FEC. Then, we have,

$$p_i^{(c)} = \begin{cases} q_i^{(c)} \prod_{k=0}^{i-1} (1 - q_k^{(c)}) & \text{if } 0 \le i < L, \\ \prod_{k=0}^{L-1} (1 - q_k^{(c)}) & \text{if } i = L. \end{cases} \tag{5.3}$$

Using equal packet sizes for data and FEC layers, and an FEC code with maximum error-correcting capability, all layer i data packets are recoverable if any M packets are received out of the M data and k_i FEC packets. Assume packets are lost with probability p, and that losses are independent and identically distributed across all data and FEC layers. Then, the layer i packet of component c is lost with probability p. Furthermore, the lost packet cannot be recovered only when less than a total of M packets are received from the remaining $M + k_i - 1$ data and FEC packets. Thus, we have:

$$q_i^{(c)} = p \cdot \left[\sum_{w=0}^{M-1} \binom{M + k_i - 1}{w} (1-p)^w \cdot p^{M+k_i-1-w} \right] \tag{5.4}$$

Equations 5.2 through 5.4 allow the computation of distortion at specific packet loss rate and subscription level of data and FEC layers. Together with Equation 5.1, we can then numerically determine an optimal set of data and FEC layers, through exhaustive search, for each prescribed bandwidth constraint and observed packet loss rate. This computation is used in two later Sections. In Section 5.5, it is used to determine how optimal FEC changes with bandwidth constraint and packet loss rate. In Section 5.7, the computation is used, in both the simulations and the experiments, to provide each receiver with an optimal subscription of data and FEC layers. A more elaborate analysis assuming different number of packets in different data layers can be found in [156].

5.5 Construction of Layered FEC

Before we look into the problem of constructing the layers menu, we need to first choose an error control code for layered FEC. An effective FEC scheme will allow recovery of lost data packets with as few redundancy packets as possible. This is achieved when the loss of any m data packets are recoverable from any m parity packets. This property is often referred to as maximal distance separable (MDS) in algebraic coding theory [147]. An example of an MDS code is the Reed-Solomon code, which has long been used in data communications to provide incremental redundancy [80]. Fig. 5.5 illustrates the use of a systematic (n, k) MDS code that takes k data packets and produces $n-k$ FEC packets, which are partitioned to form FEC layers. Let m_i denote the number of packets in the $i-$th FEC layer. Since punctured MDS codes are also MDS [147], the first FEC layer can correct up to m_1 packet losses, the maximum any error control code with m_1 redundancy packets can achieve. Similarly, the first l FEC layers achieve maximum error correcting capability by being able to correct up to $\sum_{i=1}^{l} m_i$ lost packets. Hence, there is no loss in error correcting capability by using MDS codes for layered FEC.

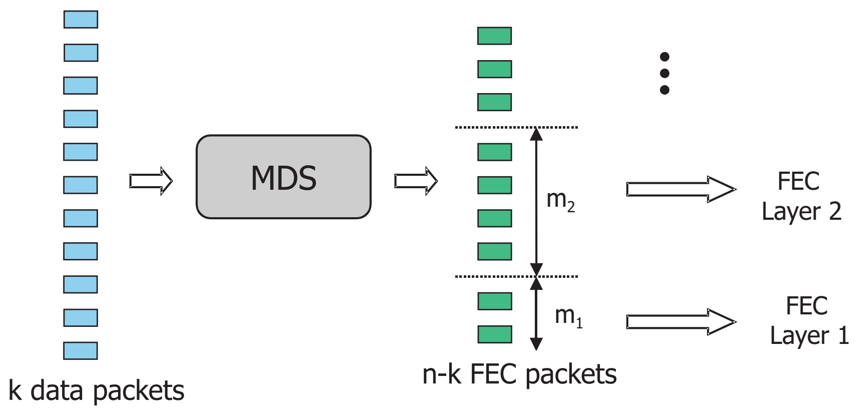


Figure 5.5: Generating FEC layers using MDS codes.

Scalable video compression algorithms typically produce embedded bit-streams which means to decode video data layer k, all lower layers 0 to $k-1$ must also be available to decode. In such cases, we call the layers *totally ordered*. In the special case when FEC protection is provided for the first data layer only, an embedded FEC-stream spread over multiple layers can be used in a prescribed order, with layer FEC-0-1 before FEC-0-2, and so forth [128]. One way to extend this approach to protect multiple video layers is to order all FEC layers corresponding to all video layers in a prescribed way, and have each receiver decide how many layers of this ordered list to which it subscribe. However, as we shall see later in this section, such total ordering unnecessarily restrict the choices for the various receivers, which may desire different sets of FEC layers, depending on their available bandwidth, experienced loss rate, and the data dependence of the scalable video compression used.

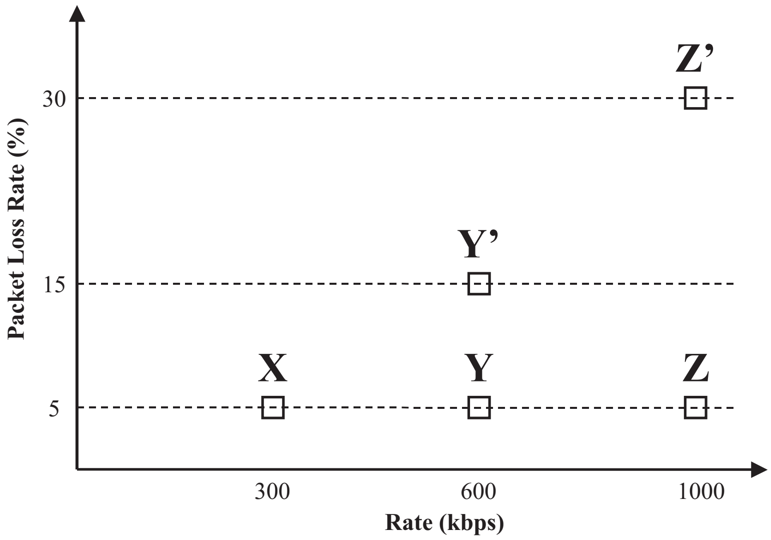
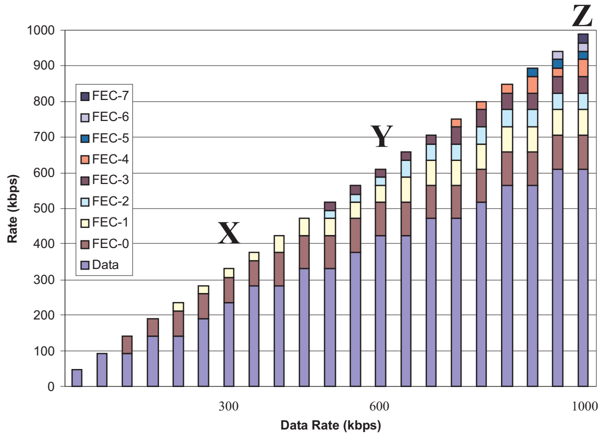
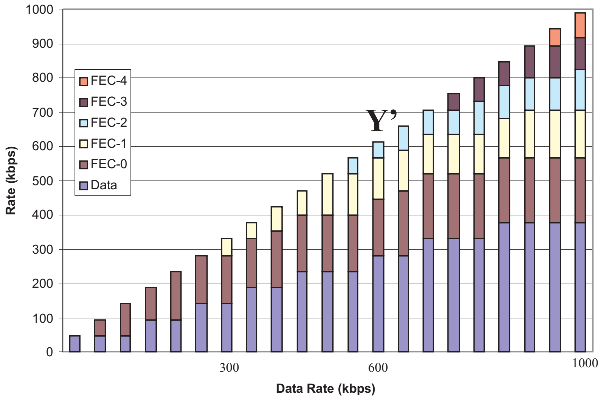


Figure 5.6: Some possible operating points on the loss-bandwidth plane.

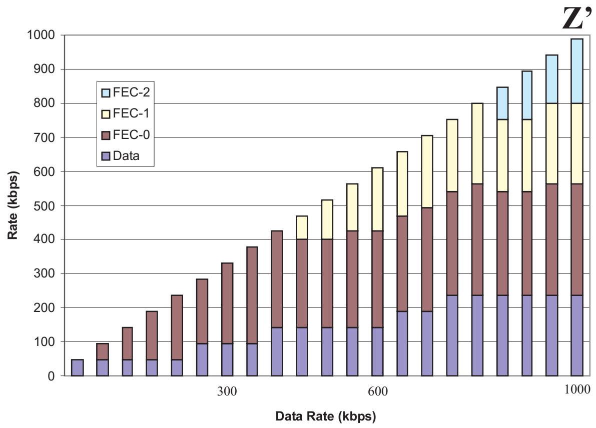
In Section 5.3, we discussed how to optimally choose a set of data and FEC layers



(a): *Optimal subscription for 5% loss.*



(b): *Optimal subscription for 15% loss.*



(c): *Optimal subscription for 30% loss.*

Figure 5.7: Optimal FEC protection for layered video source at different available bandwidth and packet loss rates. The video layers are of constant rate of 47.3 *kbps*.

given a bandwidth constraint and packet loss rate. Fig. 5.6 shows the optimal sets for receivers operating at different operating points, as characterized by different values of p and B in the p-B plane. Fig. 5.6-(a) shows the loss-bandwidth operating points of various receivers and Figs. 5.6-(b), (c), and (d) show the resulting optimal subscription levels for each operating point. The aggregate bit-rate of all the video layers combined is labeled "Data" in Fig. 5.6, while the bit-rate for FEC protection of video layer i is labeled "FEC-i". We see that for receivers operating at point X, with $B = 300$ *kbps* and $p = 5\%$, the optimal protection is to use roughly 70 and 23.5 *kbps* to protect the first and second data layers respectively, and leave the other data layers unprotected. Moving along the 5% loss line in Fig. 5.6-(a) to points Y and Z, with bandwidth of 600 *kbps* and 1000 *kbps* respectively, we see from Fig. 5.6-(b) that the optimal number of layers to protect the first data layer does not increase significantly when the total bandwidth is increased from 300 *kbps* to 600 and 1000 *kbps*. This is because the first data layer is already sufficiently protected, and additional bandwidth is used to protect other data layers instead. Thus, to satisfy the FEC needs of receivers X, Y and Z using FEC layers that are added in a fixed order, it is desirable that the first few FEC layers provide enough FEC for receiver X, with additional FEC layers extending FEC protection to more data layers. Moving along the path $X \rightarrow Y' \rightarrow Z'$ however, with loss rates of 5, 15, and 30% respectively and bandwidth of 300, 600 and 1000 *kbps* respectively, we see from Figs. 5.6-(b) to (d) that the desired protection level for both data layers 1 and 2 increases significantly from X to Y', and from Y' to

Z'. Thus, to efficiently satisfy the FEC needs of receivers Y' and Z', after the FEC need of X is met, it is desirable that additional FEC layers provide more protection to the first few data layers rather than extending protection to other data layers. Thus, it is clear that the FEC requirements of receivers Y and Z are different from those of Y' and Z'. This can also be seen by comparing the composition of FEC layers in Figs. 5.6-(b) to (d). This implies that there exists no one total ordering scheme for the FEC layers that results in optimal subscription levels for different receivers with different loss/bandwidth characteristics.

A more flexible alternative is to employ a partial ordering as follows: Each FEC layer can be used to protect only one data layer. While no explicit join order exists for different FEC layers that protect different data layers, receivers are required to join in sequence the FEC layers that protect the same data layer. In such a way, FEC layers are partitioned into sets, each of which protects a different data layer. A receiver then need only decide for each set how many FEC layers to subscribe to, depending on its total available bandwidth and experienced loss rate. In such a way, each receiver can individually decide how much FEC to protect each data layer without the unnecessary constraints of a single ordering. While this approach is more flexible, it does require more computation compared to a total-ordering approach. This is because finding the optimal combination of data and FEC layers is a linear search when total-ordering is used, but a combinatorial search when partial-ordering is used. We show experimental results corresponding to partial-ordering in Section 5.7.

Thus, our goal in system design is to provide sufficient number of FEC layers for each data layer in order to enable heterogeneous population of receivers to optimally choose the appropriate subscription level from the available "menu".

Using many thin FEC layers generally yields finer grain protection. This however, consumes a large number of multicast addresses and induces considerable overhead costs. In practice, the number of multicast addresses is limited. To better utilize the multicast address space, one can strategically use the FEC layers to cover a typical region in the p-B plane. In that case, a receiver outside the typical region may not find the best possible FEC from the menu, but can nevertheless still optimize within the menu. The tradeoff between the number of multicast addresses used and the granularity of available protection requires a cost/benefit analysis, and is beyond the scope of this chapter. Alternative schemes based on collecting receiver interests to adaptively change the source [3, 141] require additional protocols to provide closed-loop feedback to the sender, thereby limiting scalability to large group of receivers. However, given channel conditions of all receivers, an optimal menu that minimizes the average distortion of all receivers can be found using iterative techniques [21].

5.6 Equation-based Multicast Rate Control

A central part of a layered multicast system is the rate control algorithm at the receivers. The traditional algorithm drops a layer when congestion is detected and probes for more bandwidth when losses are absent [84, 140]. One early example is receiver-driven layered multicast (RLM) [84]. Another example is receiver-driven layered congestion control (RLC) [140] which achieves TCP-friendliness by using exponentially distributed layer sizes and carefully choosing the time instants at which layers can be added. Since the act of probing for bandwidth may cause congestion, leading other receivers to inadvertently reduce their subscription levels, the difficult tasks of coordinating probes [140] and communicating their results are necessary [84]. While such coordination techniques may be effective for a single multicast session, there are no existing mechanisms to coordinate probes across different sessions, leading to potentially unfair allocation of bandwidth. A more recently proposed scheme, equation-based rate control (ERC), does not involve probing [139, 129]. Instead, the available bandwidth is computed directly at the receivers based on measured quantities such as packet loss rate.

One characteristic of ERC is its ability to allocate bandwidth fairly among competing sessions [77]. For example, two simultaneous video multicast sessions which share one source-destination pair but with different starting times, may initially have different rates. However, since the two sessions will measure the same channel conditions, and therefore compute the same rates under ERC, they will eventually converge towards the same target rate for the shared source-destination pair. This is achieved by adding layers when the current layer is below the target rate and dropping them otherwise. In contrast, layer adding/dropping techniques based on simple thresholding, as employed by RLM, do not have a common target rate for competing sessions.

As a result, two sessions sharing the same bottle-neck but with different initial rates will likely add and drop layers synchronously rather than converging towards fair sharing [50]. In effect, ERC achieves fairness by exploiting packet loss rate as an implicit side-information to compute a fair target rate for competing sessions.

Existing ERC implementations are typically based on the TCP-friendly equation [81] which gives the throughput T of a TCP connection under packet loss rate p and round trip time rtt,

$$T = \frac{1.22 \cdot MTU}{rtt \cdot \sqrt{p}} \tag{5.5}$$

where MTU, the maximum transport unit, is a constant often taken to be 500 bytes. While Equation 5.5 and its variants may be used directly to provide TCP-friendliness [139, 119], its use in a multicast context has several drawbacks. First, it requires the inherently non-scalable rtt measurement [139], which has partly been addressed by [119] recently. Second, as shown in Fig. 5.8, a dependence on rtt can cause two receivers, R_1 and R_2, connected to the same source but with different rtt to have different rates even if they share the same bottleneck, resulting in under-utilization of resources.

Generally speaking, our proposed layered FEC scheme can be used with different implementations of equation-based rate control algorithm, be it dependent on round-trip time and thus TCP-friendly, or independent of round-trip time to avoid network under-utilization. In this thesis however, we choose to employ an equation-based scheme that that does not depend on rtt but replaces it by a constant in Equation. 5.5. The choice of the constant dictates the aggressiveness of ERC in consuming bandwidth under different packet loss rates. While actual *rtt* vary between hundreds of milliseconds for inter-continental traffic to under 10 milliseconds for local traffic, we choose for our experiments in Section 5.7 an intermediate value of 80 milliseconds, which yields reasonable communication rate as a function of packet loss rate for the experiments.

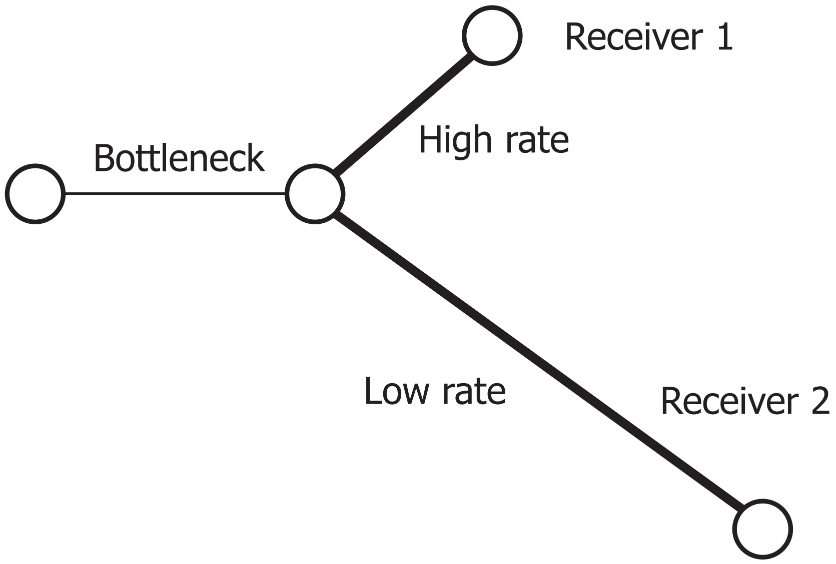


Figure 5.8: Under-utilization of network resources caused by multicast rate control algorithms with dependence on rtt.

Another desirable feature in multicast rate control algorithms for multimedia is to adapt slowly in time to changing network conditions. There are two reasons for this. First, while scalable video is well suited to time varying channels, rapid variations in communication rates results in visually unpleasant video. Second, even after a multicast group is unsubscribed, packets in that group will still be delivered until pruning of the multicast tree is completed. Such leave-latency can last several seconds [50], making it impractical to adapt on a time-scale shorter than a few seconds. For instance, when losses are bursty, direct application of Equation 5.5 using instantaneous estimates of p may cause the estimated rate to fluctuate rapidly. Such short term fluctuations can be suitably reduced by smoothing algorithms. In our ERC implementation, an averaging time-window is used at the receivers to estimate the instantaneous packet loss rate $\hat{p}$, which is averaged over all data and FEC layers. Generally, a smaller window is able to track changes in network conditions faster than a larger window, but at the expense of larger overhead. Since a multicast rate control algorithm should not react faster than a few seconds, the higher complexity of a sub-second window is not justified. To measure $\hat{p}$ in our experiments in this chapter, we therefore choose a three seconds window to provide sufficient responsiveness to changing network conditions. The p used in rate control is then obtained by smoothing $\hat{p}$ using first order auto-regression: $p \rightarrow \lambda p + (1 - \lambda)\hat{p}$. The parameter λ controls the tradeoff between responsiveness and smoothness of output rates, and a value of 0.9 is found experimentally to yield a reasonable tradeoff for video communications.

To evaluate the performance of ERC compared to existing implementations of RLM and RLC when multiple multicast sessions are present, we perform simulations using the network simulator (*ns*) and the dumbbell topology in Fig. 5.9. In the contemporary and independent work of [119], the TCP-friendliness of RLC is examined when only a single multicast session is present. Our simulation differs in that we are interested in the interaction and fairness between independent and competing multi-

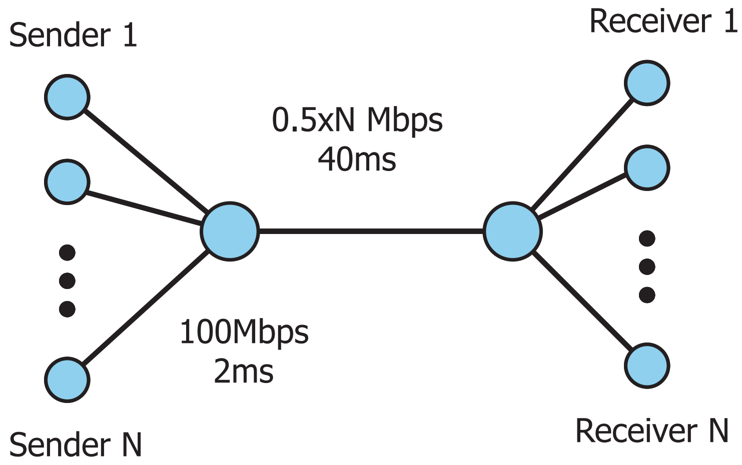


Figure 5.9: “Dumbbell” topology.

cast sessions. A constant layer size of 47 *kbps* is used for ERC and RLM to result in fine rate granularity without using an excessive number of multicast addresses. Since RLC requires exponential layer sizes, a base layer size of 47 *kbps* is used, with each additional layer doubling the total bandwidth. The simulation consists of 4 sessions between 4 different pairs of senders and receivers as shown in Fig. 5.9. The throughput of the 4 sessions as a function of time are shown in Fig. 5.10. Of the 4 sessions, two are ERC sessions with one receiver, and two are TCP sessions. The 4 protocol instances share a bottleneck bandwidth of 2 *Mbps* with the two ERC instances starting at time 0 and 300 seconds respectively and the TCP instances at 100 and 200 seconds, respectively. We see that the ERC instances finally settle to similar eventual rates despite the fact that at $t = 300$s they have different rates. While Fig. 5.10 shows fair sharing of bandwidth by the 4 protocol instances, fair sharing of bandwidth between TCP and ERC instances does not necessarily happen in general, since unlike TCP, our ERC implementation does not depend on round trip time. As mentioned earlier, this lack of dependence on *rtt* is done deliberately in order to achieve scalability and avoid under-utilization of network resources. Regardless of fair sharing with TCP, we have shown experimentally that different ERC instances share the available bandwidth fairly amongst themselves.

