ERROR CONCEALMENT FOR ROBUST IMAGE AND VIDEO TRANSMISSIONS ON THE INTERNET

BY

XIAO SU

B.E., Zhejiang University, 1994
M.S., University of Illinois at Urbana-Champaign, 1997

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2001

Urbana, Illinois

Abstract

Coding and transmission of images and videos have been a popular but very challenging topic. Traditional coding algorithms are usually designed to optimize compression ratio in an error-free environment but not for the current Internet that is only a best-effort, packet-switched and unreliable network. This conflict presents a number of challenges for high-quality image and video transmissions.

Information loss and bandwidth limitation are two major factors that affect the quality of video streaming. In this thesis, we have developed various error concealment and reconstruction-based rate control schemes to address these two issues. First, we have proposed a sender-receiver based approach for designing a multiple-description video coder that facilitates recovery of *packet losses*. Second, we have studied artificial neural network-based reconstruction algorithms for compensating nonlinear *compression losses* due to multiple-description coding. Third, we have employed syntax-based packetization and decoder-side feedback for reducing *propagation losses*. Last, we have incorporated the reconstruction process in the design and evaluation of rate control schemes.

Likewise, delay and packet loss are two primary concerns for image transmissions. Existing approaches for delivering single-description coded images by TCP give superior quality but very long delays when the network is unreliable. To reduce delays, we have proposed to use UDP to deliver sender-receiver based multiple-description coded images. To further improve the trade-offs between delay and quality of either TCP delivery or UDP delivery, we have investigated a combined TCP/UDP delivery that can give shorter delay than pure TCP delivery and better quality than pure UDP delivery.

To my husband, Tao, and my parents

Table of Contents

Chapter

Acknowledgments

Looking back on my graduate study in the University of Illinois, I am truly grateful to many people who have helped me become a mature and confident person, ready for challenges in my future career.

First, special thanks go to my advisor, Prof. Benjamin Wah, for his valuable insights, encouragement and consistent support. He not only introduced me to the networking area but also inspired me to meet the challenges along the way. His help and impact on me can hardly be acknowledged using just a few words.

I would like to thank Prof. Thomas Huang, Prof. Klara Nahrstedt, and Prof. Lui Sha, for taking their valuable time to be on my thesis committee and for providing many helpful comments and suggestions.

I would also like to thank Dong Lin and all the other members in our research group for providing constructive comments on my work and for providing a congenial environment for me to work in.

I also wish to thank all my friends, Jie Yu, Qiongfei Qiu, Chen Wang, Yu Zhong, Guoming Shou, Tianjiao Qiu, and many others, for their incessant help in my work and life.

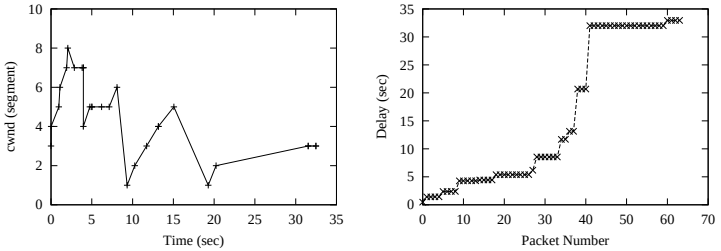
I am also indebted to my husband, Tao, and my parents for their everlasting love, support, and motivations. They have always been my source of strength at times of difficulties.

This research was supported by a grant from Conexant, Inc., and Mindspeed Technologies.

List of Tables

List of Figures

algorithms. Its congestion control schemes lead to variable sending rates, which in turn result in variable end-to-end packet delays from the applications' point of view. At the same time, the exponential backoffs of retransmissions and coarse grained timeouts can sometimes lead to long transmission delays at time of network congestion.



a) b)

Figure 1.1: The congestion window and corresponding end-to-end round-trip packet delays for a TCP connection between a host (`cw.crhc.uiuc.edu`) in Urbana, USA, and a remote echo server (`www.ecust.edu.cn`) in Shanghai, China on April 2, 2001.

As an illustration of a typical TCP transmission, Figure 1.1 shows the change of the congestion window and the corresponding end-to-end round-trip packet delays for a TCP connection between a host (`cw.crhc.uiuc.edu`) in Urbana, USA, and a remote echo server (`www.ecust.edu.cn`) in Shanghai, China on April 2, 2001. In this connection, the sender process sent a burst of 64 packets to the remote echo server, and the receiver process continuously read the TCP packets bounced back and recorded packet delays. Figure 1.1a shows that at the sender side, the congestion window is constantly changing, alternating between increasing phases and decreasing phases. As the congestion window defines the amount of data, in segments, that a TCP connection can transmit without waiting for their acknowledgements, the change of its size led to dynamic sending rates, which in turn resulted in irregular arrival patterns of incoming packets, as plotted in

Chapter 1

Introduction

1.1 Motivations

As the Internet is experiencing explosive growth, real-time delivery of multimedia data, such as image and video, is becoming more popular. The current Internet supports two kinds of transport services: reliable transfers by TCP and lossy transfers by UDP.

TCP, known as the Transmission Control Protocol, is a connection-oriented, reliable end-to-end stream service. Although TCP has proven to be essential to many networking applications, such as *ftp, telnet* and *http*, it is not suitable for delivery of real-time data for two reasons.

First, real-time images and video sequences, unlike conventional data and text, do not need to be precise. Human perception can tolerate a certain degree of inaccuracies in high frequency areas; hence, retransmissions of packets that contain visually insignificant information are a waste of network bandwidth, which is already very precious in image and video communications.

Second, TCP transfers have long and variable transmission delays. To enable reliable transfer, TCP incorporates a set of sophisticated congestion control and retransmission

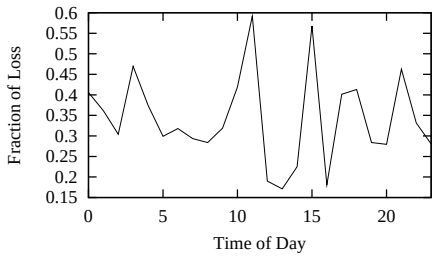


Figure 1.2: Loss rates of sending 2000 UDP packets from a host (`cw.crhc.uiuc.edu`) in Urbana to a remote echo server (`www.ecust.edu.cn`) in Shanghai, in a 24-hour period on April 2, 2001.