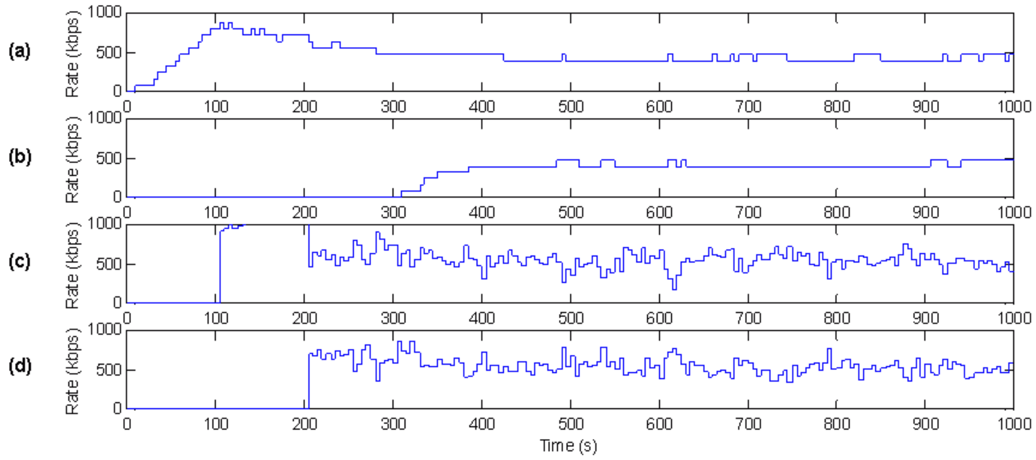


Figure 5.10: Sharing of bandwidth by (a) ERC flow starting at time 0, (b) ERC flow starting at time 300s, (c) TCP flow starting at time 100s, and (d) TCP flow starting at time 200s.

Figs. 5.11 and 5.12 show the corresponding results for RLM and RLC, respectively. We see in Fig. 5.11 that the 2 RLM instances fail to share bandwidth fairly. In particular, the first RLM instance, starting at time 0, hogs most of the bandwidth, and since the packet loss rate is lower than the drop threshold, it refuses to relinquish bandwidth to later TCP and RLM instances. In Fig. 5.12, the relatively large layer size of RLC near 500 *kbps* induces vigorous reaction and oscillation when layers are

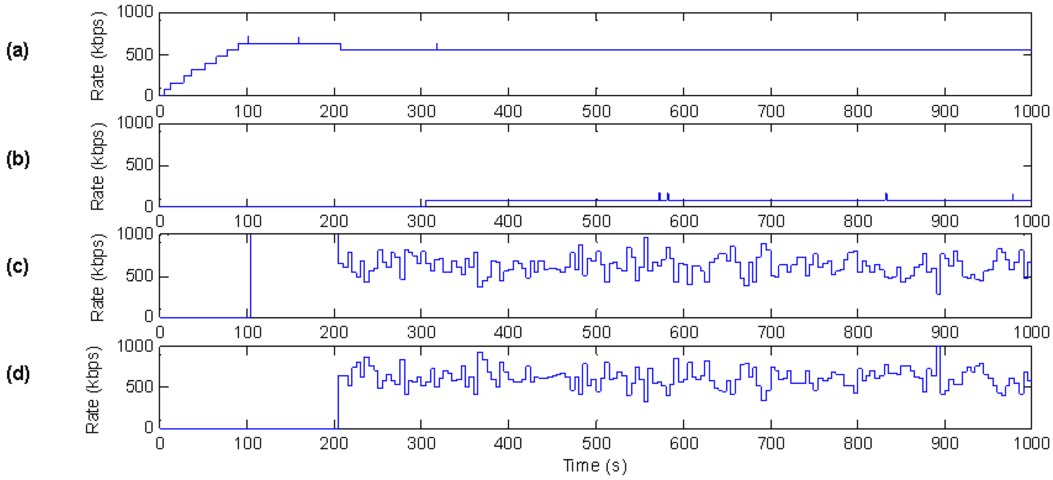


Figure 5.11: Sharing of bandwidth by (a) RLM flow starting at time 0, (b) RLM flow starting at time 300s, (c) TCP flow starting at time 100s, and (d) TCP flow starting at time 200s.

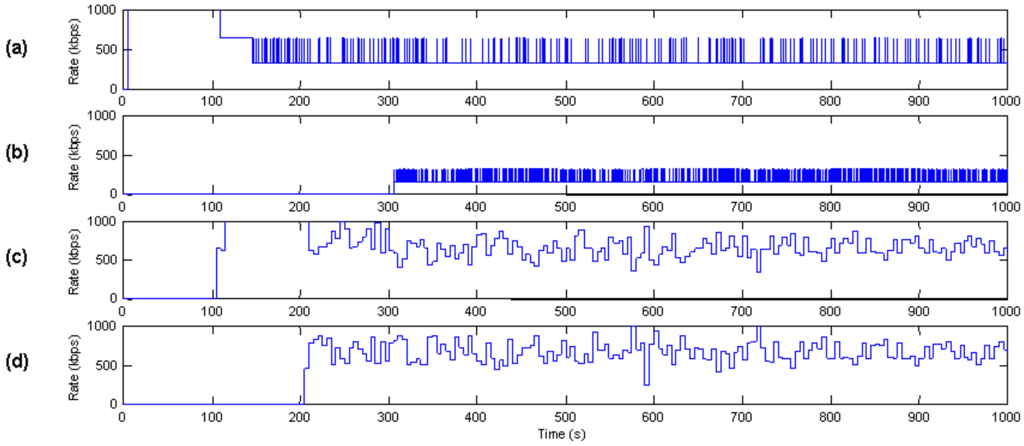


Figure 5.12: Sharing of bandwidth by (a) RLC flow starting at time 0, (b) RLC flow starting at time 300s, (c) TCP flow starting at time 100s, and (d) TCP flow starting at time 200s.

added. In particular, the first RLC instance starting at time 0 consistently adds large layers, causing congestion, to which the second instance, starting at $t = 300$s would react by dropping layers. This can be attributed to the fact that even though one

RLC session can prevent oscillation amongst its receivers through synchronization, when there are multiple sessions, the lack of inter-session synchronization may give rise to oscillations.

To further quantify the fairness properties of the various protocols, a fairness index can be used [140]. A fairness index F of a set of N rates X_i can be defined to be a function between 0 and 1, that is 1 when all the rates are identical, and close to 0 when a large fraction of the sum of the rates $\sum_i X_i$ is concentrated in one rate, say X_j. One such function is:

$$F = 1 - \frac{1}{2(N-1)} \sum_{i=1}^{N} \left| \frac{X_i}{\mu} - 1 \right| \tag{5.6}$$

where μ is the average rate. Thus, $F(1,3,5,7,9) = 0.7$ and $F(50,51,52) = 0.990$, matching the intuition behind fairness index.

Fig. 5.13 summarizes the fairness indices for ERC, RLM, and RLC on the dumbbell topology of Fig. 5.9 using two layer size organization methods. Under the *linear* organization, layers are of equal size so that the aggregate bandwidth is a linear function of the number of layers. On the other hand, under the *exponential* organization, the aggregate bandwidth doubles with each additional layer. The bottleneck bandwidth is adjusted proportionally to the number of flow to result in 500 *kbps* per flow. The results are obtained by running simulations of 1000 seconds duration and measuring the average rate of each flow using the last 300 seconds to exclude transient effects. Protocol instances are started at random times between 0 and 500 seconds, and reported results are averaged over 3 independent runs. We see that ERC using

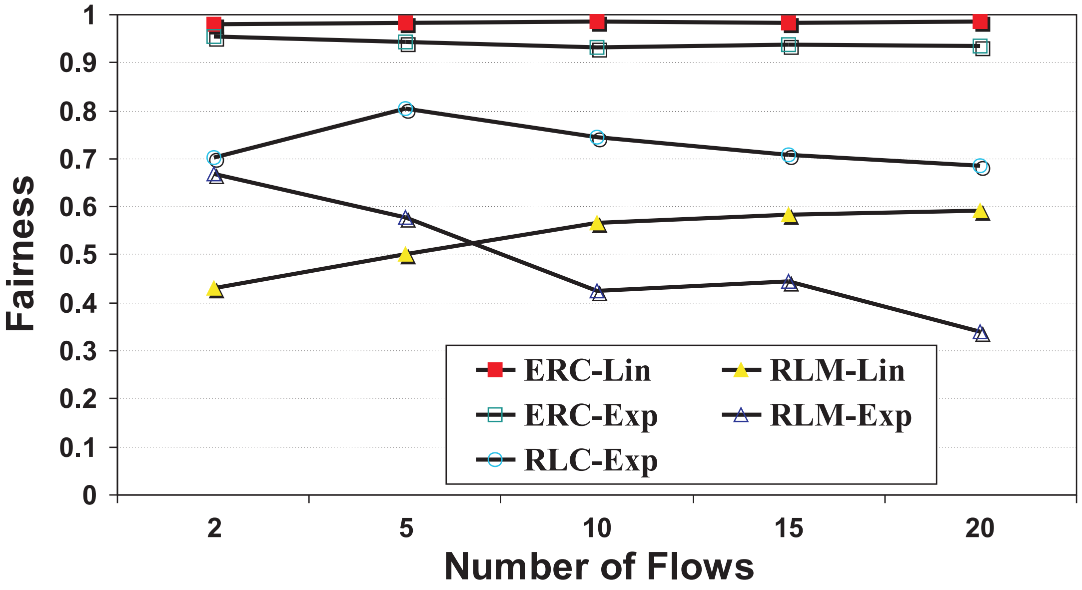


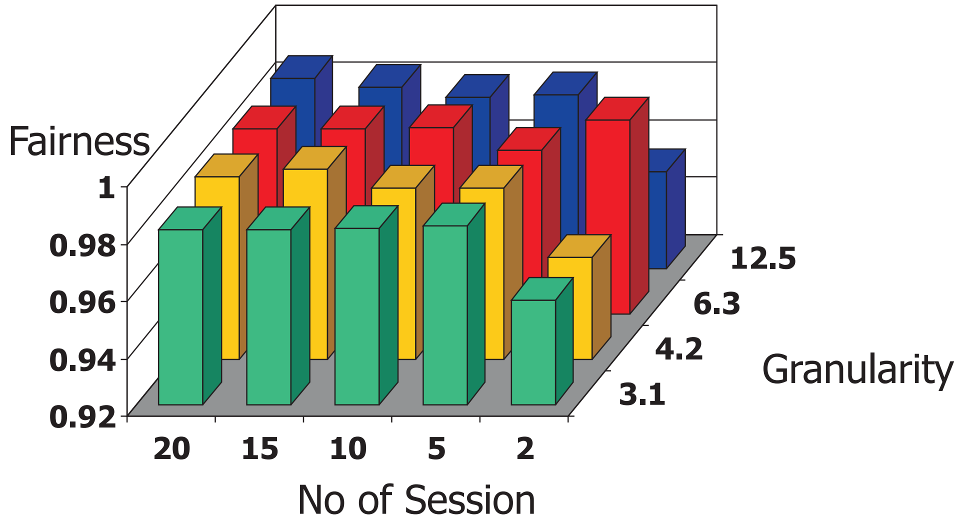
Figure 5.13: Fairness comparisons of ERC, RLM and RLC using linear and exponential layer sizes.

both linear and exponential layer sizes score significantly higher fairness index than both RLM and RLC.

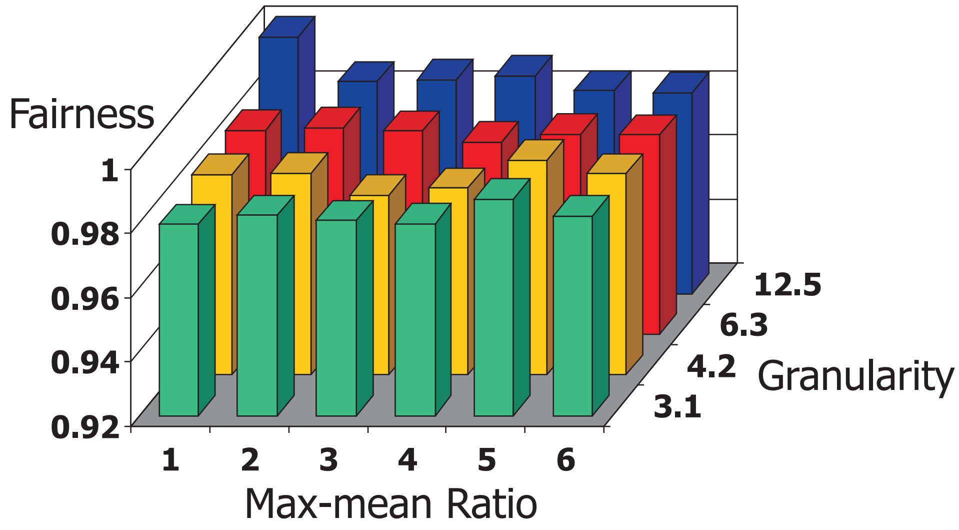
Finally, to further examine the fairness property of ERC, we compute the fairness index as a function of the number of flows, granularity, and max-mean ratio. Granularity is defined as the number of constant size layers that can be supported by a flow's fair share of bandwidth, and is obtained by varying the layer sizes while fixing the bandwidth so that each flow has 500 *kbps* on the average. Max-mean ratio is the ratio of the maximum video source rate available at the sender to the average fair rate, and signifies the worst-case bandwidth hogging a session can achieve. For example, when max-mean ratio is close to 1, the amount of bandwidth a session can hog is close to its fair share, thereby guaranteeing fair allocation. On the other hand, when max-mean ratio is much larger than one, a session can potentially hog much more bandwidth than its fair share, leading to potentially unfair allocations. It is varied by changing the maximum video source rate available at the sender. Five independent runs are performed and the results are averaged and shown in Fig. 5.14. We observe that high level of fairness, above 0.95 in all cases and close to 0.98 in most cases, is achieved across the parameter space. From Fig. 5.14-(a), we see that fairness index is generally lower when the number of sessions is two. This is because for very small number of sessions, the resulting change in loss rate due to a receiver adding or dropping a layer is large, causing large variations in measured average packet loss rate, and thus transmission rate. In Fig. 5.14-(b), we see that fairness is largely independent of the max-mean ratio and granularity when 10 sessions are used. This is because with the multiplexing of at least a moderate number of sessions, relatively consistent measurements of loss can be maintained, giving rise to fair sharing of bandwidth.

5.7 Video Transmission Results: Simulations and Experiments

This section describes simulations and actual MBONE experiments using layered FEC with scalable video coding algorithm. Scalable compression technique described in Chapter 3 and equation-based rate control described in Section 5.6 are used. The purpose of the simulations is to evaluate the scheme under varying network loads.



(a): *Using flows of maximum rate 1.5 Mbps.*



(b): *Using 10 sessions.*

Figure 5.14: Fairness of ERC under different conditions.

Actual MBONE experiments, on the other hand, provide realistic testing conditions. A 25 second clip from the movie "Raiders of the Lost Ark" of size 320×224 at 12 *fps* is used as the video source. It is compressed using a GOP of 4 for low latency and

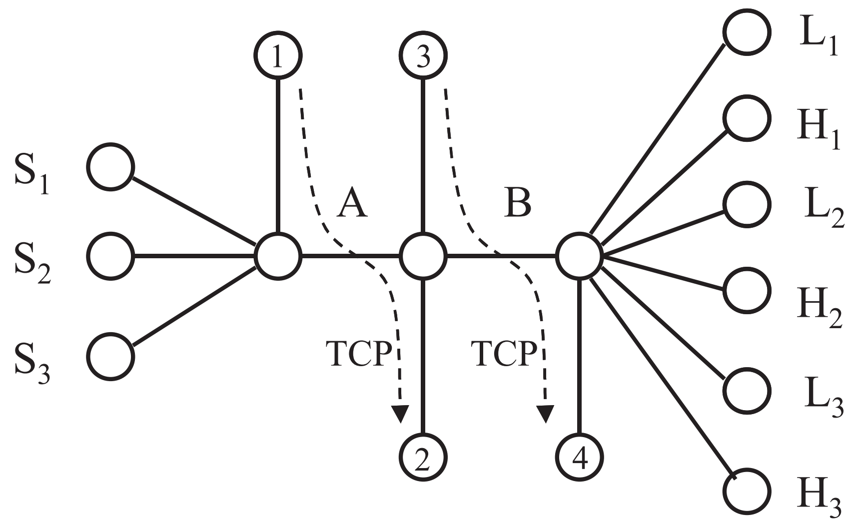


Figure 5.15: The modified "dumbbell" topology.

repeatedly transmitted.

Fig. 5.15 shows the modified dumbbell topology used in the *ns* simulation. Three independent sources S_1, S_2 and S_3 are started at random times between 0 and 50 seconds. All three sources use the same menu of data and FEC layers, which contains 10 layers of data at 100 *kbps* each, and 10 FEC layers each with 50 *kbps*. Five FEC layers are used to protect the first data layer, while 3, 1 and 1 FEC layers are used to protect the second, third and fourth data layers, respectively. This configuration is optimized for packet loss rates in the range of 5 to 15%. Each source S_i has two receivers, L_i, a low-latency viewer who will not use any FEC to achieve low latency, and H_i, a high-latency viewer who is willing to trade delay for better video quality by taking advantage of the proposed layered FEC method.

To serve as cross traffic, N instances of TCP from node 1 to node 2, and another

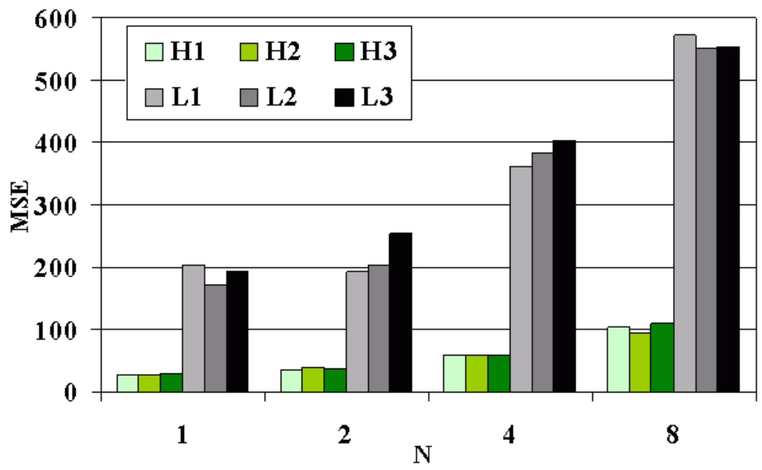


Figure 5.16: Average MSE experienced by low-latency (L_i) and high-latency (H_i) viewers under different number of TCP cross traffic (N).

N instances from node 3 to node 4 are each started at a random time between 0 and 50 seconds. The links A and B, shared by the layered FEC sources and TCP, each has a bandwidth of 3 $Mbps$. This bandwidth is barely enough to carry the 10 data layers of 100 $kbps$ for the 3 sources. By setting N to 1, 2, 4, and 8, average loss rates of 4, 5, 8, and 12% are sustained respectively on the shared link, yielding available bandwidths of 620, 550, 420 and 340 $kbps$.

Fig. 5.16 shows the average distortion, measured in mean squared error (MSE), of the low and high latency viewers, for different numbers, N, of TCP cross-traffic. As we expect, the average distortion experienced by the high-latency viewers is significantly lower than that of the low-latency viewers under the same conditions. By using layered FEC, the MSE is reduced by approximately a factor of 6 at the cost of an additional delay of at most 10 GOPs or 3.3 seconds. Due to the adaptability of layered FEC to different channel conditions, we see that the average distortion increases only slightly as the number of cross-traffic flows is increased when layered FEC is used. Fig. 5.17 shows the distortion traces as a function of time observed by the low-latency and high-latency viewers L_1 and H_1. We see that the use of layered FEC not only yields a lower distortion, but also smaller variation in distortion over time. The large variation in distortion for the low-latency viewers is caused by traffic burstiness. Even though the average loss rate may be as moderate as 4%, the instantaneous loss rates during a GOP time or 1/3 second can sometimes reach 30%. The problem is aggravated by the small number of packets per layer per GOP. Supporting reasonable packet sizes and layer sizes, such as 400 bytes and 100 *kbps* respectively results in a small number of packets per layer per GOP, such as 10. This means that a single packet loss in a data layer corresponds to 10% loss of data in the GOP, regardless of how low the average packet loss rate is. By sufficiently protecting only the more important data layers, lower average distortion and smaller variation in distortion can be achieved by layered FEC. Fig. 5.17 shows that variation in distortion remains low as N increases.

To illustrate the importance of delaying transmission of FEC layers, the above simulation is repeated with the modification that all FEC layers for a GOP are transmitted in the same time slot as the data layers. The distortion traces for viewer H_1 is shown in Fig. 5.18 for different number, N, of TCP cross-traffic. We see that while results similar to Fig. 5.17 are obtained when N is 1 or 2, occasional "peaks" of high MSE appear for $N = 3$ and 4 when transmission of FEC layers are not delayed.

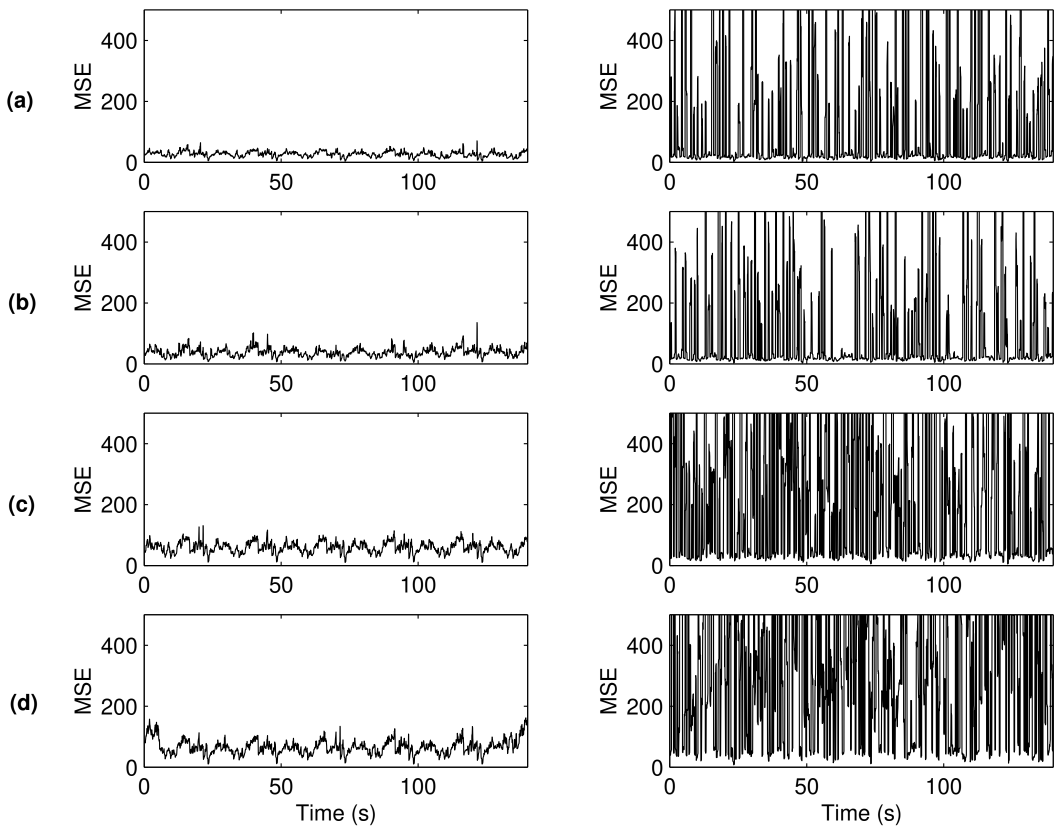


Figure 5.17: Distortion traces for receivers using layered "staggered" FEC (left) and not using layered FEC (right). The numbers of TCP cross traffic used are (a) 1, (b) 2, (c) 4, and (d) 8 respectively.

Such peaks are the result of bursty losses, which induce heavy packet drops to both data and FEC layers of a GOP if they are transmitted concurrently. In this case, since data and FEC layers are of size 100 and 50 *kbps* respectively, even if only one FEC layer is used to protect the first data layer, bursty losses of over 33% are needed to cause distortion peaks. Such large bursts are less likely when N is small, resulting in low distortion traces in Fig. 5.18 for $N = 1$ and 2.

We next describe our MBONE experiments using 3 hosts located at Berkeley,

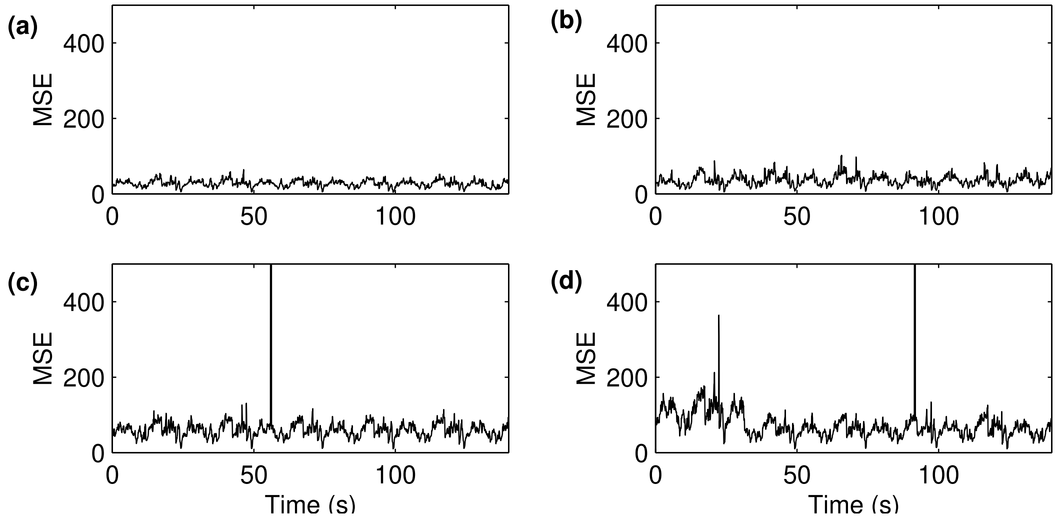


Figure 5.18: Distortion traces when all FEC layers are transmitted in the same time slot as data layers. The numbers of TCP cross traffic used are (a) 1, (b) 2, (c) 4, and (d) 8 respectively.

Indiana, and Sweden. A menu consisting of 6 data layers of 100 *kbps* each and 6 FEC layers of 50 *kbps* each is used. Of the 6 FEC layers, 4 and 2 are used to protect the first and second data layers respectively. This configuration was chosen because the first data layer always needs more protection than the second data layer and so forth. Furthermore, from Fig. 5.6, FEC is needed for data layer 3 only at relatively high available rates of 600 *kbps* and above, which is unlikely in our experiment. The video source is located at Indiana. One low-latency and one high-latency viewers is simultaneously present at Berkeley and Sweden.

While low packet loss is observed for video flowing from Indiana to Berkeley, relatively bursty losses appear in the video received at Sweden. The low loss observed for the Berkeley to Indiana flows may be due to the existence of a high-speed Internet2

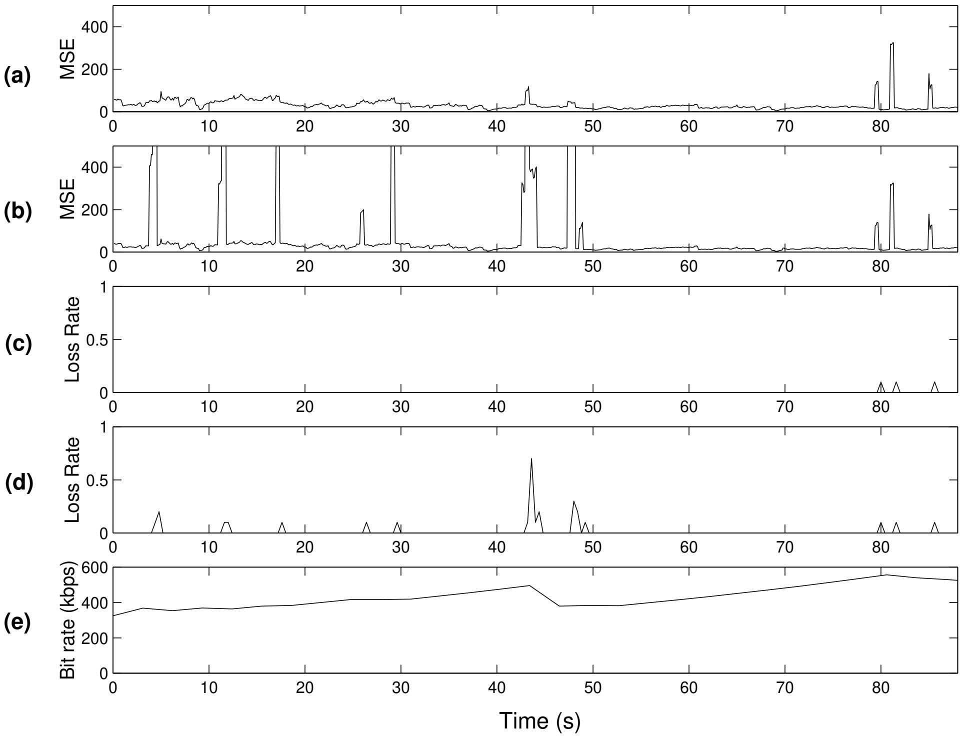


Figure 5.19: *MBONE transmission from Indiana to Sweden (10 am, 12/14/99). (a) MSE when layered FEC is used, (b) MSE when no FEC is used, (c) effective packet loss rate in first data layer after correction by FEC, (d) packet loss rate for first data layer, and (e) available bandwidth calculated by ERC rate controller.*

connection between the two sites.[1] Figs. 5.19 (a) and (b) show the distortion or MSE trace in the absence and presence of layered FEC for transmission from Indiana to Sweden. The packet loss rate seen by the first data layer after and before correction by FEC are shown in Figs. 5.19 (c) and (d) respectively. We see that the use of layered FEC helps to significantly reduce the effects of bursty losses, which are manifested in "peaks" of high MSE. However, the use of FEC reduces the bandwidth available for data and when there is no packet loss, the resulting distortion is slightly higher

[1]Both sites are participants in the Abilene network which is part of the Internet2. The list of participants in Abilene can be found at http://www.ucaid.edu/abilene.

with layered FEC. While the burst near $t = 43$s causes more packet losses than the burst near time 80 seconds, the resulting distortion is higher in the latter case when layered FEC is used. This is because the prolonged period of loss-free transmission between time 50-80 seconds causes the high-latency receiver to abandon FEC protection, thereby increasing susceptibility to sudden losses. The average MSE of the traces are 31.8 and 82.7 with and without layered FEC respectively.

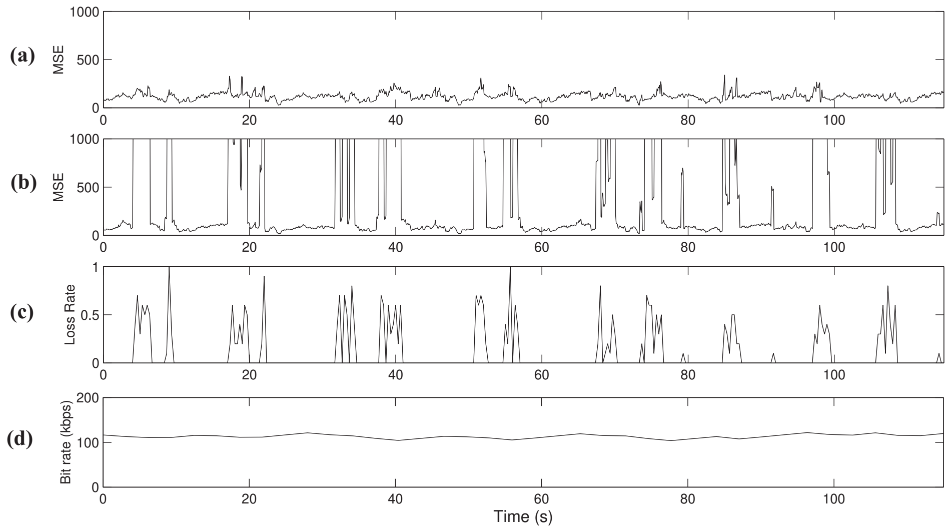


Figure 5.20: *MBONE transmission from Indiana to Sweden (2 pm, 15/5/00). (a) MSE when layered FEC is used, (b) MSE when no FEC is used, (c) packet loss rate for first data layer, and (d) available bandwidth calculated by ERC rate controller.*

The same MBONE experiment is repeated at a later time to verify the effectiveness of our technique. Again, no significant loss is observed between Indiana and Berkeley, while heavy and bursty losses are observed between Indiana and Sweden. Figs. 5.20 (a) and (b) show the MSE trace in the presence and absence of layered FEC for the Indiana-Sweden communication. Results are similar to Fig. 5.19 in that the use of layered FEC not only results in a lower average MSE of 120 compared to 450 when

FEC is not used, but also causes smaller variation in distortion. Fig. 5.20 (c) shows the packet loss rate seen by the first data layer. Despite bursty losses that often exceed 50%, all first layer data packets are recovered through the use of the layered FEC technique.

5.8 Conclusions

We have proposed the use of layered FEC as an error control method that allows each receiver to individually trade-off latency for better reception quality. The scheme can be combined with layered multicast to provide simultaneously flow and error control. This scheme is efficient in that FEC is provided only for the more important layers of the video data, and that there is no loss in error correcting capability by imposing a layered FEC structure. Simulation and MBONE experiments demonstrated the utility of the scheme.

Chapter 6

Streaming MPEG Video over Delay and Loss Differentiated Networks

Differentiated services (DiffServ) is currently under investigation by the IETF to provide relatively simple and coarse traffic differentiation in the Internet. In this Chapter, we present a framework for transmitting MPEG-compressed video by dividing the bit-stream into sub-streams of different delay and loss requirements. The sub-streams are then transported using multiple DiffServ traffic classes of different bandwidth, transmission delay, and packet loss characteristics. The use of multiple traffic classes to carry video improves network utilization as packets are transmitted in traffic classes with QoS commensurate to their requirements, rather than all in the "best" class. This technique is related to the existing use of scalable coding for loss-differentiated networks. However, in this work, we consider non-scalable MPEG video due to the abundance of existing content, and also consider loss and delay differentiations simultaneously. We present a number of packet classification schemes for MPEG bit-streams based on delay and loss characteristics of the data, and compare them using commercial DVD content. We simulate transmission of DVD movies over a DiffServ enabled network, and show a distortion reduction of over 4 dB using a packet classification scheme optimized for loss. Another packet classification scheme optimized for delay is shown to reduce end-to-end playback time by 30 ms as compared to packet classifiers that treat the MPEG stream as homogeneous.

6.1 Introduction

A large and ever-increasing body of digital video content is being stored in the form of MPEG-compressed bit-stream. In particular, content in the popular DVD and VCD formats is compressed using MPEG-2 and MPEG-1 respectively, making MPEG the dominant format for entertainment video. Nevertheless, MPEG is primarily designed for storage purposes, and as such, is ill-suited for transmission over best-effort packet networks [13].

One possible approach to facilitate communication of MPEG and other media types that require enhanced quality of service (QoS) is to augment best-effort packet-switched networks with service differentiation. In such a network, traffic that requires better QoS can be preferentially treated as compared to traffic that does not require preferential treatment. The Integrated Services (IntServ) model of the Internet, for instance, is an attempt to provide end-to-end QoS guarantees in terms of bandwidth, packet loss rate, and delay, on a per-flow basis. Availability of such QoS guarantees would greatly facilitate the streaming of existing video content and is achieved through explicit resource allocation using the Resource Reservation Protocol (RSVP) [146]. However, the high complexity and cost of deployment of the RSVP-based service architecture, together with its lack of scalability, eventually led the Internet Engineering Task Force (IETF) to consider other service differentiation mechanisms in the Internet. The Differentiated Services (DiffServ) model, in particular, is specifically designed to have low complexity and to be easily deployable, at the expense of more relaxed QoS guarantee than IntServ [9]. Thus, some complexity needs to be shifted from the network to the application in moving from IntServ to DiffServ. Under DiffServ, service differentiation is no longer provided on a per-flow basis, but rather on aggregates of flows identified by different *code-points* or tags. Thus, packets with the same tag are given the same treatment under DiffServ regardless of where they originate from.

There are a number of ways to exploit service differentiation for video communication in a DiffServ Internet. Given a fixed number of traffic classes, one strategy is to assign different data types to different classes. For example, one traffic class may be used for video, one for interactive data applications such as web browsing, and

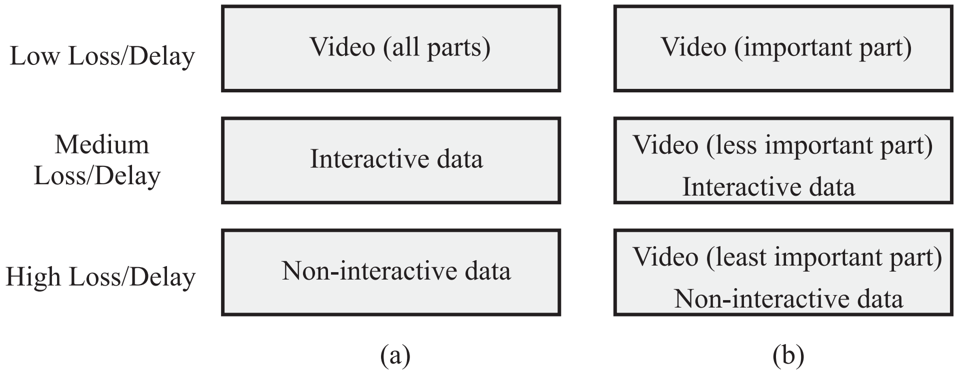


Figure 6.1: Transport of video in a DiffServ network using (a) a single traffic class, and (b) multiple traffic classes.

one for non-interactive data transfer such as email and file transfer, as illustrated in Fig. 6.1-(a). By providing lowest loss and delay to video followed by interactive data and non-interactive data respectively, the user expectations for the different applications can be better met. Despite the fact that data traffic is loss sensitive, it can be transported using higher loss classes. This is because data traffic is insensitive to delay, and the use of end-to-end retransmission protocols such as TCP can effectively translate loss into additional delay and delay-jitter.

Our approach in this Chapter is to use multiple traffic classes even within a single video application. This technique is implemented by decomposing a video bit-stream into multiple sub-streams of different delay and loss sensitivities, and assigning each sub-stream to a different traffic class with a different QoS characteristic, as illustrated in Fig. 6.1-(b). This "multi-class" approach generally achieves better utilization of network resources than the "single-class" approach due to the ability to transmit packets using traffic classes with commensurate QoS. Another advantage of the multi-class approach is the fine granularity tradeoff between network resources and reception quality. This results in a spectrum of cost-performance tradeoff in transmitting video over DiffServ networks.

To facilitate transmission of the large body of existing MPEG content using multiple traffic classes in a DiffServ Internet, methods of classifying and tagging an MPEG-compressed bit-stream into different traffic classes are needed. This Chapter presents one such tagging method for transmission over a network that offers multiple traffic classes with different bandwidth, transmission delay, and packet loss characteristics. This Chapter is organized as follows. An overview of the proposed approach is presented in Section 6.2. Related work in the area of loss and delay differentiation for video communication is discussed in Section 6.3. Our proposed technique are discussed in Sections 6.4 and 6.5, respectively. Specific packet classification schemes for a loss and delay differentiated network are discussed in Section 6.6. Simulation results of transmitting a DVD movie over a loss and delay differentiated network are presented in Section 6.7. Finally, conclusions are given in Section 6.8.