If image or video data is transmitted in its raw RGB or YCbCr format, packet losses will only result in errors in their corresponding areas. However, transmission of images and video sequences in their raw formats is too costly. For example, sending a CIF format (352 by 288) video clip at 30 frames per second requires a bandwidth of 70 Megabit per second. Considering the speeds of current modems, cable modems and ISDN lines, this is certainly infeasible. Therefore, efficient compression algorithms are normally applied to reduce the bandwidth consumed.

On the other hand, compression does its job by removing inherent spatial and temporal redudancies from data, making the compressed bitstream extremely vulnerable to packet losses. Due to the use of temporal-difference coding and variable-length entropy coding, isolated packet losses may result in a cascade of losses in a bitstream. Figure 1.3 illustrates the propagation effects. In this experiment, we first coded ten frames of the *missa* sequence, in CIF format and of size 352×288, using H.263, a uniform quantization factor of 10 and a coding pattern of IPP$\cdots$P. Next, we put the coded bitstreams into packets no greater than 536 bytes (to avoid fragmentation in the Internet) based on

Figure 1.1b: sometimes packets arrived in a burst, while at other times, packets were separated by long duration of delays. From Figure 1.1b, we can also see that the end-to-end round-trip delays of these 64 packets are quite long, due to three retransmissions at 4s, 9s and 19s.

Variable end-to-end delays will limit the performance of real-time video applications, since in such applications, video frames need to be played at prescribed constant time intervals. Those packets that do not arrive in time for playback are discarded and considered lost, leading to degraded playback quality.

Long transmission delays are also annoying for image transfers. Currently, images are primarily transmitted using TCP. From our Web surfing experience, we know that image downloading can be rather slow sometimes, especially when the network is congested. On the other hand, we may not be interested in all the details of an image and would rather have an approximate representation in a shorter time.

The above analysis indicates that TCP is not a good choice for the delivery of real-time images and video data. In contrast, UDP transmissions are found to have shorter delays and smaller transmission jitters, which explains why real-time applications normally resort to UDP for transmission. However, UDP is subject to packet losses because it is simply a de-multiplexing service built on top of packet-switched, best effort, and unreliable IP networks. UDP packets may get delayed, duplicated, or dropped during their delivery. Figure 1.2 illustrates the loss rates of 2000 UDP packets, sent at a rate of 50 packets per second, for the same connection between Urbana and Shanghai, in a 24-hour period on April 2, 2001. The loss rates for some part of the day can be as high as over 50%, and range from 17% to 47% at other times.

groups of blocks (GOB, a basic synchronization unit in H.263 coding, to be discussed in Chapter 4). This resulted in 22 packets with the first 13 packets containing GOBs from the I-frame, and each subsequent packet containing a P-frame. Finally, we decoded it, assuming that the packet carrying the 7th GOB of the I-frame was lost and the remaining 21 packets were received correctly. The effects due to the lost blocks could be seen in all subsequent frames (due to space limitations, we only show frames 3, 6 and 9), although packets carrying the coded information of the remaining frames were received correctly. Therefore, due to loss propagation, the effects of packet losses can be long-lasting and visually annoying.

Figure 1.3: Decompressed frames 0, 3, 6 and 9 of the *missa* sequence, assuming the 7th group of blocks of frame 0 was lost.

In summary, real-time transmissions of compressed image and video data may result in information loss that needs to be concealed effectively for high-quality playback.

1.2 Problem Statement

The objective of this thesis is to *design, analyze and evaluate end-to-end robust coding, error concealment and delivery schemes, in order to allow reliable and timely delivery of image and video data over the Internet.*

This topic is challenging as it involves multi-disciplinary research from both networking and signal processing areas. Specifically, our work is to divided into the following three major tasks.

1. Study of Internet traffic characteristics. As we are concerned with transmissions over the Internet, we need to first characterize properties of the two available transport services on the Internet: TCP and UDP. How large are TCP end-to-end packet delays and transmission jitters? What are typical loss rates of UDP packets? Are UDP packet losses random or bursty? The answers to the above questions will be used as guidelines in the following tasks.
2. Design of coding, error concealment and delivery strategies. Since we are dealing with compressed images and videos, such schemes are closely related to the type of coding algorithms used. Although there are many other competitive coding techniques in the literature, like fractal-based coding [39, 88], model-based coding [19, 51], and object-based coding [68, 74], block transform-based coding is by far the most popular for video compression, and subband-based coding is the best for image compression. Therefore, we develop coding, error concealment and delivery schemes based on these two coders.

3. System evaluation. To test the performance of our schemes, we need to build a prototype system that simulates the transmission process. A challeging question is how to do fair comparisons of different error concealment techniques under the same network conditions, using this prototype.

1.3 Problem Scope

As we all know, real-time transmissions of image and video data belong to a broad research topic, and we need to define the scope of our study clearly in the first place. In this section, we discuss a number of assumptions we have made regarding to the type of losses to be handled, the type of networks for transmission, the type of test images and videos, and the methods for performance evaluations.

The type of networks we are interested in are like the current Internet, which is a packet-switched, best effort network without guarantees on its quality of service (QOS). We also assume no priority support for different types of services.

The kind of losses we concentrate on are packet losses, because these are the form of losses most frequently encountered in Internet transmissions. Although there may be bit errors due to interferences in a transmission media, some of them are recoverable at the data link layer using forward error correction (FEC) codes, and the rest of them that fail the checksum tests are converted to packet losses, since they are simply dropped by end hosts. We study schemes for concealing packet losses with rates commonly seen in Internet transmissions, *i.e.*, from a few percent to 50%. From our Internet experiments, we know that only 5-6% connections are with consistently high percentage of loss rates that go above 50-60%; therefore, such infrequent cases are not considered in our study.