6.2 Tagging MPEG for DiffServ Networks

A DiffServ network typically provides multiple traffic classes with different QoS in terms of loss and delay. One possible approach to exploit such QoS diversity is to transcode the MPEG video into alternative formats that can be easily mapped to the different QoS parameters. For instance, transcoding MPEG video into a layered representation [47, 4] would allow natural mapping of layers to traffic classes with increasing loss rates. In particular, the ability of some scalable compression techniques to produce more than two layers [134, 103, 129] further increases the level of achievable traffic differentiation. However, this transcoding approach suffers from three drawbacks. The first drawback is the high complexity associated with the transcoding operation, which in many situations needs to be performed on a real-time basis. The second drawback is the inherent video quality degradation associated with transcoding. Finally, the third drawback is the additional latency introduced by transcoding.

In this Chapter, we seek a transcoding-free approach to sending video using multiple traffic classes in a DiffServ network. Our approach can be summarized as follows. An MPEG bit-stream is first split into sub-streams with different delay and loss sensitivities. Different methods of computing such sensitivities are discussed later in Sections 6.4 and 6.5. In such a way, a map of loss-delay spread is obtained showing the amount of video data with each value of delay and loss sensitivity, as illustrated in Fig. 6.2-(a). This map is a property of the video only and can be computed independently of the traffic classes prior to transmission. Since the map orders different parts of an MPEG video according to delay and loss sensitivities, it can be used for packet classification and tagging in delay and loss differentiated networks. For example, Fig. 6.2-(b) shows the classification of the MPEG video bit-stream into two traffic classes with different loss priorities.

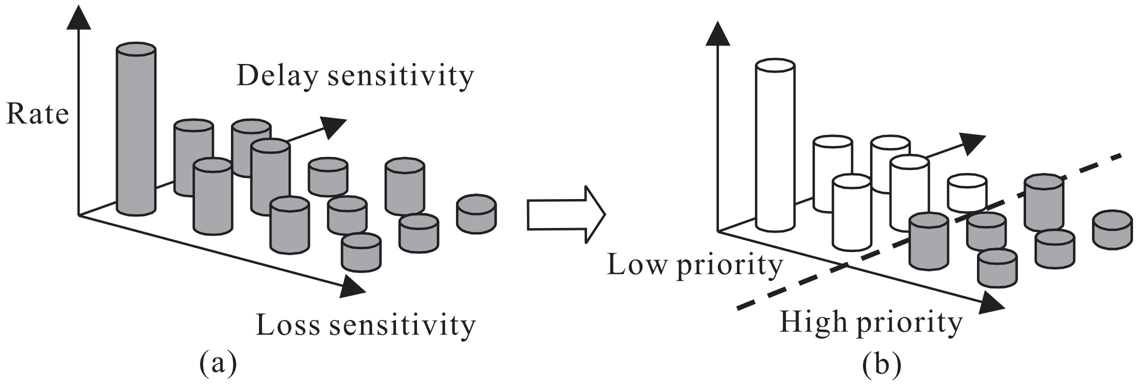


Figure 6.2: Using a map of delay-loss spread to classify traffic into low-loss and high-loss classes.

More generally, suppose a video server streaming MPEG video over a DiffServ network has at its disposal N traffic classes C_1 to C_N. Each of the classes C_i will have different QoS parameters typically expressed in terms of delay and loss, and an associated maximum bandwidth R_i. Suppose further that the network offers differentiation in loss only with increasing loss rate ranging from C_1 to C_N. Thus, class C_i is rate limited to R_i. Then, a packet tagging scheme that attempts to minimize the effect of packet loss would proceed by first classifying the most loss-sensitive data to class C_1 until the available bandwidth R_1 is reached. It will then continue traffic classification by tagging and assigning the most loss-sensitive part of the remaining data to class C_2 until the available bandwidth R_2 is reached. The process will continue until the entire bit-stream is classified. In such a way, we are guaranteed that the more loss-sensitive part of a video bit-stream is transmitted using a traffic class with lower loss. For networks with delay differentiation only, tagging proceeds in a completely analogous manner.

For networks that offer differentiation in both delay and loss, the packet tagging algorithm is more challenging. Generally, if the available rates R_i for different traffic classes are given, then a tagging scheme that simultaneously optimizes loss and delay may not exist. It is possible to construct an aggregate cost as a function of loss and delay so that the different traffic classes can be ordered in terms of QoS, and the delay-loss spread map can be ordered in terms of importance. While such an approach would allow simple mapping of data to different traffic classes, the construction of such a cost function requires a trade-off between delay and loss. Since delay and loss give rise to different types of degradations in visual communication, we focus instead on schemes that either optimize for loss or delay. In Section 6.6.1, we will describe specific packet tagging schemes that optimize for either loss or delay for the case when the available bandwidth R_i for each traffic class is fixed.

Rather than assuming a pre-specified amount of bandwidth available to each traffic class, an alternative and less restrictive formulation is possible by assuming a pricing structure for the different traffic classes. The constraint then can be given by a fixed total budget to transmit the entire video. This alternative formulation allows maximization of reception quality for a given cost, as will be shown later in Section 6.6.2.

6.3 Related Work

The idea of providing differential treatment to different parts of compressed digital video with different error sensitivity is not new. In particular, it has been shown in [47, 28, 45] that by separating the source material into a base layer and an enhancement layer, and by applying priority queuing [28, 45] or different loss probabilities [47], results superior to the one layer approach can be obtained for statistically multiplexed networks. In terms of MPEG, it has also been shown that two-layer scalable MPEG outperforms single-rate MPEG with loss prioritization [4, 148]. However, in this Chapter, we only consider the transmission of non-scalable MPEG video due to the abundance of existing content.

For existing content compressed using one-layer MPEG, different priorities can be assigned according to the different frame types. For example, [73] uses three different priorities for the I, P, and B frames of an MPEG video to provide unequal error protection. In [69], the performance of two priority assignment schemes are compared to each other for MPEG video multiplexed in a fixed network buffer. The two schemes are: (1) marking I, P as important and B as less important, and (2) marking I as important and P, B as less important.

Another approach to traffic prioritization for MPEG video is to assign higher priority to the motion information than the texture information. For instance, [54] has applied unequal error protection to protect motion information for transmission of MPEG-4 video over wireless channels.

There is considerably less work in the area with separating a video into parts with different delay sensitivities. One notable exception is the delay-cognizant video compression of [17], where the generation of video sub-streams of different delay sensitivities is enabled by the use of asynchronous frame update, i.e., by rendering different parts of a given source video frame at different times according to perceived delay sensitivity. Unfortunately, the techniques of [17] cannot be easily applied to MPEG video, where each video frame needs to be rendered at fixed given times. In terms of MPEG, a scheme is suggested in [111] that reduces variation between displayed frames when the decoder is incapable of decoding at full frame rate. Thus, instead of decoding and displaying frames 1, 9, 10, the video frames 1, 5, 10 will be decoded and displayed instead.

6.4 Delay Separation for MPEG Video

While MPEG compression does not make special provisions for transport over delay-differentiated networks, limited delay separation is achievable by exploiting the difference between the order in which video frames are stored in the bit-stream, and the order in which they are displayed. We shall refer to the order in which compressed video frames are stored in the bit-stream as the *bit-stream order*, and the order in which frames are captured at the source, or rendered and displayed at the sink as the *display order*.

Fig. 6.3 shows a typical frame pattern commonly used in the production of DVD

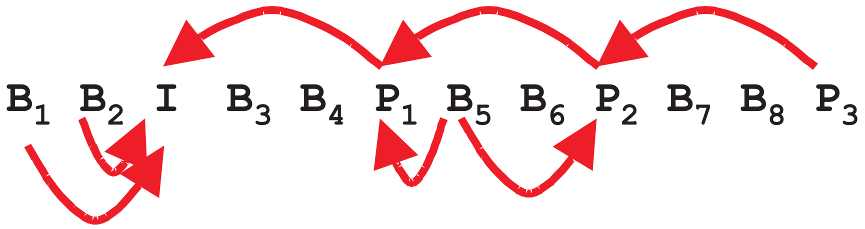


Figure 6.3: Dependence between different frame types in an MPEG video.

movies. The frames are shown in their display order, with the arrows indicating the dependence between them. In general, Intra or I frames are coded by themselves while Predicted or P frames depend only on the previous P or I frames. Bi-directionally predicted or B frames may depend on the previous non-B frame, the next non-B frame, or both [89]. As shown in Fig. 6.3, since both B_1 and B_2 depend on I, the bit-stream is organized so that the bits for I are available before those of B_1 and B_2 despite the fact that I is displayed after B_1 and B_2. Similarly, bits for P_1 appear in the bit-stream before those of B_3 and B_4 for the same reason. This difference between the bit-stream and display order of MPEG video means that certain frames can afford to have larger transmission delays than others. Fig. 6.4 shows the case where compressed video frames are transmitted in bit-stream order at periodic intervals, with arrows mapping frames in bit-stream order to the instants they are first needed in display order. We see that generally, I and P frames can tolerate extra delay corresponding to one video frame as compared to B frames. For a full frame rate movie at 30 fps, this corresponds to a 33 ms difference between the delay requirements of B frames and non-B frames. While such delay spread is small compared to the duration of a

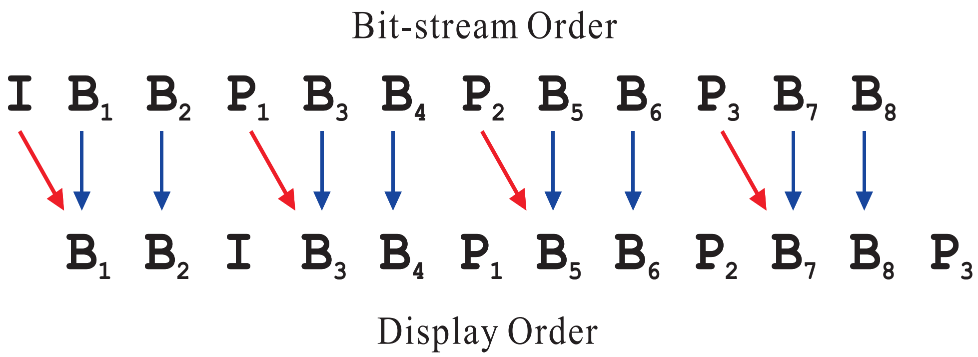


Figure 6.4: Exploiting the difference between display and bit-stream order for an MPEG-compressed video.

movie, it is sizable in the timescale of network delays.

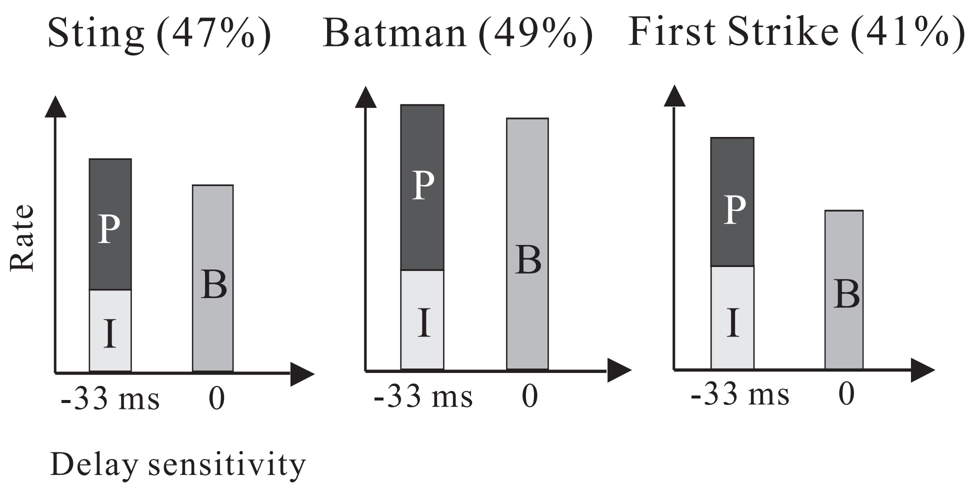


Figure 6.5: Delay spread for three DVD movies. Percentage of bytes in B frames are shown in brackets.

Fig. 6.5 shows the delay spread for three DVD movies with increasing motion content: *Sting*, *Batman*, and *First Strike*. The delay sensitivity of the B frames is normalized to be zero while the delay sensitivities of the I and P frames are -33 ms, indicating that both frame types can tolerate an additional 33 ms of delay. We see that B frames account for between 40 to 50% of the total bits, yielding roughly equal amount of data in the more delay-sensitive and the less delay-sensitive categories.

The delay spread of Fig. 6.5 is obtained using only the frame-level information of bit-stream and display order. As such, implementation complexity is extremely low since both the bit-stream and display order are available in the MPEG picture headers, and can be read off directly. The delay spread can be improved by exploiting the fact that each MPEG frame is typically organized into multiple *slices*, and each slice could potentially have different delay tolerances. For example, the frame size for DVD movies fitted for the TV screen is 720×480. Each video frame is typically

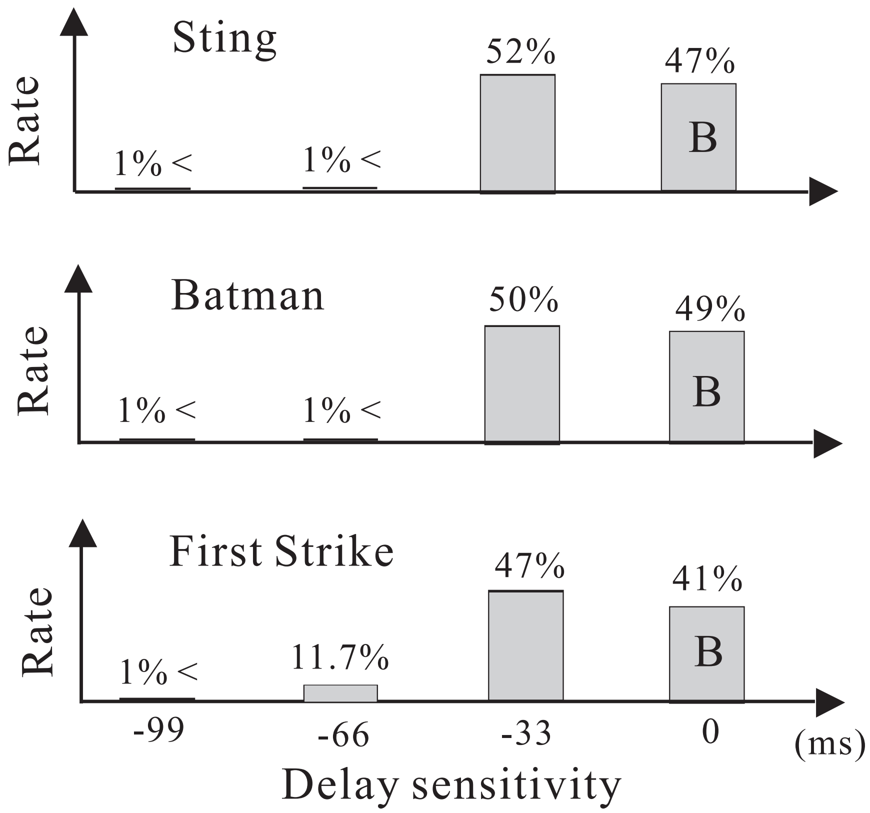


Figure 6.6: Delay spread using motion vector information.

composed of 30 slices, each corresponding to a horizontal strip of size 720×16. By partially decoding the MPEG bit-stream to obtain the motion vectors, it is possible to determine exactly when each slice in an I or P frame is needed. For example, if B_3 does not depend on one particular slice in P_1 in Fig. 6.4, that slice may not be needed until B_4 is decoded, resulting in a delay tolerance of two frames over the B frames. Furthermore, if B_4 does not depend on the same slice, it can gain yet another one frame worth of delay tolerance.

The resulting delay spreads obtained using motion vector information for the 3 DVD titles are shown in Fig. 6.6. We see that only minor additional spread is obtained as compared to the delay spread of Fig. 6.5 for the three DVD movies tested. This is probably due to the fact that at high frame rates such as 30 fps for DVD movies, successive video frames are well correlated so that almost all slices in I and P frames are used as reference for motion prediction. The largest improvement comes from the action movie *First Strike* which by virtue of its high motion content, tends to have larger portions of non-referred slices in the I and P frames.

The delay spread achieved through the above described technique can potentially reduce network resources to transmit video. Specifically, consider two traffic classes, one with a small delay guarantee of Δ_L seconds, the other with a larger delay guarantee of Δ_H seconds, where $\Delta_L < \Delta_H$. Assume $\Delta_H - \Delta_L \leq 33\ ms$. Then, no additional delay is incurred by sending the B frames in the low-delay class and the other frames in the high-delay class, as compared to transmitting all frames in the low-delay class. This is because I and P frames can tolerate an additional delay of at least 33 ms. Since B frames account for 40 to 50% of the bits in the 3 DVD movies used in Fig. 6.6, more than half of the traffic can be transmitted in the high-delay class without incurring any penalty. Thus, the delay decomposition technique described in this section reduces the network resources required to transmit video over networks that can process packets in a differentiated manner.

6.5 Loss Separation for MPEG Video

In this section, we will propose and compare several methods for decomposing the bit-streams for DVD movies based on their loss sensitivity characteristics. There are a number of possible mechanisms for such decompositions in the MPEG context. One possible approach is to assign higher priority to motion information in favor of texture information [54]. Another approach is to assign highest priority to I frames, followed by P frames and B frames respectively [73, 69]. The latter approach can be generalized using the notion of *reference count*, in which each frame or slice is assigned importance according to how many frames or slices are dependent on it. In this section, we will compare five schemes for loss differentiation summarized as follows:

1. *Uniform*: Treats an MPEG bit-stream as homogeneous and assigns equal importance to all bits. This scheme serves as a base-line for our comparisons.

2. *Frame Level Reference*: The importance of all slices in a specific frame equals the number of frames that refer to it. For example, every B frame has a reference count of one since no other frames besides itself refers to it. On the other hand, the I frame in Fig. 6.5 has a reference count of 12 since it is used to predict every frame in the given GOP. Similarly, P_1 and P_2 have reference counts of 9 and 6 respectively.

3. *Slice Level Reference*: In this scheme, if a macroblock in frame i is predicted from K macroblocks in a previously coded frame j, each of the K macroblocks in frame j will receive a contribution of $1/K$ to their reference count. The importance of each slice is then computed as the average reference count of its macroblocks.

4. *Motion*: Assigns more importance to motion information than texture.

5. *Motion + Slice Level Reference*: Motion vector is assigned highest importance. The importance of texture information is ordered by the method of Slice Level Reference.

To compare the performance of the above schemes, we simulate the transmission of the DVD movie *First Strike* using a simple loss-differentiated network with two traffic classes of 0.3% and 3% packet loss. A two-state Markov process with parameters given in the Appendix is used to generate packet losses. The setup is illustrated in Fig. 6.7 where the *packet marker* is responsible for marking the packets so that the percentage

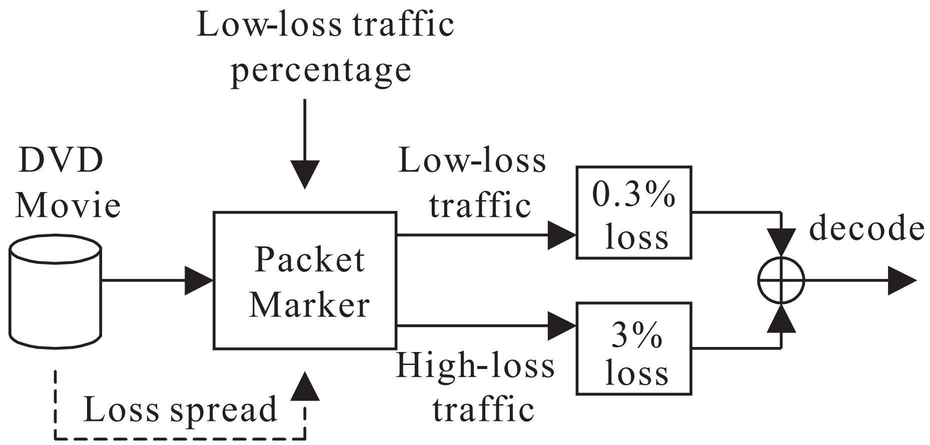


Figure 6.7: A simple two-class loss-differentiated network.

of low-loss traffic remains at a pre-specified level. In two independent simulations, we limit the ratio of low-loss traffic to be 50% and 30% of the total traffic respectively. Fig. 6.8 shows the spread of loss sensitivity for *First Strike* using the five schemes. The dashed and dotted lines show the boundary separating low and high loss traffic when the available bandwidth in the low-loss traffic class is 50% and 30% of the total bit-rate respectively. No unit or values are given for the loss sensitivities in Figs. 6.8-(a) to (e) due to the fact that (1) the interpretation of loss sensitivity is different for different schemes and hence cannot be directly compared against each other, and (2) we are only interested in the amount of separation or spread that is achievable within each scheme, and not the actual value of the loss sensitivity. As expected, *motion + slice level reference* results in the larger spread than any of the other schemes.