Table 1.1: Image formats supported in H.263.

Picture Format	Pixels per Row	Pixels Per Column
sub-QCIF	128	96
QCIF	176	144
CIF	352	288
4CIF	704	576
16CIF	1408	1152

The test videos and images to be used are of the formats supported by their coding algorithms.

The five video resolutions supported by H.263 are as listed in Table 1.1. The number of pixels per row and per column in the table refer to the dimension of the luminance component (also called the Y component), and the dimensions of chrominance components (Cr and Cb) are downsampled by two for both rows and columns. Among the five formats, 4CIF and 16CIF are meant for dedicated high bandwidth transmissions, QCIF and CIF are commonly used for transmissions in low bandwidth environment, and sub-QCIF is included for compatibilty reasons. Since we only study Internet transmissions, whose bandwidth is limited, we focus on QCIF and CIF formats. The test video sequences, along with their characteristics, are shown in Table 1.2. These videos are carefully chosen to cover videos with various degrees of motion, so that our algorithms are not biased towards just one kind of videos.

Similarly, we have selected four images to be used in our experiments: *lena*, *barbara*, *goldhill* and *peppers*. As evident in Figure 1.4, *peppers* is a smooth image; *lena* is mostly smooth with edges around the hair region; *barbara* has some texture information; and *goldhill* has many details. Again, they are chosen to represent images of different

a) *peppers* b) *lena*

c) *barbara* d) *goldhill*

Figure 1.4: Four test images used in our experiments.

Table 1.2: Test videos and their characteristics.

Video Sequence	Format	Frames	Amount of Motion
missa	CIF	150	low
boxing	CIF	400	medium
football	CIF	97	high
akiyo	QCIF	300	low
coastguard	QCIF	300	medium
river	QCIF	300	high

characteristics. All these images are of size 512 by 512 in order to facilitate recursive decompositions of subband transforms.

Image and video quality can be assessed both subjectively and objectively. Undoubtedly, the best and ultimate judges of image and video quality are human observers. Recently, ITU Recommendation P.910 [31] standardized the subjective assessment methods for multimedia applications. In the tests, a series of sequences is shown to a number of selected viewers in a controlled environment. Each viewer gives a quality score, called absolute category rating (ACR), or an annoyance scale, called degradation category rating (DCR), on the video. The five-point scales used for ACR and DCR are listed in Table 1.3. The average of ACR or DCR values, obtained from all the viewers, characterizes the overall subjective quality of the video sequence by MOS or DMOS. In a pairwise comparison method, the distortion index (DI), *i.e.*, the difference between the two scores given to the reference and test videos, is calculated and considered as the subjective measure of the test video. From the above description, one can find that subjective evaluations are time-consuming and not repeatable. In contrast, objective measures are often analytically tractable and reproducible. For images and videos, the most commonly used

Table 1.3: Five-point scale of ACR and DCR.

	ACR (quality)	DCR (impairment)
1	bad	very annoying
2	poor	annoying
3	fair	audible, but not annoying
4	good	slightly audible
5	excellent	inaudible

objective measure has been the peak signal-to-noise ratio (PSNR), which is defined as follows.

$$PSNR = 10 \log \frac{255^2}{\sum_i (x_i - \hat{y}_i)^2}, \tag{1.1}$$

where x_i and $\hat{y}_i$ are, respectively, the original and the reconstructed pixel values. The definition indicates that the larger the PSNR value is, the better will be the image quality. However, PSNR values do not always correlate well with subjective quality evaluations. Different sequences characterized by the same PSNR often show quite different subjective quality. For example, some additive noise in bright areas or texture regions is hardly noticeable, but the same noise in dark and smooth regions turn out to be visually annoying. This behavior can be explained by considering that human visual systems have different sensitivity to imperfections happening in various areas, but PSNR takes equal weightings of distortions from every pixel in a frame, regardless of its intensity level, and spatial and temporal frequencies.

Since there do not exist commonly accepted objective measures that can reflect subjective quality, PSNR is still widely used despite its drawback. In this thesis, we use PSNR as our quality measure.

1.4 Contributions of This Thesis

In this research, we have studied related issues on both coding and transmissions, in order to enable reliable and timely image and video transmissions over the Internet. Specifically, there are four major contributions of this thesis.

First, we have carried out a series of experiments to characterize Internet traffic. As we are interested in all sorts of Internet connections, short-distanced or long-distanced, domestic or international, heavy-loaded or light-loaded, we need to evaluate a large number of sites around the world. Since it is impossible to have previleged access to every site we are interested in, our workaround is to send packets to remote echo ports and collect packet statistics from packets that are "echoed" back. For a fair comparison between TCP and UDP statistics through echo port experiments, we have modified the Linux kernel to eliminate the inter-dependent windowing control in TCP transmissions. Based on the results collected using the above setup, we have arrived at two conclusions: a) TCP is not good for real-time image and video applications due to its large transmission jitters and long end-to-end packet delays. b) UDP packet losses are observed commonly on all kinds of connections but usually happen in small bursts. These observations motivate us to resort to UDP for both image and video delivery and develop error concealment algorithms in order to handle UDP packet losses.

Second, we have developed various error concealment techniques for H.263 coded videos. A careful analysis of the problem indicates that there are three kinds of losses present in H.263 videos, namely, bitstream losses due to packet losses, propagation losses due to temporal-difference coding and variable-length entropy coding, and compression losses due to lossy quantization. For bitstream losses, we have proposed to use a multiple-description coding method to facilitate the reconstruction of such losses. In a sender, we

have applied an *optimized reconstruction-based discrete cosine transform* (ORB-DCT) with an objective of minimizing the mean squared error, based on the reconstruction method used at the receiver. In a receiver, we have used a simple interpolation-based algorithm, as sophisticated concealment techniques cannot be employed in real time. The novel idea behind our approach is to design coders at the sender for each description using a joint sender-receiver approach, taking into account the reconstruction process at the receiver. For propagation losses, we have proposed to modify the motion compensation part of the decoder so that the reference frame is more accurate in case of packet losses. Further, we have also adopted a syntax-based packetization strategy in order to effectively reduce propagation losses among packets. For compression losses, we have studied an *artificial neural network (ANN)-based reconstruction* in order to compensate for losses introduced in multiple-description coding. ANNs are used because compression is a highly nonlinear complex process, and ANNs have been shown effective in modeling nonlinear behavior. Experimental results show that our proposed algorithms perform well in real Internet tests.

Third, we have studied rate control and allocation problems in H.263 coding, with the objective of minimizing reconstruction losses. Although a lot of rate-control and allocation schemes for single-description systems can be found in the literature [18, 29, 43, 49, 60, 61, 66, 72], there exist no such schemes that take into account the reconstruction process at receivers. We have studied this problem in a top-down fashion. At the GOB level, we have verified empirically that rate-distortion relationships after reconstruction are still convex, so that existing "constant-slope methods" based on the convexity assumption can still be applied, despite the introduction of the reconstruction process. At

the macroblock level, we have studied the design of reconstruction-based quantization matrices.

Last, we have investigated issues involved in the delivery of subband-coded images. Transmissions of subband-coded images on the Internet involves a tradeoff between time and quality by using either the TCP or the UDP protocol. Delivery by TCP gives superior decoding quality but with very long delays when the network is unreliable, whereas delivery by UDP has negligible delays but with degraded quality when packets are lost. Although images are delivered primarily by TCP today, we have proposed in this thesis the use of UDP and combined TCP/UDP for delivering MDC coded images. To generate such images, we have applied to each image an *optimized reconstruction-based subband transform* (ORB-ST), which is derived similarly as in the H.263 case, with an objective of minimizing the mean squared error, and by taking into account the reconstruction procedure. Internet experiments have been conducted to evaluate the time and quality tradeoffs of delivering MDC coded images using TCP, UDP, and combined TCP/UDP.

1.5 Outline of the Thesis

The rest of the thesis is organized as follows. Chapter 2 discusses related work in both image and video transmissions. Chapter 3 describes our experiments conducted to characterize the Internet traffic. The conclusions drawn from this study are used as guidelines for later chapters. Chapter 4 presents error concealment schemes to cope with various kinds of information loss, and Chapter 5 studies reconstruction-based rate control and allocation schemes, both for H.263-based video coding. Chapter 6 presents error concealment and delivery mechanisms for subband-coded images, and finally Chapter 7 concludes the thesis.

Chapter 2

Previous Work

The real time coding and transmission of images and video sequences is an active area of research, with efforts from both the signal processing and networking communities. While researchers in signal processing have focused on designing robust coding and error concealment schemes, those in networking have studied transmission-related issues. In this section, we overview existing work in these two aspects.

2.1 Robust Coding and Error Concealment Schemes

Depending on where error concealment is performed, existing schemes in the literature can be classified as *sender-based*, *receiver-based* and *sender receiver-based*.

2.1.1 Sender-Based Schemes

In this category, a sender usually employs certain techniques to improve the error resilience of a bit stream sent over a network.

Robust entropy coding (REC) decreases the effects of error propagation when transmission errors result in wrong decoding states during variable length decoding.

The first way of achieving REC is to periodically insert synchronization codes in a bit stream [20, 38, 41]. Undoubtedly, this adds additional overhead in bit-rate, which is determined by the length of synchronization code words. Although a shorter synchronization code word adds less redundancy, it increases the probability of false synchronization caused by bit errors. Therefore, in popular video coders, such as MPEG-2 and H.263, relatively long synchronization codes are used, depsite of the redundancy introduced.

A second scheme is to distribute code words from individual blocks into slots of equal size [35, 58, 73]. Initially, each slot is occupied by data from a designated block either partially or fully. Then, for those slots that are only partially occupied, a predefined sequence is used to search for blocks that are longer, than the slot size in order to fulfil the remaining space. By doing this, it converts variable length codewords into fixed-length structures so that decoding can automatically find the beginning of these structures and start decoding them in case of transmission errors, without adding synchronization code words. However, in this approach, a proper slot size is hard to determine, because a large slot size leads to wasted bandwidth and a small slot size results in very sophisticated block distribution and reassembly procedures.

The third technique is included in the error-resilient mode of MPEG-4 [68] in which a reversible variable-length code (VLC) is employed in such a way that, once a synchronization word is found, the coded bit stream can be decoded backwards until the first decodable code word after the corrupted data. This approach effectively eliminates error propagations between an errorneous code and its next synchronization; however, it achieves this robustness with reduced coding efficiency, because the code words used in reversible VLC have to be longer than those in regular VLCs in order to allow reversible decoding.

Finally, a recent technique [26] is proposed to increase the error tolerance of coded images by designing resynchronizing variable-length codes (RVLC). The authors demonstrated that RVLC can combat both bit errors or packet losses with less channel coding is needed. However, they also pointed out that overhead is non-trial for the RVLC-wavelet images and low-rate RVLC-JPEG images.

Restricting prediction domain (RPD) attempts to reduce picture-quality degradations due to temporal-difference coding. A good example is the *independent segment decoding* (ISD) mode in H.263 [15, 87]. In ISD, a segment is normally defined as a GOB or a number of consecutive GOBs with empty GOB headers, and treated as if it is an independent picture. In such a case, motion estimation and compensation are only carried out within segment boundaries; therefore, losses inside a segment will not propagate outside, and similarly losses outside a segment will not corrupt data inside the segment. This scheme, however, decreases the adverse effects of error propagation, at the expense of reduced coding gains and increased complexities of encoders.

Source coding and channel coding (SCCC) improves error-resilience of bit streams by adding error correction codes [7, 50]. Its distribution of protection is closely related to the source coder output. For example, I frames are guarded by more protection bits in H.263 alike coders [7]. The problem with this approach is that the error protection codes have to be designed to accomodate the worst channel condition (an over-pessimistic assumption), which implies the need for a very powerful code, leading to unnecessary wasted bandwidth. In addition, operating outside the assumed worst channel condition can result in total failure in recovery.