Figs. 6.9 and 6.10 show the simulation results of applying the five loss differentiation schemes for 50% and 30% low-loss traffic respectively. The distortion measured

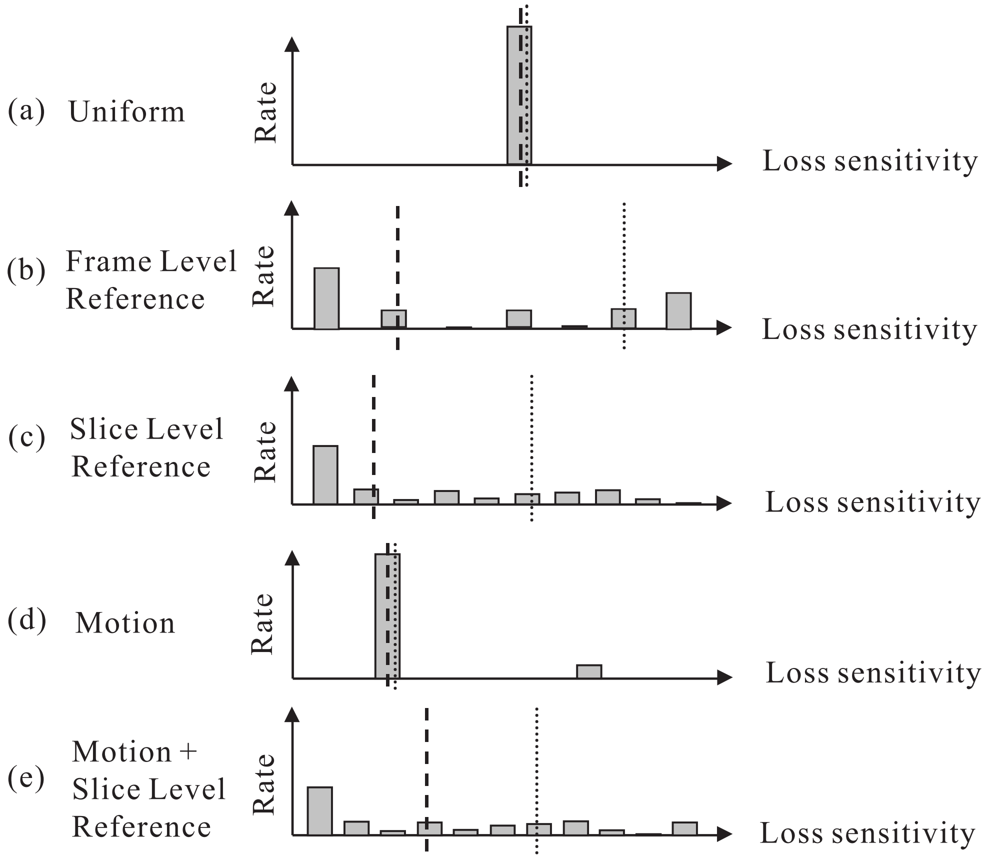


Figure 6.8: Loss spread for DVD movie *First Strike* using different methods of loss differentiation. The dashed and dotted lines show the boundary separating low and high loss traffic when the available bandwidth in the low-loss traffic class is 50% and 30% of the total bit-rate respectively.

in mean square error (MSE) as a function of time is shown for the movie *First Strike* where the loss rate for the low-loss and high-loss traffic classes are 0.3% and 3% respectively. The MSE induced by packet loss is computed for every video frame, and an average number is reported for every minute or 1800 frames. In the case when a slice is missing due to packet loss, it is approximated either by the corresponding slice in the previous reconstructed frame, or the previous slice in the current frame, depending on which is estimated to be more similar. The main conclusion from the

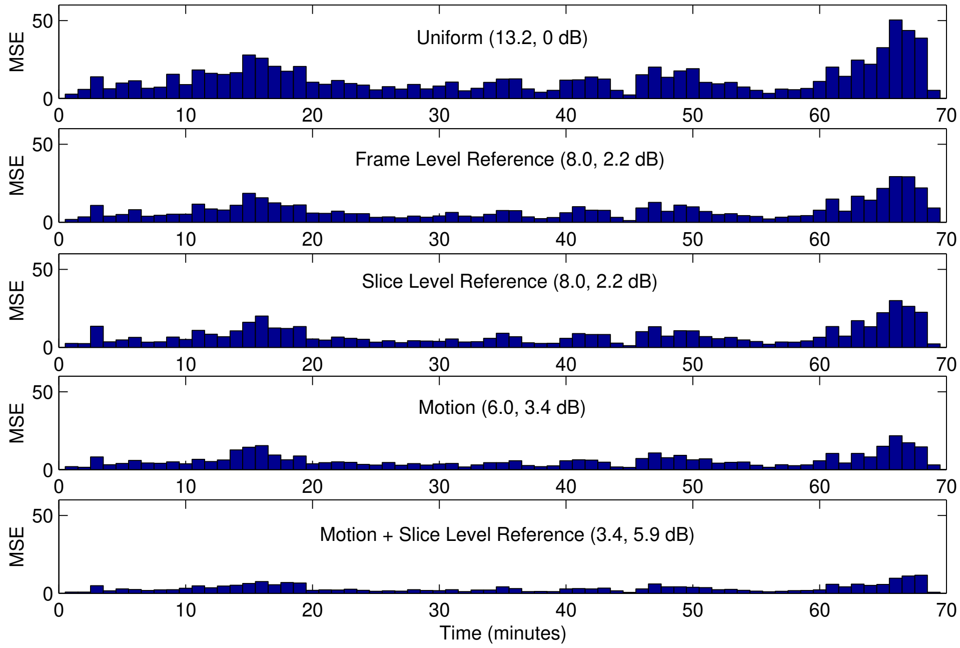


Figure 6.9: Comparison of loss differentiation schemes using 50% low-loss traffic. Average MSE, and distortion reduction in dB are shown for each scheme in brackets.

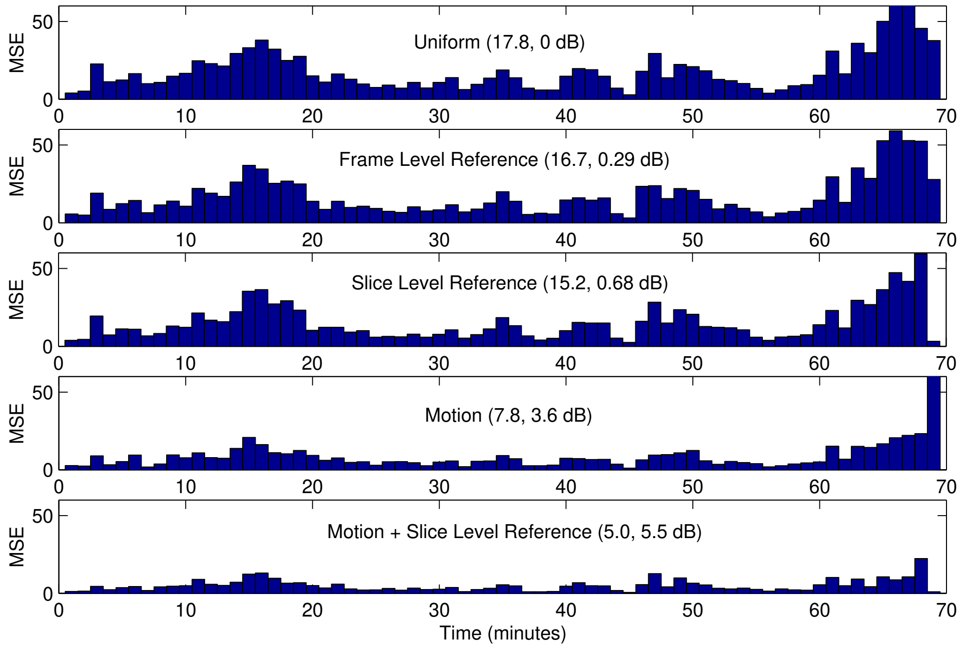


Figure 6.10: Comparison of loss differentiation schemes using 30% low-loss traffic. Average MSE, and distortion reduction in dB are shown for each scheme in brackets.

MSE traces in Figs. 6.9 and 6.10 is that the *uniform* scheme suffers from the highest distortion whereas *motion+slice level reference* provides the lowest distortion in both cases. This is because the former offers no loss differentiation while the latter offers the most loss differentiation, as shown in Fig. 6.8. There are two other observations we can make from Figs. 6.9 and 6.10. First, the *slice level reference* scheme consistently outperforms the *frame level reference* scheme due to its larger degree of loss differentiation at the expense of larger complexity. Second, it is found that loss differentiation based on motion and texture information is more effective than those based purely on reference counts. Nevertheless, the lowest distortion is achieved by combining both loss-differentiation methods, as in the *motion + slice level reference* scheme.

To verify the above observations, we repeated the simulated transmission of the *First Strike* movie at different combinations of packet loss rates for the low-loss and high-loss traffic classes. The percentage of low-loss traffic is maintained at 30% in all cases, and the results are shown in Table 6.1. We see that the *motion + slice-level reference* scheme still performs the best, followed by the *motion*, *slice-level reference*, *frame-level reference* and *uniform* schemes respectively.

Fig. 6.11 shows the distortion of sending the entire *First Strike* movie using the 0.3% loss traffic class only. The average distortion in this case is 2.5, compared to 3.4 in the *motion + slice level reference* scheme of Fig. 6.9. Thus, by sending all traffic using the premium class, a distortion reduction of 1.3 dB is achieved at

Loss rates of low-loss / high-loss traffic classes in percentage	0.3%/3%	0.3%/5%	1%/3%
uniform	17.8	28.9	19.0
frame-level ref	16.7	26.3	18.5
slice-level ref	15.2	24.2	17.5
motion	7.8	11.1	12.1
motion + slice-level ref	5.0	6.9	10.15

Table 6.1: MSE for different loss differentiation schemes for different loss rates for the low-loss and high-loss traffic classes. 30% low-loss traffic is used.

the cost of consuming twice as much bandwidth of the "premium" low-loss traffic. The distortion reduction is expected to become smaller as the difference in loss rate between the low-loss and high-loss classes becomes smaller.

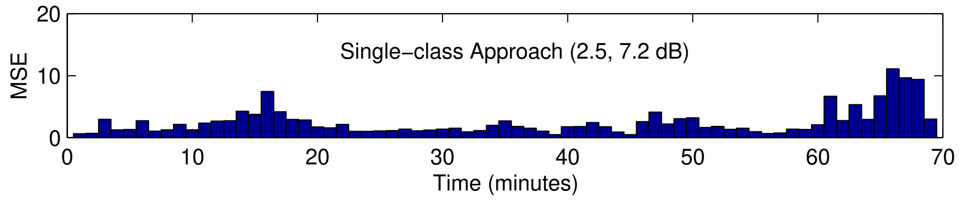


Figure 6.11: Distortion when 0.3% loss is applied to DVD movie *First Strike*. Average MSE, and distortion reduction in dB compared to the *uniform* scheme of Fig. 6.9 are shown in brackets.

6.6 Packet Classifiers Optimized for Loss and Delay

In this Section, we will describe packet classifiers for the DiffServ network with four traffic classes with loss rates of p_L and p_H, and delays of d_L and d_H, where

$p_L < p_H$ and $d_L < d_H$, as depicted in Fig. 6.12. We consider two models of bandwidth provisioning. In the *fixed-bandwidth* model, the available bandwidth of each traffic class is given. As an example, we will denote the available bandwidth of the low-loss, high-delay class by R_{LH}. In the *fixed-price* model, a pricing structure for the traffic classes is given and the user is free to choose the amount of bandwidth he or she wishes to consume for each class subject to a fixed total budget. As an example, the price per unit volume for the low-loss, high-delay class is denoted by π_{LH}.

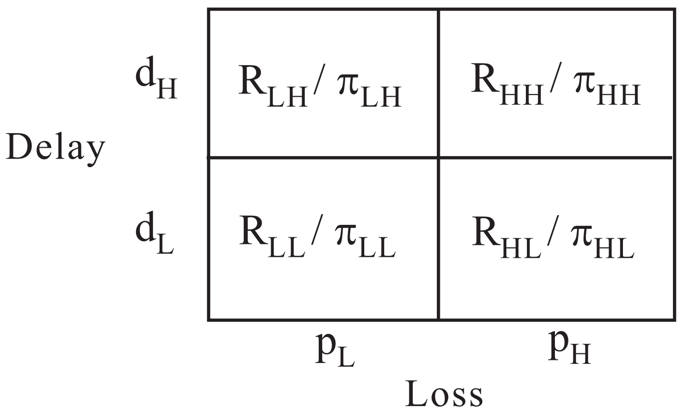


Figure 6.12: Four traffic classes that differ in loss rate ($p_L < p_H$) and delay ($d_L < d_H$). The class with low loss and high delay can be specified to have an available bandwidth of R_{LH} or a price of π_{LH} per unit volume.

6.6.1 The Fixed-bandwidth Model

When the available bandwidth for each traffic class is given, it may not be possible to simultaneously optimize for both delay and loss. In that case, a *loss-optimized* packet classifier assigns the more loss-sensitive data to the low-loss traffic classes first. As shown in Fig. 6.13-(a), this is achieved by first assigning the most loss-sensitive

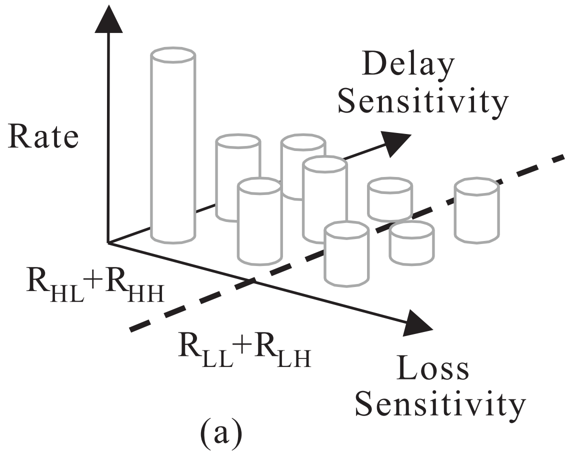


Figure 6.13: Loss-optimized classification of into four traffic classes with delay and loss differentiation.

data to the low-loss traffic classes until a rate of $R_{LL} + R_{LH}$ is reached. Then, each of the low-loss and high-loss parts is further split according to delay sensitivity, as illustrated in Fig. 6.13-(b). In such a way, it is guaranteed that the more loss-sensitive data is transmitted using a lower loss traffic class.

A *delay-optimized* packet classifier can be constructed in an analogous fashion,

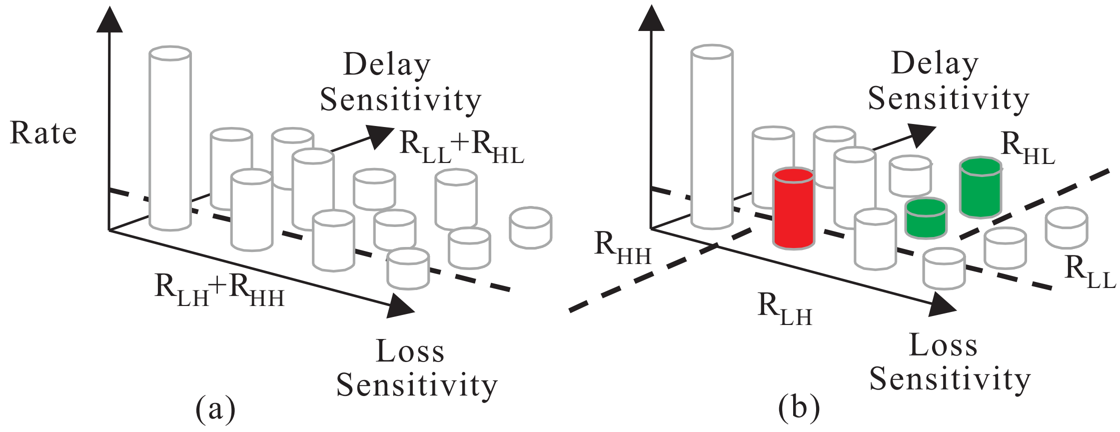


Figure 6.14: Delay-optimized classification of into four traffic classes with delay and loss differentiation.

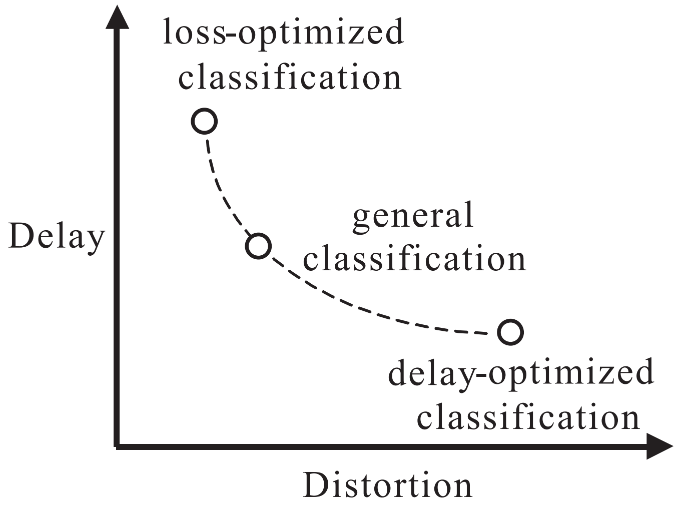


Figure 6.15: Performance of different packet classification scheme in terms of distortion and delay.

and is illustrated in Fig. 6.14. Comparing Figs. 6.13-(b) and 6.14-(b), we observe that the loss-optimized and delay-optimized packet classifiers are generally different. In fact, the *loss-optimized* and *delay-optimized* packet classification schemes achieve minimum distortion and minimum delay respectively. More general packetization schemes achieving intermediate distortion and delay are also possible, as shown in Fig. 6.15. For example, consider packet classifiers that minimize a linear function of delay and loss:

$$cost = (1 - \rho) \times loss\ sensitivity + \rho \times delay\ sensitivity.$$

By varying the relative cost ρ from 0 to 1, a family of packet classifiers that begin with the *loss-optimized* classifier and end with the *delay-optimized* classifier is obtained, as shown by the dashed curve in Fig. 6.15[1]. Using other cost functions will

[1]Note that the dashed curve in Fig. 6.15 is not generated using any experimental or simulation results. Rather, it is drawn purely for illustrative purposes.

result in different curves. The problem of characterizing the achievable region in the distortion-delay plane in Fig. 6.15 is open and as such is a potential subject for further investigation.

6.6.2 The Fixed-price Model

One consequence of the more restrictive fixed-bandwidth model is that network resources may not be most efficiently used. For example, in the delay-optimized scheme of Fig. 6.14-(b), more loss-sensitive data, such as those with green color, may be transmitted in the high loss class while less loss-sensitive data, such as those with red color, is transmitted in the low loss class. In the fixed-price model, one can eliminate such inefficiencies by using a special class of packet classifiers that guarantees loss sensitive data in low-loss traffic class and delay sensitive data in low-delay class. Fig. 6.16 shows two such packet classifiers with different amount of bandwidth usage in different traffic classes.

We perform simulations on DVD movie *First Strike* to evaluate the delay and distortion of several classifiers with different bandwidth usage in the different traffic classes. In the simulations, the delay in each traffic class is generated using a refined M/M/S model[2] to take into account the finite buffer size of practical routers [64]. It has been shown empirically in [64] that the model can match the actual measured delay distribution relatively accurately. In particular, the model parameters for a

[2]The first "M" indicates traffic arrival is memoryless. The second "M" indicates service time is memoryless. "S" is an integer representing the number of servers or output links.