Layered coding (LC) can generate error resilient bit streams for networks with priority support [9, 21, 55, 95]. In layered coding, data is partitioned into a base layer and a

few enhancement layers. The base layer contains visually important video data that can be used to produce video output of acceptable quality, whereas the enhancement layers contain complementary information that allows higher-quality video data to be generated. In networks with priority support, the base layer is normally assigned a higher priority so that it has a larger chance to be delivered error free when network conditions worsen. Layered coding has been popular with ATM networks but may not be suitable for Internet transmissions for two reasons. First, the Internet does not provide priority delivery service for different layers. Second, when the packet-loss rate is high and part of the base layer is lost, it is hard to reconstruct the lost bit stream since little redundancy is present.

Multiple description coding (MDC), in contrast, aims to improve the robustness of coded bit streams for networks without priority support. It divides image or video data into equally important streams such that the decoding quality using any subset is acceptable, and that better quality is obtained by more descriptions. It is assumed in MDC that the probability of losing all the descriptions is small.

MDC was first implemented by the approach of scalar quantizers [10, 79, 80, 81], in which two side-scalar quantizers were applied to produce two descriptions. In order to minimize reconstruction errors when both descriptions were received, it then mapped a proper subset of index pairs formed from side quantizers to central-quantizer intervals. The difficulties with this approach are that optimal index assignments are hard to achieve in real time, and that suboptimal approaches, such as A2 index assignment [79], introduce a large overhead in bit rate [85]. A recent approach for subband coded images produces two descriptions by choosing one index assignment per subband and by encod-

ing explicitly this choice as map bits [63, 64]. However, the complexity involved in both coders and decoders may make it not attractive for low-delay transmissions.

An alternative implementation is by pair-wise correlating-transform (PCT) [47, 84]. Instead of putting each pixel in every description, PCT introduces correlations in each pair of transform coefficients and distributes the two resulted coefficients into two descriptions. This approach has high coding efficiency when both descriptions are available but has mediocre reconstruction quality with one description. It is, therefore, not applicable in an error-prone environment like the Internet because the ultimate perceived quality may be dominated by the reconstruction quality of one description.

The above two approaches either require an optimal assignment of quantizer indexes or sophisticated statistical estimations of coded coefficients; therefore, such complicated encoding and decoding algorithms make them infeasible for real-time delivery.

2.1.2 Receiver-Based Schemes

Receiver-based postprocessing techniques are motivated by the insensitivity of human perception to high frequency components. The processing is carried out in either the spatial domain, temporal domain, frequency domain or some combinations of the above.

Spatial domain recovery makes use of the smoothness assumption of video signals through a minimization approach. One approach recovers a lost block by minimizing the sum of squared differences between the boundary pixels of the lost block and its surrounding blocks [86, 96]. This smoothness measure normally leads to blurred edges in the recovered image. The other approaches propose to minimize the variations along edge directions or local geometric structures [37, 70, 94]. They require accurate detection of image structures, and mistakes can yield annoying artifacts in the reconstructed video.

Temporal domain recovery exploits temporal correlations by replacing a corrupted block by its correponding block, which is on the motion trajectory in the previous frame [23, 36]. The difficulty with this approach is that it relies on the knowledge of motion information, which may not be available in all circumstances.

Frequency domain recovery performs reconstruction by interpolating each lost coefficienct in a damaged block from the corresponding coefficients in its four neighboring blocks [6, 28, 27]. Because the correlation of pixels in adjacent blocks is likely to be small, the interpolation does not produce satisfactory results.

Other approaches employ some combinations of the above three techniques to reconstruct lost data. The approach based on maximum smooth recovery extends the smoothness property to both spatial and temporal neighbors [95]. The projection onto convex sets (POCS) [71, 93] formulates spatial and temporal smoothness constraints into convex sets and derives a solution iteratively. The approach using genetic algorithms conceals a corrupted block by iteratively performing reproduction/crossover/mutation operations and evaluating a proposed fitness function until a stopping criterion is satisfied [67]. Besides computationally expensive, designing postprocessing at the receiver independent of the encoder at the sender may not result in high-quality reconstruction because the two are usually closely related.

2.1.3 Sender Receiver-Based Schemes

In the following schemes, the sender and the receiver cooperate in the process of error concealment.

Retransmissions have been generally considered inappropriate for real-time streaming applications because of delays introduced. To make use of retransmitted data, a naive

decoder will wait for the requested retransmitted packets before playing subsequent data, leading to long freezes in the playback. In more complex decoders, the lost video part is first concealed by a certain recovery method, without waiting for the retransmitted packets [22]. At this point, error concealment introduces certain degree of inaccuracies in the video data, which will propagate in the playback. Upon the arrival of the retransmitted data, the affected pixels due to error propagation will be corrected according to a complex relationship. This approach, however, is difficult to handle cascaded loss scenarios, in which packet losses happen again before the arrival of retransmitted packets. An alternative work [59] reduces the adverse effects of long retransmission delays by rearranging temporal dependencies of frames in such a way that a displayed frame will be referenced in the decoding of its subsequent dependent frame at a later time. As a result, even if a retransmitted frame comes in late for playback, it can still be used in the decoding of its dependent frames. The difficulty with this approach is that the distance between the reference and dependent frames is hard to choose, and this distance may need to be adapted even within a connection.

Joint source channel coding (JSCC) approaches [16, 45, 46, 82] minimize transmission errors by jointly designing the quantizer and the channel coder, according to a given channel error model and feedbacks from receivers. To cope with noisy channels, they try to optimally partition bandwidth between the source and the channel coders depending on the channel loss status, normally characterized by some parameters. They, however, are hard to apply in the Internet, since the Internet does not have a well-defined channel model.

Interleaving with reconstruction [53, 78] is a very simple MDC-based approach that involves efforts from both senders and receivers. In a sender, it partitions adjacent pix-

els in different descriptions and assigns them to distinct packets for transmission. In a receiver, the lost pixels are approximately reconstructed using their surviving neighbors. This simple approach can give good reconstruction quality in lossy networks, because adjacent images pixels generally have high correlations. However, in this approach, the generation of multiple descriptions in the sender side is done independent of the reconstruction operations in the receiver side, leading to suboptimal performance since these two operations are usually correlated.

Sender receiver-based schemes are usually more effective than either sender-based or receiver-based schemes, because a sender can exploit the characteristics of receivers and feedback information to better adapt its control. Among the three approaches, *interleaving with reconstruction* is a good choice to improve the error resilience of coded bit streams to be transmitted on the best-effort Internet. However, as discussed before, its drawback lies in the separation of multiple description generation in senders and the reconstruction process in receivers. In this thesis, our goal is to develop error concealment algorithms in senders, with the objective of optimizing reconstruction quality, based on the reconstruction method used in receivers.

2.2 Transmission Schemes

Besides coding, there are also many challenging issues in the transmission part. Image transmission is a relatively simple process, because it does not involve real-time and bandwidth constraints; hence, currently it is treated as normal text data and delivered by TCP. On the other hand, video transmissions are more complicated; therefore, it has spurred interests from both research and industrial communities. In the following, we describe both research efforts and industrial standards related to video transmissions.

2.2.1 Research Efforts on Video Transmissions

In streaming video on the Internet, one needs to address the following three important issues. First, due to the heterogeneity and the dynamic nature of the Internet, connections to various sites at different times can have very different bandwidth. Hence, how to adapt the transmission rate based on the available bandwidth greatly affects the playback quality. Second, using a best-effort delivery service like IP, how to recover from packet losses? Last, as illustrated in Chapter 1, losses in coded bit streams tend to propoagate across frames. Hence, the design of packetization schemes for minimizing loss propagation effects becomes an important topic. In the literature, there have been some efforts dedicated to each of the above three problems.

Rate-Based Control. In [12], Cen *et al* employs a feedback loop between the client and the server to control the transmission of MPEG video data over the Internet. In this scheme, a client constantly sends feedback information to a server, and the server adapts its sending rate based on the client's feedback about the network condition. The proposed adaptation method, however, is very primitive, which is simply based on scaling frame rates. Vosaic [14], the first Web based continuous media system, uses a more sophisticated adaption scheme by exploiting the relative importance of coded streams. In Vosaic, rate adaptation is achieved by stream thinning. For example, an MPEG coded pattern IBBPBB at 30f/s can be trimmed to be at 20f/s, 10f/s and 5f/s with coding patterns IBPB, IP, and I, respectively, with selected B and P frames dropped. A second motivation behind rate adaptation is the need for fair resource sharing among multimedia flows and regular TCP flows, which is the dominant source of Internet traffic. One way to ensure fair competition with TCP flows is to adopt the same adaptive window

control algorithm of TCP [32]. In this case, the sending window size is usually reduced by half when a single packet loss is detected, resulting in abrupt changes of sending rates and jerky video playbacks. Instead of matching the TCP traffic pattern exactly at every time instance, a more relaxed form used in [75] is to match the average TCP throughput over a period of time. Such multimedia flows that either strictly or approximately follows TCP traffic patterns are considered as *TCP friendly*.

Retransmission-Based Loss Recovery. The common feature of this class of algorithms is that they rely on retransmissions to recover from packet losses. In cyclic-UDP [69], video frames are stored in a buffer in the order of their importance. For example, MPEG I frames can be placed at the front of the buffer, P frames, behind I frames, and B frames, at the end of the buffer. During an allocated transmission period, lost frames are retransmitted cyclicly in the order of their importance. The ordering of frames ensures that more important frames have more chances to get through. One drawback of this approach is its excessive bandwidth usage. To alleviate this problem, the Video Datagram Protocol (VDP), proposed in conjunction with Vosaic, only retransmits lost frames upon requests from receivers. A receiver only asks for the retransmission of a lost frame when it finds that the current bandwidth and its own computing resources are sufficient; otherwise, the lost frame is simply skipped. However, the above two schemes do not address the issue of extra delays introduced by retransmissions and whether the added delay would render retransmitted packets useless. The two approaches described in Section 2.1.3 address this issue by either rearranging the temporal dependencies of frames so that a retransmitted frame can be used for the decoding of a later frame, or employing some complex relationship to incorporate information from retransmitted frames into the frame that is currently decoded.

Packetization Schemes to Prevent Loss Propagation. In the sender side, *intelligent packetization* schemes were developed to prevent two kinds of propagation losses. First, because of exclusive use of variable length coding (VLC) in compression standards, packet losses often cause the loss of synchronization in a coded bit stream, rendering subsequent packets useless. One technique proposes to divide a coded bit stream into packets according to inserted synchronization points [77]. If a synchronization unit (e.g., GOB in MPEG and H.263) cannot fit into a single packet, it is further divided according to smaller syntactic units (e.g., macroblocks). Second, most coding schemes rely on temporal-difference coding in order to achieve coding efficiency, thereby introducing a pervasive dependency structure into the bit stream. To prevent the propagation effects due to difference coding, dependency tree-based schemes put all the information derived from a common ancestor into a single packet [13]. In this approach, a lot of side information has to be sent in order to ensure the proper assembly of received packets in the decoder.

2.2.2 Industrial Standards for Video Transmissions

In this section, we discuss the following standards that support real-time video transmissions on the Internet: Resource ReSerVation Protocol (RSVP), Real-time Transport Protocol (RTP), Real Time Streaming Protocol (RTSP), and the ITU H.323 standard.