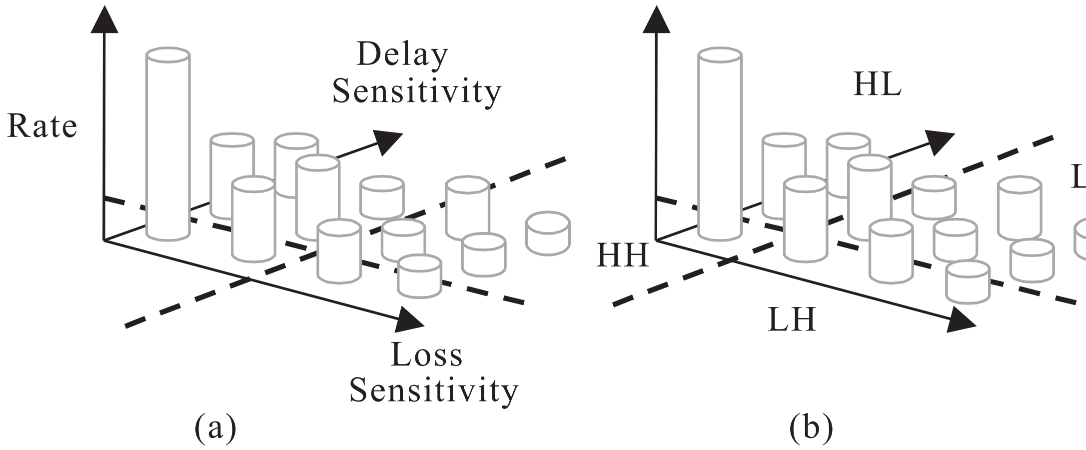


Figure 6.16: Packet classifiers for the fixed-price model.

number of actual sessions of different delays are reported in [64]. In simulating delay, we use the reported parameters for both the low-delay and high-delay traffic classes. The maximum queue length is 10 packets, and S is 2 for all cases. However, the delay for the low-delay classes is generated using a lower traffic intensity of 4 units as compared to 6 units for the high-delay classes. Packet losses are generated by a two-state Markov process described in the Appendix. The loss rate for the low-loss and high-loss classes are 0.3 and 3 percent respectively. The results for different distribution of data into the four traffic classes are shown in Table 6.2. Distortion in MSE and the minimum end-to-end delay needed in order for 95% of the video frames to be on time are reported. As expected, lower distortion is achieved as the total low-loss traffic, $R_{LL} + R_{LH}$, increases. Similarly, lower delay is experienced as the total low-delay traffic, $R_{LL} + R_{HL}$, increases from 0% to 40%. Even though the delay values are simulated using a stochastic model, the end-to-end delay needed for 95% of video frames to be on time remains the same for different simulations corresponding to fixed volume of low-delay traffic. This can be attributed to law of large numbers and long simulation times, an entire DVD movie in our case.

Label	R_{LL}	R_{LH}	R_{HL}	R_{HH}	Distortion (MSE)	Delay for 95% on time (ms)
A	0.07	0.58	0.33	0.01	2.9	58
B	0.07	0.48	0.33	0.11	3.3	58
C	0.07	0.36	0.33	0.23	4.2	58
D	0.07	0.25	0.33	0.34	6.5	58
E	0.07	0.07	0.33	0.52	8.8	58
F	0	0.55	0	0.45	3.3	90
G	0	0.44	0	0.56	4.2	90
H	0	0.32	0	0.68	6.5	90
I	0	0.14	0	0.86	8.8	90

Table 6.2: Achievable distortion and delay by assigning different percentage of traffic to different traffic classes.

Assuming π_{LL}, π_{LH}, π_{HL}, and π_{HH} are 3, 2, 2, and 1 respectively, the total cost needed to achieve different values of distortion and delay are shown in Figs. 6.17-(a) and (b) respectively. Given the requirements of an application in terms of loss and delay, Figs. 6.17-(a) and (b) allows the computation of the minimum transmission cost. For example, an application requiring lowest possible delay and a maximum MSE of 4 will choose the packet tagging scheme corresponding to B with an associated cost of 1.9. Conversely, given a total cost constraint, Figs. 6.17-(a) and (b) show the achievable values of MSE and delay respectively.

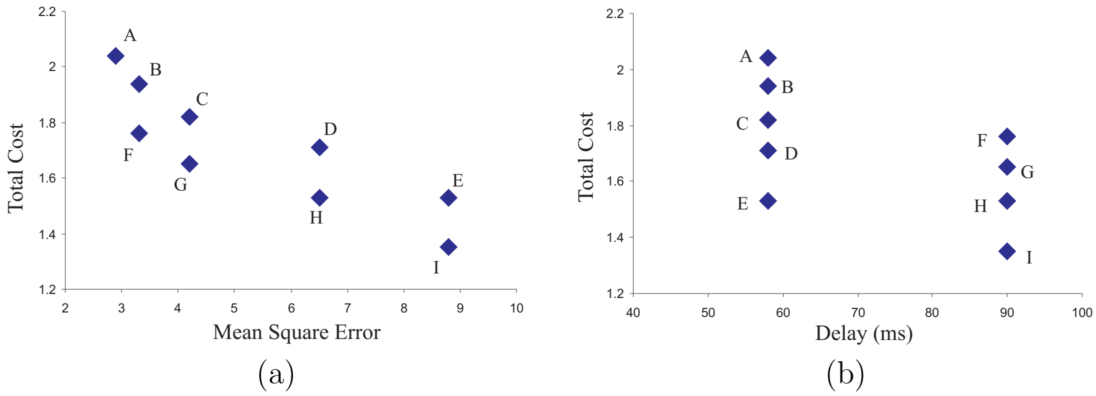


Figure 6.17: The cost needed to support different values of (a) distortion and (b) delay for various packet classifiers A to I.

6.7 Simulations using DiffServ Networks

Even though DiffServ is still under development in IETF, two distinct approaches of providing traffic differentiation have evolved. The first approach is the *expedited service*, which strives to provide absolute performance guarantees via resource reservations but without the per-flow states as required by RSVP [62]. The second approach is the *assured service*, which aims at providing relative service differentiation [53]. Since each DiffServ domain can independently choose its own service disciplines, the DiffServ Internet is likely to consist of a mix of networks providing expedited, assured, and best-effort services. As such, the most general assumption about QoS on an end-to-end basis is relative differentiation.

In this section, we simulate a DiffServ network that provides relative differentiation. There are many possible implementations of relative differentiation, e.g., those that use the random early drop (RED) routers with In/Out bit or *RIO* [24]. In our experiments, we have chosen a special form of assured service known as the Proportional Differentiation Model [38] due to its ability to control the relative QoS in different traffic classes.

6.7.1 Overview of Proportional Differentiation Model

Suppose $q_i(t, t+\tau)$ is the observed QoS, such as packet loss rate or queuing delay, for traffic class i during the time interval $(t, t+\tau)$, where $\tau > 0$ is the duration of QoS observation period. The proportional differentiation model maintains that for any two traffic classes i and j:

$$\frac{q_i(t, t+\tau)}{q_j(t, t+\tau)} = \frac{c_i}{c_j}$$

where c_i and c_j are specified quality parameters.

Mechanisms for realizing the proportional differentiation model for both queuing delay and loss rate are given in [38] and can be summarized as follows. To achieve loss differentiation, when a queue is full and a new packet arrives, one packet is dropped from the class k with the lowest normalized loss rate where:

$$k = \arg\min_i \frac{l_i(t, t+\tau)}{\lambda_i} \tag{6.1}$$

and l_i and λ_i are the observed loss rate and loss quality parameter for class i. To achieve delay differentiation, the packet p^* with the largest normalized waiting time in the set of all queued packets Q is chosen for transmission:

$$p^* = \arg\max_{p \in Q} \frac{w(p)}{\delta(p)} \tag{6.2}$$

where $w(p)$ is the time packet p has spent in the queue, and $\delta(p)$ is the delay quality parameter for the traffic class p belongs to.

6.7.2 Simulation Setup

In this section, we describe our simulations on performance improvement achieved by the *delay-optimized* and *loss-optimized* packet classification schemes over the basic *uniform* scheme described in Section 6.5. We do not consider packet classifiers under the fixed-price model of Section 6.6.2 due to the lack of realistic pricing information in a DiffServ framework at the present time.

The simulations are performed using the setup shown in Fig. 6.18, where there is a bottleneck link of 17 $Mbps$, with the routers shown in shaded circles implementing the proportional differentiation model described in Section 6.7.1. Four traffic classes with both loss and delay differentiations are provided. The quality parameters of the four classes are given by (λ, δ) = (1,1), (1,2), (10,1) and (10,2) respectively. According to Equation 6.1, the traffic classes with $\lambda = 1$ will statistically have one-tenth the loss rate of traffic classes with $\lambda = 10$ during periods of network overload. Similarly, Equation 6.2 maintains that traffic classes with $\delta = 1$ will statistically have half the queuing delay as traffic classes with $\delta = 2$ when backlog is present. Thus the set of traffic classes assume the form given in Fig. 6.12.

Three sources are used, with S_1 and S_2 streaming the movie *Sting*, and S_3 streaming the movie *First Strike* to receivers R_1, R_2 and R_3 respectively. Senders S_1 and S_2

start streaming 5 and 1 minutes before that of S_3 respectively. In three independent runs of the experiment, the *uniform* scheme of Section 6.5, the *delay-optimized* and the *loss-optimized* schemes of Section 6.6.1 are employed in S_3 with cross-traffic provided by S_1 and S_2, both employing the uniform scheme. In all cases, loss-differentiation according to *motion + slice level reference* and the simpler delay-differentiation according to B frames and non-B frames are used.

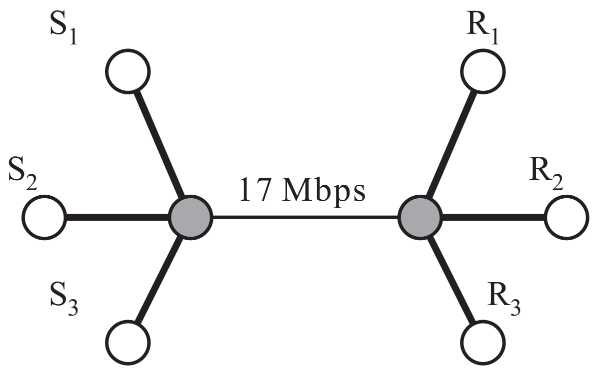


Figure 6.18: Simulation setup using 3 instances of the same movie. The sources and receivers are S_i and R_i respectively, where i denotes different methods of traffic classification. DiffServ enabled routers are shaded.

In each run of our simulation, each source is given an available bandwidth equal to the average bit rate of the movie, i.e., 6.2 *Mbps* for *First Strike* and 4.6 *Mbps* for *Sting*. The available bandwidth for each source is distributed equally in the four traffic classes. Thus the average total traffic in the bottleneck link is 15.4 *Mbps* for the three streams combined, which is lower than the bottleneck capacity of 17 *Mbps*. However, due to the time-varying nature of the video sources, congestion is still experienced at various times.

One compressed video frame at bit-stream order is transmitted every 1/30 second.

Thus, the amount of data sent over the network by each source is time-varying, and may differ depending on whether it is an I, P or B frame, and the complexity of the video frame being sent. Nevertheless, the same amount of instantaneous traffic is generated in all three runs of the experiment. The video packets are tagged at the sources into one of the four classes according to the respective packet classification algorithm employed. Packets from different sources and with the same tag are given the same treatment at the bottleneck link. One hour of video transmission is simulated and the results in terms of loss and delay are reported in Section 6.7.3

6.7.3 Simulation Results

For each run of the experiment, even though S_3 uses the same amount of bandwidth in each of the traffic classes, due to the employment of different packet classification schemes, i.e., delay-optimized, loss-optimized, and uniform, the resulting distortion and delay are different. Fig. 6.19 shows the MSE trace for the *First Strike* movie streamed by S_3 using the three packet classification schemes. In all three cases, an identical bandwidth of one-fourth the average rate of *First Strike* or 1.55 $Mbps$ is used in each of the four traffic classes. The average MSE are 2.15, 1.6 and 0.76 for the uniform, delay-optimized and loss-optimized schemes respectively. Thus, the reduction of distortion for the *delay-optimized* and *loss-optimized* schemes over the *uniform* scheme are 1.3 dB and 4.5 dB respectively. As expected, the *loss-optimized* scheme achieves lowest distortion due to strategic classification of more important data to low-loss traffic classes.

We next look at the delay performance of the three schemes in terms of a metric we call *variable latency*. Denote by T the time between successive video frames at the source. Due to the difference in the display order and bit-stream order of an MPEG-compressed video, the *allowable delay*, or the time difference between the transmission of a packet p and the time instant at which p is needed at the decoder, may vary from packet to packet, as illustrated in Fig. 6.20. Consider a packet p belonging to the I frame in Fig. 6.20. Denote by τ_A the time interval between when packet p is transmitted and the time it is needed at the decoder. If p is not needed by B_1, but is needed by B_2, then the allowable delay is $\tau_A + T$. Finally, if p is needed by neither B_1 nor B_2, then the allowable delay is $\tau_A + 2T$. Generally, if K B-frames are used, the allowable delay is of the form $\tau_A + \kappa^{(p)}T$, where $\kappa^{(p)}$ is an integer between 0 and

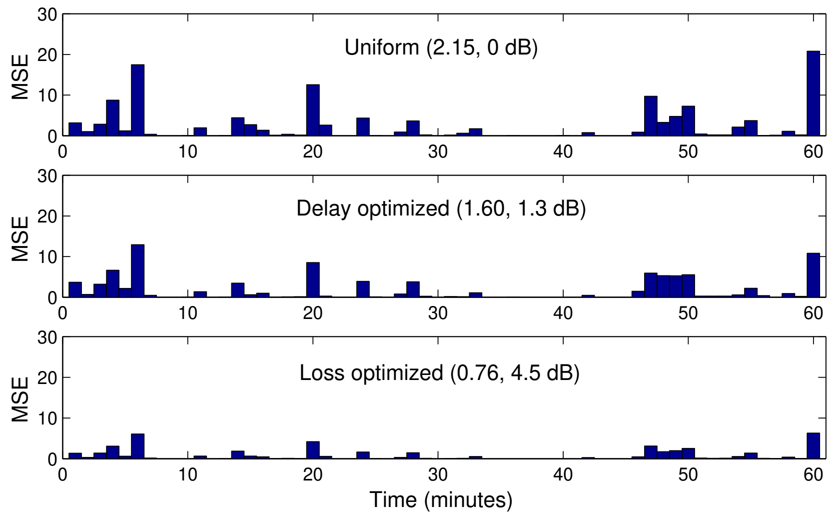


Figure 6.19: Distortion traces for three packet tagging schemes streaming a DVD movie over a DiffServ network.

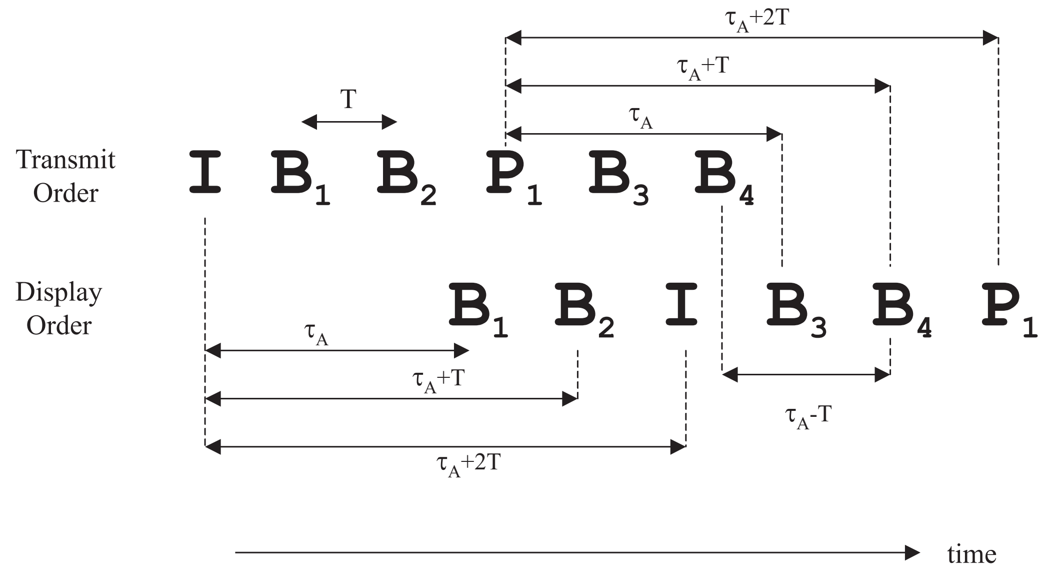


Figure 6.20: Timeline for the transmission of packets of different frame types.

K inclusively. A completely analogous result holds for packets belong to P frames in that the allowable delay between transmission time of a packet p and the time at which p is needed for decoding other P or B frames is also of the form $\tau_A + \kappa^{(p)}T$. For packets belonging to B frames however, the difference in the display and bit-stream order dictates that they have a smaller allowable delay of $\tau_A - T$. Since actual delay of a packet p can be considered to be the sum of a constant propagation delay τ_p and a variable queuing delay $\tau_q^{(p)}$, p is considered to be on time if the allowable delay is no less than the actual delay, or:

$$\text{Packet } p \text{ on-time if } \begin{cases} \tau_A + \kappa^{(p)}T \geq \tau_p + \tau_q^{(p)} & \text{if } p \text{ belongs to an } I \text{ or } P \text{ frame,} \\ \tau_A - T \geq \tau_p + \tau_q^{(p)} & \text{if } p \text{ belongs to a } B \text{ frame.} \end{cases} \tag{6.3}$$

When the simple delay differentiation scheme of differentiating between B and non-B frames is used, as in the case of our simulations, the allowable delay for I and P frames can be simplified to τ_A. In the example of Fig. 6.20, the simple delay differentiation scheme conservatively assumes that every packet in the I frame might be needed to decode B_1 and B_2, and all the packets in both the I frame and P_1 might be needed to decode B_3 and B_4. In such cases, Equation 6.3 can be reduced to:

$$\text{Packet } p \text{ on-time if } \begin{cases} \tau_A \geq \tau_p + \tau_q^{(p)} & \text{if } p \text{ belongs to an } I \text{ or } P \text{ frame,} \\ \tau_A - T \geq \tau_p + \tau_q^{(p)} & \text{if } p \text{ belongs to a } B \text{ frame.} \end{cases} \tag{6.4}$$

To further simplify Equation 6.4, we define for a packet p the *variable latency*, $l^{(p)}$, to be:

$$l^{(p)} \triangleq \begin{cases} \tau_q^{(p)} & \text{if } p \text{ belongs to an } I \text{ or } P \text{ frame,} \\ \tau_q^{(p)} + T & \text{if } p \text{ belongs to a } B \text{ frame.} \end{cases}$$

Thus, the variable latency of an I or P packet is simply the queuing delay of that packet, and the variable latency of a B packet is the queuing delay plus the a time offset of T to adjust the difference between the bit-stream and display order of MPEG video. Then Equation 6.4 can be simplified to:

$$\text{Packet } p \text{ on-time if} \qquad \tau_A \geq \tau_p + l^{(p)} \tag{6.5}$$

For a video frame f, the variable latency $L^{(f)}$ is defined to be the maximum variable latency of all the packets that the frame contains. Thus, a frame f is on time for display if the following holds:

$$\text{Frame } f \text{ on-time if} \qquad \tau_A \geq \tau_p + L^{(f)} \tag{6.6}$$

If the variable frame latencies for all frames are consistently small, then a small τ_A is sufficient to guarantee smooth video playback, yielding low end-to-end delay and small buffer storage at the receiver. On the other hand, if the variable frame latencies have very large maximum values, then large end-to-end delay and buffer sizes are necessary for smooth playback.

Figs. 6.21 and 6.22 show the variable frame latencies achieved by the three packet classification schemes at times near 6 and 38 minutes from the start of the *First Strike* movie respectively. For each of the schemes, an identical bandwidth of 1.55 *Mbps* is used for each of the traffic classes. The load on the network is relatively high at $t = 6$ minutes and relatively low at $t = 38$ minutes, as can be inferred from Fig. 6.19 by the high and low distortion during the respective times. As expected, the variable latency value during the heavily loaded periods is generally larger than that of the lightly loaded periods for all packet tagging schemes. However, for the delay-optimized scheme, the maximum variable latency during the congested period is about 58 *ms*, while the maximum latency for the uniform and loss-optimized schemes are about 75 and 85 *ms* respectively during the same period. This means the end-to-end latency for the delay-optimized stream can be about 30 *ms* smaller than the other streams. During the lightly congested period, the latency difference between high-delay and low-delay classes is small, leading to little difference in the actual end-to-end latency.