RSVP. The design of Resource ReServation Protocol (RSVP) [4] is probably the first effort to support real-time delivery of multimedia data over the Internet. Its design was cooperatively done by Xerox Corp.'s Palo Alto Research Center, MIT, and Information Sciences Institute of University of California. Basically, RSVP is the network control protocol that allows applications to request and reserve special end-to-end quality of

service (QoS) for its data flows. At connection setup, an application uses RSVP to request a specific quality of service in all the routers along its data path. In this phase, RSVP is reponsible for converting the end-to-end QoS requirements to the ones that need to be met at intermediate routers. After the reservation is made, RSVP is also in charge of scheduling packets and maintaining router and host states in order to provide the requested QoS.

There are many good features in RSVP. First, RSVP is receiver-oriented and is capable of handling heterogeneous receivers. In a network of hosts with different capabilities and QoS requirements, the receiver-oriented RSVP facilitates the handling of heterogeneous receivers by allowing them to choose their own levels of desired QoS, initiate reservations, and keep them active as long as needed. Second, RSVP is designed for both unicasts and multicasts. Since reservations are initiated by receivers, RSVP can easily handle the change of memberships and routes. Last, RSVP has good compatibility with existing IP protocols and is targetted to run over both IPv4 and IPv6.

Yet there are a number of limitations with RSVP. First, it does not provide mechanisms to prevent users from asking for more bandwidth than what is needed; therefore, network resources can be easily exhausted with "greedy" users. Second, it does not address the issue of user authentication. This can cause problems because if malicious users are granted their requests, qualified users will be starved. Last, perhaps the largest limitation of RSVP is that it is hard to deploy using existing network infrastructure. In order for RSVP to be set up, all the routers need to be modified in order to support the protocol, which is hard to accomplish in the near future. Therefore, RSVP is often regarded as the mechanism for future Integrated Services Internet.

RTP. As a complementary approach, Real Time Protocol (RTP) [2, 3] was proposed as an IP-based protocol providing support for the transmission of real-time data on the current Internet. RTP is designed to work in conjunction with the auxiliary Real Time Control Protocol (RTCP). In an RTP session, each media source sends packetized multimedia data to receivers, together with sequence numbers and timestamps. To help receivers detect losses, senders periodically generate sender reports (SR) that include the total number of packets and octets sent. To facilitate adaptation at senders, receivers also periodically send receiver reports (RR) that indicate current loss status, jitters, and highest sequence number received.

RTP is designed for a wide range of applications, both unicasts and multicasts of real-time data, and both one-way transport services such as video on demand and interactive services like video conferencing. Therefore, it is now widely used by many commercial products to support real-time video transmissions.

However, RTP, by its nature, only provides feedback and synchronization information to applications. Many important issues, such as rate adaptation, loss detection, and recovery have to be addressed in applications. In other words, RTP alone will not enhance the quality of image and video transmissions, but will need cooperative efforts from applications.

RTSP. It was jointly developed by RealNetworks, Columbia University, and Netscape Communications based on the streaming experiences of RealNetworks' RealAudio and Netscape's LiveMedia [5]. RTSP is essentially a client-server multimedia presentation protocol to enable the controlled delivery of streamed multimedia data over IP network. Its main functionality is to provide "VCR-style" remote control for multimedia streams, such as pause, fast forward, reverse and absolute positioning, while common data flows

are still encapsulated in RTP packets for transmission. We need to keep in mind that RTSP is merely a presentation protocol that is meant to enhance the look and feel of multimedia playback.

H.323. This is a conferencing framework, standardized by the International Telecommunications Union (ITU) [1] that specifies the components, protocols and procedures in order to provide multimedia communication services over packet networks. It is targetted for peer-to-peer, two-way interactive delivery of audio and video data. One of its important objectives is for it to work well for a moderate number of participants and be able to interact well with existing standard phone or Internet phone gateways. Again, similar to RTSP, H.323 is only a presentation and control protocol for video conferencing applications.

Discussions. RSVP is a protocol defined for future Internet that has the underlying support for integrated services. For the current Internet, people normally resort to RTP for the delivery of real-time multimedia data. Both RTSP and H.323 are protocols that sit on top of RTP, defining additional control operations for their intended applications. In fact, they are complementary in functionalities: H.323 is designed to enable audio video conferencing in moderately sized peer-to-peer groups, whereas RTSP is useful for large-scale broadcasts or on demand streaming aplications. Both H.323 and RTSP use RTP as their standard mechanism to deliver multimedia data. This data-level compatibility makes it very easy to convert between RTSP and H.323 in gateways, since only control information needs to be taken care of.

As said above, all these industrial standards used alone cannot improve the quality of image and video transmissions, if applications fail to provide techniques to cope with

network losses and adapt to dynamic networking conditions. In addition, these transmission protocols do not have mechanisms to exploit the internal characteristics of individual image and video coders; therefore, they are not necessarily the most efficient for some applications.

The above discussions indicate that there have been active research activities and industrial efforts to support video transmissions over the Internet; hence, video delivery is a topic that has been well studied. On the contrary, there is little attention paid on image transmissions. On the current Internet, images are delivered primarily using TCP packets, although TCP transmissions are sometimes not desirable because of their aggressive rate control schemes. Therefore, in this thesis, we plan to study transmission schemes for images, but not for videos.

Chapter 3

Internet Traffic Study

This chapter presents traffic experiments that were carried out to simulate both image and video transmissions over the Internet. These two kinds of transmissions have different characteristics. For video transfers, data are sent periodically at the rate that is commensurate with the frame rate, and we are more interested in whether these packets can arrive in time for playback, *i.e.,* whether the average end-to-end packet delays and transmission jitters are small enough to support real-time playback. For image transfers, data are normally sent in a big chunk on request, and we are more interested in its response time, *i.e.,* the aggregate delay between the initiation of a request and the arrival of the last received packet. We compare TCP and UDP performances based on these measures for both cases. In addition, UDP packets can get delayed, duplicated or lost during transmission; hence, we also study the characteristics of packet losses for UDP in the above two cases.

The traffic experiments were done for two purposes. First, the conclusions drawn from the study are essential for the design of error concealment strategies and transmission protocols in this thesis. Second, the packet traces collected provide a good basis for fair

comparisons of different strategies. Applying the same trace to different schemes ensures that they are compared under the same network conditions. In this chapter, we also describe trace-based simulations used in later chapters.