According to Equation 6.6, a video frame f is late if the following holds:

$$\text{Frame } f \text{ is late if} \qquad \tau_A - \tau_p < L^{(f)}$$

We call the quantity $\tau_A - \tau_p$ the *delay parameter*, since the larger τ_A is, the larger end-to-end delay becomes. If the delay parameter is set to be small, then end-to-end delay is small, but the likelihood of a late frame is large. Conversely, one can set the delay parameter to be large, and hence decrease the likelihood of a late frame at the

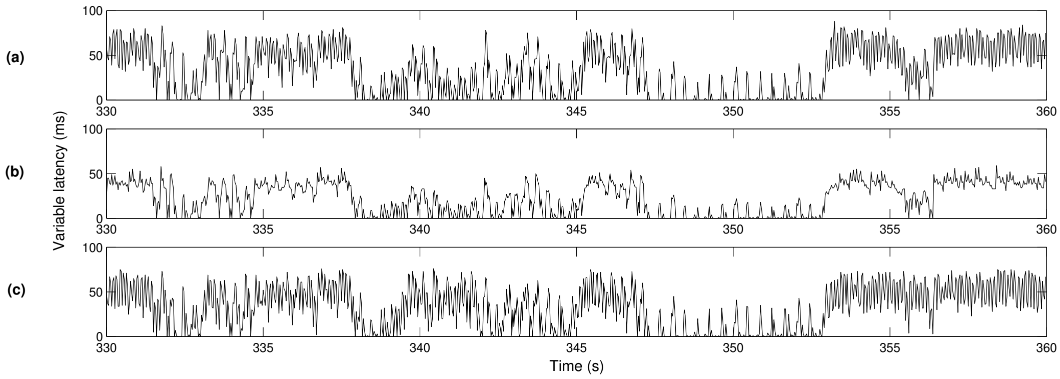


Figure 6.21: Variable frame latencies during periods of heavy congestion at time 6 minutes for (a) *uniform*, (b) *delay-optimized*, and (c) *loss-optimized* schemes.

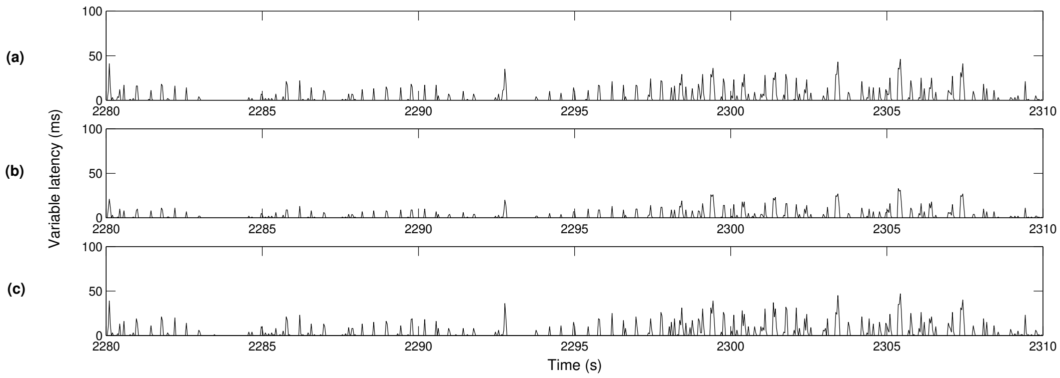


Figure 6.22: Variable frame latencies during periods of light congestion at time 38 minutes for (a) *uniform*, (b) *delay-optimized*, and (c) *loss-optimized* schemes.

expense of large end-to-end delay. To characterize the percentage of late video frames as a function of the delay parameter, we compute the cumulative density function of the variable frame latency $L^{(f)}$ for each of the traces shown in Figs. 6.21 and 6.22. The results are shown in Fig. 6.23 which characterizes the trade-off between end-to-end latency and the percentage of frame that will be late for the different packet classifiers. For instance, Fig. 6.23-(a) show that if we are willing to receive up to 10%

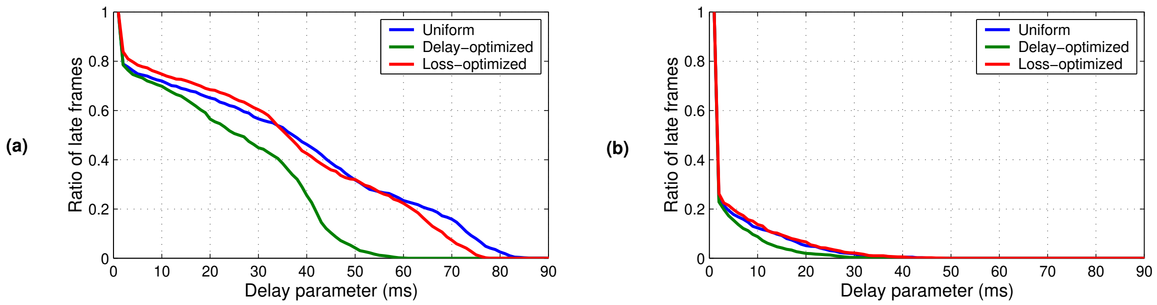


Figure 6.23: Ratio of late frames as a function of delay parameter for (a) period of heavy congestion at time 6 minutes, and (b) period of light congestion at time 38 minutes.

and 0% late frames during periods of congestion, we need a delay parameter of about 45 and 60 ms respectively under the delay-optimized scheme. Thus, a reduction in delay by 15 ms can be achieved at the cost of 10% late frames under the delay-optimized scheme. From Fig. 6.23-(a), we see that in order to achieve between 0 to 20% late frames during the congested period, the delay parameter required by the delay-optimize scheme is about 20 ms less than that of the loss-optimized scheme and 30 ms less than that of the uniform scheme.

6.8 Conclusions

This Chapter presented a framework for decomposing MPEG-compressed video bit-stream into a map of delay-loss spread. This delay-loss spread map can be used to tag packets for transmission using multiple traffic classes with different delay, loss and bandwidth. We have presented a mechanism of splitting an MPEG stream into parts with different delay requirement, and also compared experimentally several mechanisms for achieving loss and delay differentiation using commercial DVD movies. Finally, we presented simulation results, using an implementation of a DiffServ network and a DVD movie. We showed that distortion reduction of 4 dB is achievable by a packet classifier that optimizes for loss as compared to a classifier that treats the MPEG stream as homogeneous in importance. Another packet classifier that optimizes for delay achieves a reduction in end-to-end delay of 30 ms. The trade-off between end-to-end delay and the percentage of late frames are characterized using simulation for packet classifiers that optimize for loss and delay respectively.

6.A Appendix – Loss Process Parameters

In this Appendix, we describe the loss process that is used in this paper to generate fixed packet loss rates. A two-state Markov loss process of Fig. 6.A is used, with a *good* state corresponding to no packet loss, and a *bad* state corresponding to packet loss. Denote the transitional probability from the good state to the bad state by p,

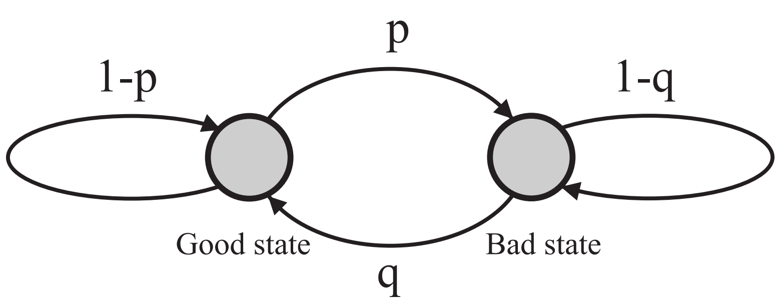


Figure 6.A: A two-state Markov process for modelling network losses.

and that from the bad state to the good state by q. Then, the measurement reported in [151] yields $(p, q) = (0.0347, 0.9091)$ when loss rate is 3.68%, and (0.0004, 0.9792) when the loss rate is 0.041%. To interpolate intermediate values of loss rate, we parametrize p and q using a parameter t:

$$\begin{aligned} p &= 0.0004 + 0.0343t \\ q &= 0.9792 - 0.0701t. \end{aligned} \tag{6.A}$$

Thus, by varying t between 0 and 1, all intermediate loss rates can be generated. Since the loss rate r is given by $r = p/(p + q)$, where p and q are functions of t, we can re-arrange and express the parameter t as a function of r:

$$t = \frac{0.9832r - 0.0004}{0.0343 + 0.0358r}. \tag{6.B}$$

Equations 6.A and 6.B allow the computation of the model parameters p and q as a function of the desired loss rate r.

Chapter 7

Summary and Recommendations for Future Work

7.1 Summary

In this dissertation, we have explored several areas of video compression and communication. We have presented an error-resilient and bandwidth-scalable video compression technique, SVIP, as a plausible choice for Internet communications due to its ability to facilitate rate control and error control. While prioritized transport is assumed by most scalable compression schemes, SVIP can be effectively used in best-effort networks by producing individually decodable packets of equal visual importance. We have also demonstrated possible ways in which SVIP can be used in both the unicast and multicast environments.

For unicast networks, we showed that the use of SVIP with a TCP-friendly transport protocol can achieve TCP-friendliness, low-latency, and small bandwidth variation when transporting pre-compressed video. TCP-friendliness on a macroscopic scale is established by using a model that relates TCP throughput to the measurable quantities of packet loss rate and round-trip propagation delay. Since video compressed under SVIP can be easily filtered to instantaneously meet the bandwidth constraint imposed by the rate-control algorithm, low latency can be achieved. Finally, by matching TCP throughput on a macroscopic rather than a microscopic scale, unnecessary bandwidth fluctuations are eliminated for our proposed rate-control algorithm.

For multicast networks, we have presented layered FEC as an efficient error-control mechanism under a layered multicast framework. We showed that imposing a fixed order in which FEC layers must be added generally results in suboptimal protection. We then described a more general scheme of organizing the FEC layers and showed how the optimal combination of data and FEC layers can be determined. For rate control algorithms at the receivers, we presented an equation-based rate control mechanism and show that it can achieve more fair bandwidth allocations than existing multicast rate control schemes.

Motivated by the possible deployment of DiffServ in the Internet, and the dominance of existing video contents in the MPEG format, we presented a transmission scheme for MPEG video using multiple traffic classes in a DiffServ network. By dividing an MPEG-compressed bit-stream into sub-streams of different delay and loss sensitivities, and carrying these sub-streams in traffic classes with corresponding quality of service, better overall network utilization is achieved.

7.2 Recommendations for Future Work

The error-resilience property of SVIP is related to the idea of multiple description coding (MDC) that has recently received much attention in the research community. The idea behind MDC is that multiple *descriptions* of a source are produced so that (a) each description individually corresponds to a low-quality, but usable representation of the source, and (b) combining more descriptions leads to a higher quality reproduction of the source. Thus, SVIP can technically be considered as a multiple description code. However, SVIP differs from existing MDC codes in several aspects. First, existing MDC codes often provide mechanisms for changing the correlation between descriptions while SVIP does not. Second, existing MDC codes often produce only two descriptions while SVIP can produce tens of descriptions. To facilitate error control, one would ideally desire both the ability to adjust the correlation between descriptions and to produce many descriptions. Thus, an interesting research area is to seek MDC schemes that achieves both.

In the area of rate control, one interesting research problem is to look beyond TCP-friendliness and study the global implications of rate-control algorithms. Since it has been shown that there exist rate allocations that optimize overall network throughput while denying access to certain users [46], the goal of rate-control is often to achieve some form of globally fair allocation [83]. It has recently been shown mathematically in [65] that in a scheme in which a network advertises the appropriate usage price to the senders, it is possible for uncooperative receivers to achieve a globally fair allocation. It is interesting to investigate how such a scheme may be implemented in a distributed environment like the Internet.

Finally, the notion of fair bandwidth allocations are typically developed for unicast communications. In a network in which unicast and multicast traffic coexist, it becomes necessary to develop a notion of fairness that takes into account both traffic types. Currently, there is no multicast rate control schemes that allow a multicast session with more receivers to assume more bandwidth than another session with fewer receivers.

Bibliography

[1] A. Albanese, J. Blömer, J. Edmonds, M. Luby, and M. Sudan. Priority encoding transmission. *IEEE Trans. Info. Theory*, 42(6), pp. 1737–44, November 1997.

[2] K. Almeroth. The evolution of multicast: From the MBone to interdomain multicast to Internet2 deployment. *IEEE Network Magazine*, pp. 10–20, January 2000.

[3] E. Amir, S. McCanne, and R. Katz. Receiver-driven bandwidth adaptation for light-weight sessions. In *Proc. ACM Intl. Multimedia Conf. '97*, pp. 415–26, Seattle, WA., November 1997.

[4] R. Aravind, M. Civanlar, and A. Reibman. Packet loss resilience of MPEG-2 scalable video coding algorithm. *IEEE Trans. Circuits Sys. Video Tech.*, 6(5), pp. 426–35, October 1996.

[5] S. Bajaj, L. Breslau, and S. Shenker. Uniform versus priority dropping for layered video. In *Proc. ACM Sigcomm '98*, pp. 131–43, Vancouver, Canada, September 1998.

[6] A. Ballardie. Core based trees (CBT) multicast routing architecture. *RFC2201*, September 1997.

[7] A. Banerjea, W. Tan, and A. Zakhor. A layered compression scheme for multicasting medical images across heterogeneous networks. In *Proc. SPIE Medical Imaging 1997*, volume 3031, pp. 265–76, Newport Beach, CA., February 1997.

[8] T. Bates, R. Chandra, D. Katz, and Y. Rekhter. Multiprotocol extensions for BGP-4. *RFC2283*, February 1998.

[9] S. Blake, D. Black, M. Carlson, E. Davis, Z. Wang, and W. Weiss. An architecture for differentiated services. *RFC2475*, December 1998.

[10] J. Bolot, S. Fosse-Parisis, and D. Towsley. Adaptive FEC-based error control for Internet telephony. In *Proc. IEEE Infocom '99*, volume 3, pp. 1453–60, New York, NY., March 1999.

[11] J. Bolot and A. Garcia. The case for FEC-based error control for packet audio in the Internet. To appear, *ACM/Springer Multimedia Systems*, 1999.

[12] J. Bolot, T. Turletti, and I. Wakeman. Scalable feedback control for multicast video distribution in the Internet. In *Proc. ACM Sigcomm '94*, pp. 58–67, London, England, September 1994.

[13] J. Boyce and R. Gaglianello. Packet loss effects on MPEG video sent over the public Internet. In *Proc. ACM Intl. Multimedia Conf. '98*, pp. 181–90, Bristol, U.K., September 1998.

[14] D. Brown. A bitrate control algorithm for the Berkeley MPEG-1 video encoder. Master's thesis, University of California, Berkeley, 1997.

[15] J. Byers, M. Luby, M. Mitzenmacher, and A. Rege. A digital fountain approach to reliable distribution of bulk data. In *Proc. ACM Sigcomm '98*, pp. 56–67, Vancouver, Canada, September 1998.

[16] V. Cerf and R. Kahn. A protocol for packet network intercommunication. *IEEE Trans. Comm.*, 22(5), pp. 637–48, May 1974.

[17] Y. Chang. *Delay Cognizant Video Coding*. PhD thesis, University of California, Berkeley, 1998.

[18] Y. Chawathe, S. McCanne, and E. Brewer. RMX: Reliable multicast in heterogeneous networks. In *Proc. IEEE Infocom '00*, Tel Aviv, Israel, March 2000.

[19] S. Cheung, M. Ammar, and X. Li. On the use of Destination Set Grouping to improve fairness in multicast video distribution. In *Proc. IEEE Infocom '96*, pp. 553–60, San Francisco, CA., March 1996.

[20] P. Chou, A. Mohr, A. Wang, and S. Mehrotra. FEC and pseudo-ARQ for receiver-driven layered multicast of audio and video. Submitted, *IEEE Data Compression Conference*, March 2000.

[21] P. Chou and K. Ramchandran. Clustering source/channel rate allocations for receiver-driven multicast with error control under a limited number of streams. Submitted, *IEEE Intl. Conf. Multimedia Expo.*, July 2000.

[22] I. Cidon, A. Khamisy, and M. Sidi. Analysis of packet loss processes in high-speed networks. *IEEE Trans. Info. Theory*, 39(1), pp. 98–108, January 1993.

[23] D. Clark. The design philosophy of the darpa internet protocols. *ACM Computer Comm. Review*, 25(1), pp. 102–11, January 1995.

[24] D. Clark and W. Fang. Explicit allocation of best-effort packet delivery service. *ACM/IEEE Trans. Networking*, 6(4), pp. 362–73, August 1998.

[25] D. Clark and D. Tennehouse. Architectural considerations for a new generation of protocols. In *Proc. ACM Sigcomm '90*, pp. 200–8, Philadelphia, PA., September 1990.

[26] G. Conklin and S. Hemami. Evaluation of temporally scalable video coding techniques. In *Intl. Conf. Image Processing '97*, volume 2, pp. 61–4, Santa Barbara, CA., October 1997.

[27] G. Cote, B. Erol, M. Gallant, and F. Kossentini. H.263+ video coding at low bit rates. *IEEE Trans. Circuits Sys. Video Tech.*, 8(7), pp. 849–66, November 1998.

[28] A. Dagiuklas and M. Ghanbari. Priority mechanisms in ATM switches carrying two layer video data. *Electronic Letters*, 29(3), pp. 273–4, February 1993.

[29] J. Danskin, G. Davis, and X. Song. Fast lossy internet image transmission. In *Proc. ACM Intl. Multimedia Conf. '95*, pp. 321–33, San Francisco, CA., November 1995.

[30] W. Davis and J. Irza. Cable modems take the early lead. *Byte*, 23(1), January 1998.

[31] S. Deering. Host extensions for IP multicasting. *RFC1112*, August 1989.

[32] S. Deering. Internet multicast routing: State of the art and open research issues. In *Multimedia Integrated Conferencing for Europe (MICE) Seminar at the Swedish Institue of Computer Science, Stockholm*, October 1993.

[33] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, A. Helmy, D. Meyer, and L. Wei. Protocol independent multicast version 2 dense mode specification. *Internet draft, draft-ietf-pim-v2-00.txt*, August 1998.

[34] S. Deering, D. Estrin, D. Farinacci, V. Jacobson, C. Liu, and L. Wei. The PIM architecture for wide-area multicast routing. *ACM/IEEE Trans. Networking*, 4(2), pp. 153–62, April 1996.

[35] A. Demers, S. Keshev, and S. Shenker. Analysis and simulation of a fair queuing algorithm. *Internetworking Research and Experience*, 1(1), pp. 3–26, September 1990.

[36] C. Diot, W. Dabbous, and J. Crowcroft. Multipoint communication: A survey of protocols, functions and mechanisms. *IEEE J. Selected Area Comm.*, 15(3), pp. 277–90, April 1997.

[37] C. Diot, B. Levine, B. Lyles, H. Kassem, and D. Balensiefen. Deployment issues for the IP multicast service and architecture. *IEEE Network Magazine*, pp. 78–88, January 2000.

[38] C. Dovrolis and P. Ramanathan. A case for relative differentiated services and the proportional differentiation model. *IEEE Network Magazine*, pp. 26–34, October 1999.

[39] P. Dubey. Inefficiency of Nash equilibria. *Mathematics of Operations Research*, 11(1), pp. 1–8, February 1986.

[40] D. Estrin, D. Farinacci, A. Helmy, D. Thaler, S. Deering, M. Handley, V. Jacobson, C. Liu, P. Sharma, and L. Wei. Protocol independent multicast-sparse mode (PIM-SM): Protocol specification. *RFC2362*, June 1998.

[41] N. Färber, K. Stuhlmüller, and B. Girod. Analysis of error propagation in hybrid video coding with application to error resilience. In *Intl. Conf. Image Processing '99*, volume 2, pp. 550–4, Kobe, Japan, October 1999.

[42] S. Floyd. Connections with multiple congested gateways in packet-switched networks. part 1: one-way traffic. *ACM Computer Comm. Review*, 21(5), pp. 30–47, October 1991.

[43] S. Floyd, M. Handley, and J. Padhye. Equation-based congestion control for unicast applications. Submitted. *SIGCOMM 2000.*

[44] S. Floyd, V. Jacobson, C. Liu, S. McCanne, and L. Zhang. A reliable multicast framework for light-weight sessions and application level framing. *ACM/IEEE Trans. Networking*, 5(6), pp. 784–803, December 1997.

[45] M. Garrett and M. Vetterli. Joint source/channel coding of statistically multiplexed real-time services on packet networks. *ACM/IEEE Trans. Networking*, 1(1), pp. 71–80, February 1993.

[46] M. Gerla and M. Staskausas. Fairness in flow controlled networks. *Journal of Telecommunication Networks*, pp. 29–38, 1982.

[47] M. Ghanbari. Two-layer coding of video signals for VBR networks. *IEEE J. Selected Area Comm.*, 7(5), pp. 771–81, June 1989.

[48] S. Golestani. Fundamental observations on multicast congestion control in the Internet. In *Proc. IEEE Infocom '99*, volume 2, pp. 990–1000, New York, NY, March 1999.

[49] K. Gong and L. Rowe. Parallel mpeg-1 video encoding. In *Proc. Picture Coding Symposium*, Sacramento, CA., September 1994.

[50] R. Gopalakrishnan, J. Griffioen, G Hjálmtýsson, C. Sreenan, and S. Wen. A simple loss differentiaion approach to layered multicast. To appear, *Proc. IEEE Infocom 2000*.

[51] J. Hagenauer. Rate-compatible punctured convolutional codes (RCPC codes) and their applications. *IEEE Trans. Comm.*, 36(4), pp. 389–400, April 1988.

[52] M. Handley and V. Jacobson. SDR: Session description protocol. *RFC2327*, April 1998.

[53] J. Heinanen, T. Finland, F. Baker, W. Weiss, and J. Wroclawski. Assured forwarding PHB group. *RFC2597*, June 1999.

[54] W. Heinzelman, M. Budagavi, and R. Talluri. Unequal error protection of MPEG-4 compressed video. In *Intl. Conf. Image Processing '99*, volume 2, pp. 503–4, Kobe, Japan, October 1999.

[55] T. Henderson, E. Sahouria, S. McCanne, and R. Katz. On improving the fairness of TCP congestion avoidance. In *Proc. IEEE Globecom '98*, volume 1, pp. 539–44, Sydney, Australia, November 1998.

[56] D. Hoffman, G. Fernando, and V. Goyal. RTP payload format for MPEG1/MPEG2 video. *RFC2250*, January 1998.

[57] H. Holbrook and D. Cheriton. IP multicast channels: EXPRESS support for large-scale single-source applications. In *Proc. ACM Sigcomm '99*, pp. 65–78, Cambridge, MA., October 1999.

[58] U. Horn, K. Stuhlmüller, M. Link, and B. Girod. Robust Internet video transmission based on scalable coding and unequal error protection. *Signal Processing*, 15(1-2), pp. 77–94, September 1999.

[59] ITU-T. Rec. H.263, video coding for low bitrate communications, version 2. International Telecommunication Union, February 1998.

[60] S. Jacobs and A. Eleftheriadis. Streaming video using dynamic rate shaping and tcp congestion control. *Journal of Visual Communication and Image Representation*, 9(3), pp. 211–22, September 1998.

[61] V. Jacobson. Congestion avoidance and control. In *Proc. ACM Sigcomm '88*, pp. 314–29, Stanford, CA., August 1988.

[62] V. Jacobson, K. Nichols, and K. Poduri. An expedited forwarding PHB. *RFC2598*, June 1999.

[63] H. Kanakia, P. Misha, and A. Reibman. An adaptive congestion control scheme for real time packet video transport. *ACM/IEEE Trans. Networking*, 3(6), pp. 671–82, December 1995.

[64] J. Kato, A. Shimizu, and S. Goto. Active measurement and analysis of delay time in the Internet. In *Proc. ICPP Workshop*, pp. 254–9, Aizu-Wakamutsu, Japan, September 1999.

[65] F. Kelly. Notes on effective bandwidths. In *Stochastic Networks: Theory and Applications*, pp. 141–68, Oxford University Press, 1996.

[66] F. Kelly, A. Maulloo, and D. Tan. Rate control for communication networks: shadow prices, proportional fairness and stability. *Journal of Operational Research Society*, 49(3), pp. 237–52, March 1998.

[67] R. Kermode. Scoped hybrid automatic repeat request with forward error correction (SHARQFEC). In *Proc. ACM Sigcomm '98*, pp. 278–89, Vancouver, Canada, September 1998.

[68] B. Kim and W. Pearlman. Low-delay embedded 3-d wavelet color video coding with SPIHT. In *Proc. SPIE Visual Comm. and Image Processing '98*, volume 3309, pp. 955–64, San Jose, CA., January 1998.

[69] M. Krunz and H. Hughes. A performance study of loss priorities for MPEG video traffic. In *Proc IEEE International Conference on Communications*, pp. 1756–60, Seattle, WA, June 1995.

[70] S. Kumar, P. Radoslavov, D. Thaler, C. Alaettinoğlu, D. Estrin, and M. Handley. The MASC/BGMP architecture for inter-domain multicast routing. In *Proc. ACM Sigcomm '98*, pp. 93–104, Vancouver, Canada, September 1998.

[71] M. Kunt. Block coding of graphics: A tutorial review. *Proc. IEEE*, 68(7), pp. 770–86, July 1980.

[72] F. Léannec, F. Toutain, and C. Guillemot. Packet loss resilient MPEG-4 compliant video coding for the Internet. *Signal Processing*, 15(1-2), pp. 35–56, September 1999.

[73] C. Leicher. Hierarchical encoding of MPEG sequences using priority encoding. Technical report, TR-94-058, ICSI, http://www.icsi.berkeley.edu/PET/pet-documents.html, November 1994.

[74] B. Leiner, V. Cerf, D. Clark, R. Kahn, L. Kleinrock, D. Lynch, J. Postel, L. Roberts, and S. Wolff. A brief history of the internet. The Internet Society, 2000.

[75] X. Li, M. Ammar, and S. Paul. Video multicast over the Internet. *IEEE Network*, 13(2), pp. 46–60, April 1999.

[76] X. Li, S. Paul, and M. Ammar. Layered video multicast with retransmissions (LVMR): Evaluation of hierarchical rate control. In *Proc. IEEE Infocom '98*, volume 3, pp. 1062–72, San Francisco, CA., March 1998.

[77] X. Li, S. Paul, and M. Ammar. Multi-session rate control for layered video multicast. In *Proc. SPIE Multimedia Computing and Networking*, volume 3654, pp. 175–89, San Jose, CA., January 1999.

[78] X. Li, S. Paul, P. Pancha, and M. Ammar. Layered video multicast with retransmission (LVRM): Evaluation of error control scheme. In *Proc. IEEE NOSSDAV '97*, pp. 161–72, New York, NY., May 1997.

[79] J. Mahdavi and S. Floyd. Tcp-friendly unicast rate-based flow control. Technical note sent to the end2end-interest mailing list, 1997.

[80] D. Mandelbaum. An adaptive-feedback coding scheme using incremental redundancy. *IEEE Trans. Info. Theory*, 20(3), pp. 388–9, May 1974.

[81] M. Mathis, J. Semke, J. Mahdavi, and T. Ott. The macroscopic behaviour of the TCP congestion avoidance algorithm. *ACM Computer Comm. Review*, 27(3), pp. 67–82, July 1997.

[82] M. May, J. Bolot, A. Jean-Marie, and C. Diot. Simple performance models of differentiated services schemes for the Internet. In *Proc. IEEE Infocom '99*, volume 3, pp. 1385–94, New York, NY., March 1999.

[83] R. Mazumdar, L. Mason, and C. Douligeris. Fairness in network optimal flow control: Optimality of product forms. *IEEE Trans. Comm.*, 39(5), pp. 775–82, May 1991.

[84] S. McCanne, V. Jacobson, and M. Vetterli. Receiver-driven layered multicast. In *Proc. ACM Sigcomm '96*, pp. 117–30, Palo Alto, CA., August 1996.

[85] S. McCanne, M. Vetterli, and V. Jacobson. Low-complexity video coding for receiver-driven layered multicast. *IEEE J. Selected Area Comm.*, 15(6), pp. 983–1001, August 1997.

[86] A. Mohr, E. Riskin, and R. Ladner. Graceful degradation over packet erasure channels through forward error correction. In *IEEE Data Compression Conference '99*, Snowbird, UT, March 1999.

[87] A. Mohr, E. Riskin, and R. Ladner. Graceful degradation over packet erasure channels through forward error correction. In A.Storer and M.Cohn, editors, *IEEE Data Compression Conference '99*, pp. 92–101, Snowbird, UT., March 1999.

[88] J. Moy. Multicast routing extensions for OSPF. *Comm. ACM*, 37(8), pp. 61–6, August 1994.

[89] CCITT Recommendation MPEG-2. Information technology - generic coding of moving pictures and associated audio information: Video. ISO/IEC 13818-2, 1996.

[90] J. Nonnenmacher, E. Biersack, and D. Towsley. Parity-based loss recovery for reliable multicast transmission. *ACM/IEEE Trans. Networking*, 6(4), pp. 349–61, August 1998.

[91] J. Nonnenmacher, L. Martin, J. Matthias, E. Biersack, and G. Carle. How bad is reliable multicast without local recovery. In *Proc. IEEE Infocom '98*, volume 3, pp. 972–9, San Francisco, CA., March 1998.

[92] J. Padhye, V. Firoiu, D. Towsley, and J. Kurose. Modeling tcp throughput: a simple model and its empirical validation. In *Proc. ACM Sigcomm '98*, pp. 303–14, Vancouver, Canada, October 1998.

[93] J. Padhye, J. Kurose, D. Towsley, and R. Koodli. A TCP-friendly rate adjustment protocol for continuous media flows over best effort networks. In *ACM SIGMETRICS '99*, pp. 220–1, Atlanta, GA, May 1999.

[94] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The single-node case. *ACM/IEEE Trans. Networking*, 1(3), pp. 344–57, June 1993.

[95] A. Parekh and R. Gallager. A generalized processor sharing approach to flow control in integrated services networks: The multiple-node case. *ACM/IEEE Trans. Networking*, 2(2), pp. 137–50, April 1994.

[96] K. Patel, B. Smith, and L. Rowe. Performance of a software mpeg video decoder. In *Proc. ACM Intl. Multimedia Conf. '93*, pp. 75–82, Anaheim, CA., August 1993.

[97] S. Paul, K. Sabnani, J. Lin, and S. Bhattacharyya. Reliable multicast transport protocol (RMTP). *IEEE J. Selected Area Comm.*, 15(3), pp. 407–21, April 1997.

[98] S. Pejhan, M. Schwartz, and D. Anastassiou. Error control using retransmission schemes in multicast transport protocols for real-time media. *ACM/IEEE Trans. Networking*, 4(3), pp. 413–27, June 1996.

[99] R. Perlman and S. Raman. Techniques for making ip multicast simple and scalable. In *Proc. First International Workshop on Networked Group Communication*, Pisa, November 1999.

[100] C. Podilchuk, N. Jayant, and N. Farvardin. Three-dimensional subband coding of video. *IEEE Trans. Image Processing*, 4(2), pp. 125–39, February 1995.

[101] J. Postel, C. Sunshine, and D. Cohen. The ARPA Internet Protocol. *Computer Networks*, 5(4), pp. 261–71, July 1981.

[102] A. Puri, L. Yan, and B. Haskell. Temporal resolution scalable video coding. In *Intl. Conf. Image Processing '94*, volume 2, pp. 947–51, Austin, TX, November 1994.

[103] H. Radha and Y. Chen. Fine-granular-scalable video for packet networks. In *Packet Video Workshop '99*, New York, NY., April 1999.

[104] D. Redmill and N. Kingsbury. The EREC: An error-resilient technique for coding variable-length blocks of data. *IEEE Trans. Image Processing*, 5(4), pp. 565–74, April 1996.

[105] J. Reynolds and R. Braden. Internet official protocol standards. *RFC2500*, June 1999.

[106] I. Rhee. Error control techniques for interactive low bit-rate video transmission over the Internet. In *Proc. ACM Sigcomm '98*, pp. 290–301, Vancouver, Canada, September 1998.

[107] I. Rhee and S. Joshi. Error recovery for interactive video transmission over the Internet. *IEEE J. Selected Area Comm.*, 18(6), pp. 1033–49, June 2000.

[108] L. Rizzo and L. Vicisano. A reliable multicast data distribution protocol based on software FEC techniques. In *Proc. 4th Workshop on High Performance Comm. Subsystems*, pp. 115–24, Chalkidiki, Greece, June 1997.

[109] J. Rosenberg and H. Schulzrinne. An RTP payload format for generic forward error correction. *RFC2733*, December 1999.

[110] L. Rowe. The Berkeley Multimedia, Interfaces, and Graphics Seminar. http://bmrc.berkeley.edu, January 1995.

[111] L. Rowe, K. Patel, B. Smith, and K. Liu. MPEG video in software: representation, transmission, and playback. In *Proc. of SPIE 1994 Int'l Symp. on Elec. Imaging: Science and Technology*, pp. 134–4, San Jose, CA., February 1994.

[112] A. Said and W. Pearlman. A new, fast, and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. Circuits Sys. Video Tech.*, 6(3), pp. 243–50, June 1996.

[113] J. Saltzer, D. Reed, and D. Clark. End-to-end arguments in system design. *ACM Trans. Computer System*, 2(4), pp. 277–88, November 1984.

[114] N. Shacham. Multipoint communication by hierarchically encoded data. In *Proc. IEEE Infocom '92*, volume 3, pp. 2107–14, Florence, Italy, May 1992.

[115] T. Shanableh and M. Ghanbari. Heterogeneous video transcoding to lower spatio-temporal resolutions and different encoding formats. *IEEE Trans. Multimedia*, 2000.

[116] C. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3), pp. 379–423, July 1948.

[117] J. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Trans. Signal Processing*, 41(12), pp. 3445–62, December 1993.

[118] S. Shenkar. Fundamental design issues for the future internet. *IEEE J. Selected Area Comm.*, 13(7), pp. 1176–88, September 1995.

[119] D. Sisalem and A. Wolisz. Mlda: TCP-friendly congestion control framework for heterogeneous multicast environments. In *IEEE Intl. Workshop on QoS*, pp. 65–74, Pittsburgh, PA., June 2000.

[120] J. Smith. VideoZoom spatio-temporal video browser. *IEEE Trans. Multimedia*, 1(2), pp. 157–71, June 1999.

[121] W. Stevens. *TCP/IP Illustrated*. Addison-Wesley, 1996.

[122] A. Swan, S. McCanne, and L. Rowe. Layered transmission and caching for the multicast session directory service. In *Proc. ACM Intl. Multimedia Conf. '98*, pp. 119–28, New York, NY., September 1998.

[123] W. Tan and A. Zakhor. A real-time software decoder for scalable video on multi-processors. In *Packet Video Workshop '96*, Brisbane, Australia, March 1996.

[124] W. Tan and A. Zakhor. Real time software implementation of scalable video codec. In *Intl. Conf. Image Processing '96*, volume 1, pp. 17–20, Lausanne, Switzerland, September 1996.

[125] W. Tan and A. Zakhor. Internet video using error resilient scalable compression and cooperative transport protocol. In *Intl. Conf. Image Processing '98*, volume 3, pp. 458–62, Chicago, IL., October 1998.

[126] W. Tan and A. Zakhor. Resilient compression of video for transmission over the Internet. In *Proc. 32nd Asilomar Conf. Signal, Systems and Computers*, volume 3, pp. 243–7, Pacific Grove, CA., November 1998.

[127] W. Tan and A. Zakhor. Error control for video multicast using hierarchical FEC. In *Intl. Conf. Image Processing '99*, volume 1, pp. 401–5, Kobe, Japan, October 1999.

[128] W. Tan and A. Zakhor. Multicast transmission of scalable video using receiver-driven hierarchical FEC. In *Packet Video Workshop '99*, New York, NY., April 1999.

[129] W. Tan and A. Zakhor. Real-time Internet video using error resilient scalable compression and TCP-friendly transport protocol. *IEEE Trans. Multimedia*, 1(2), pp. 172–86, June 1999.

[130] W. Tan and A. Zakhor. Streaming MPEG video over delay and loss differentiated networks. Submitted. *Packet Video Workshop 2001.*, May 2001.

[131] W. Tan and A. Zakhor. Video multicast using layered fec and scalable compression. To appear. *IEEE Trans. Circuits Sys. Video Tech.*, February 2001.

[132] T. Tanimoto. An elementary mathematical theory of classification and prediction. Technical report, IBM Corp., 1958.

[133] D. Taubman. *Directionality and Scalability in Image and Video Compression.* PhD thesis, University of California, Berkeley, 1994.

[134] D. Taubman and A. Zakhor. Multirate 3-d subband coding of video. *IEEE Trans. Image Processing*, 3(5), pp. 572–88, September 1994.

[135] D. Taubman and A. Zakhor. A common framework for rate and distortion based scaling of highly scalable compressed video. *IEEE Trans. Circuits Sys. Video Tech.*, 6(4), pp. 329–54, August 1996.

[136] D. Thaler, D. Estrin, and D. Meyer. Border gateway multicast protocol (BGMP). *Internet draft, draft-ietf-idmr-gum-04.txt*, November 1998.

[137] D. Tse, R. Gallager, and J. Tsitsiklis. Statistical multiplexing of multiple time-scale markov streams. *IEEE J. Selected Area Comm.*, 13(6), pp. 1208–39, August 1995.

[138] T. Turletti and C. Huitema. Video-conferencing on the Internet. *ACM/IEEE Trans. Networking*, 4(3), pp. 340–51, June 1996.

[139] T. Turletti, S. Parisis, and J. Bolot. Experiments with a layered transmission scheme over the Internet. Technical report, INRIA, http://www.inria.fr/RRRT/RR-3296.html, November 1997.

[140] L. Vicisano, L. Rizzo, and J. Crowcroft. TCP-like congestion control for layered multicast data transfer. In *Proc. IEEE Infocom '98*, volume 3, pp. 996–1003, San Francisco, CA., March 1998.

[141] B. Vickers, M. Lee, and T. Suda. Feedback control mechanisms for real-time multipoint video services. *IEEE J. Selected Area Comm.*, 15(3), pp. 512–30, April 1997.

[142] D. Waitzman, C. Partridge, and S. Deering. Distance vector multicast routing protocol. *RFC1075*, November 1988.

[143] H. Wang and M. Schwartz. Achieving bounded fairness for multicast and tcp traffic in the Internet. In *Proc. IEEE Infocom '98*, volume 1, pp. 81–92, San Francisco, CA., March 1998.

[144] Y. Wang and Q. Zhu. Error control and concealment for video communication: A review. *Proc. IEEE*, 86(5), pp. 974–97, May 1998.

[145] C. Weinstein and J. Forgie. Experience with speech communication in packet networks. *IEEE J. Selected Area Comm.*, 1(6), pp. 963–80, December 1983.

[146] P. White. RSVP and Integrated Services in the Internet: A Tutorial. *IEEE Comm. Magazine*, pp. 100–6, May 1997.

[147] S. Wicker. *Error Control Systems for Digital Communication and Storage*. Prentice-Hall, 1995.

[148] D. Wilson and M. Ghanbari. An efficient loss priority scheme for MPEG-2 variable bit rate video for atm networks. In *Proc. IEEE Globecom '96*, pp. 1954–8, London, U.K., November 1996.

[149] L. Winger and A. Venetsanopoulos. Temporally scalable motion compensated adaptive temporal subband coding of video. In *Proc. IEEE ICASSP '99*, pp. 2247–50, Phoenix, AZ., March 1999.

[150] X. Xu, A. Myers, H. Zhang, and R. Yavatkar. Resilient multicast support for continuous-media applications. In *Proc. IEEE NOSSDAV '97*, pp. 183–94, New York, NY., May 1997.

[151] M. Yajnik, S. Moon, J. Kurose, and D. Towsley. Measurement and modelling of the temporal dependence in packet loss. In *Proc. IEEE Infocom '99*, pp. 345–52, New York, NY, March 199.

[152] L. Yun. *Transport for Multimedia in Wireless Networks*. PhD thesis, University of California, Berkeley, 1995.

[153] W. Zeng and B. Liu. Rate shaping by block dropping for transmission of MPEG-precoded video over channels of dynamic bandwidth. In *Proc. ACM Intl. Multimedia Conf. '96*, pp. 385–93, Boston, MA, November 1996.

[154] H. Zhang. Service disciplines for guaranteed performance service in packet-switching networks. *Proc. IEEE*, 83(10), pp. 1374–96, oct 1995.

[155] L. Zhang, S. Deering, D.Estrin, S. Shenker, and D.Zappala. RSVP: A new resource reservation protocol. *IEEE Network Magazine*, 7(5), pp. 8–18, September 1993.

[156] T. Zhang and Y. Xu. Unequal packet loss protection for layered video transmission. *IEEE Trans. Broadcasting*, 45(2), pp. 243–52, June 1999.