3.1 Experimental Setup

To cover a variety of connections around the world, we chose both domestic and international sites in our experiments. The involved sites are listed in Table 3.1. The first three represent typical US connections, and the last three represent typical international connections to Japan, China and United Kingdom. Typically, domesitic connections have low percentage of losses (less than 10%); some international connections have medium losses (from 10% to 20%); and some international connections may have high loss rates (from 20% to 50%). Our choice of experimental sites covers these three situations: three domestic sites representing low-loss connections, the Japan and UK sites representing medium-loss connections, and the China site representing high-loss connections. In our

Table 3.1: The sites used in our traffic experiments.

Location		Host Name	IP Address
USA	California	`daedalus.cs.berkeley.edu`	169.229.62.38
	Virginia	`noc.neumedia.net`	208.192.16.2
	Texas	`infinity.ccsi.com`	216.236.168.10
Asia	Japan	`www.j-wave.co.jp`	211.2.255.147
	China	`www.shmu.edu.cn`	202.120.79.41
Europe	UK	`www.uea.ac.uk`	139.222.128.1

experiments, we found only 5-6% sites that produced higher than 50% losses; therefore, they were infrequent cases and were not considered.

Since we did not have previledged accesses to the above sites, we relied on their well known echo services to collect traffic statistics. First, we sent probe packets to the echo port of each server, and recorded the sending and receiving times of each packet from the packets "bounced" back. Last, we calculated the following traffic statistics: end-to-end packet delay, transmission jitter, total response time, and loss percentage, based on the temporal traces of probe packets. Besides providing a large variety of choices, sending packets to echo ports simplifies clock synchronization, since both sending and receiving times are measured on the same host.

Bouncing messages off the TCP and UDP echo ports of a remote server are meant to emulate a "hypothetic" TCP connection and an UDP path that are twice as long as the path from the source to the remote computer. However, their timing results are not comparable directly because TCP and UDP echo ports are implemented differently. Figure 3.1 illustrates the difference of TCP and UDP connections in this scenario. An UDP echo port reflects every incoming packet immediately after it is received, whereas a packet sent to a TCP echo port traverses two virtual links, each of which employs window-based flow control to accommodate packet losses and buffer overflows. Hence, a TCP echo port has a receiver window and a dependent sender window, since the echo port can only send a packet after all its previous packets have been received.

To avoid such reassmebly delay at a TCP echo port and make fair comparisons of TCP and UDP transmssions, we would like each TCP packet to tranverse a single virtual path as that of a UDP packet. Figure 3.2 shows how this can be done. In the sending part, the TCP packet involved in experiments (*e.g.,* from port x to port y of a local machine),

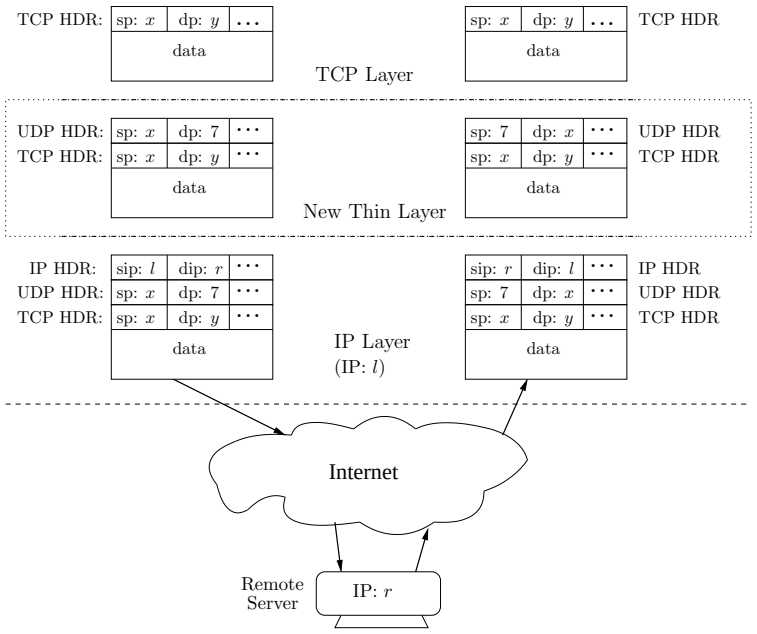


Figure 3.2: Conversion of TCP packets to UDP packets in echo port experiments.

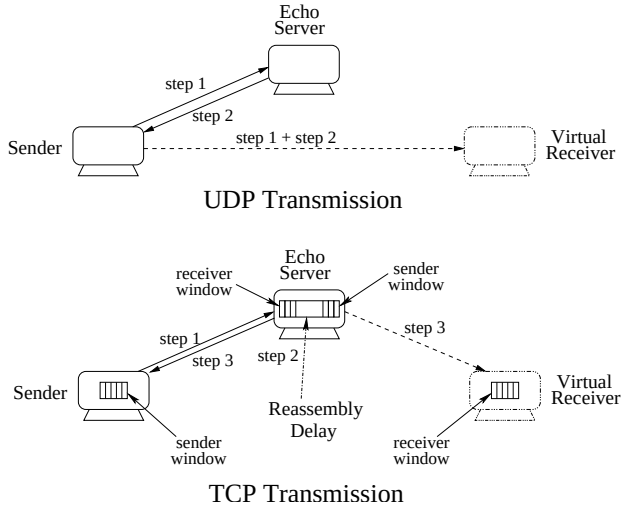


Figure 3.1: TCP and UDP transmissions to a remote echo server.

is converted into an UDP packet by prepending an UDP header to its TCP header, specifying that this is an UDP datagram destined for the echo port of a designated site. This encapsulated packet is passed down to the IP layer, which is responsible for finding the next hop based on the IP address of the designated destination. In the receiving part, the IP layer strips off the "false" UDP header after verifying that the packet is coming from the right remote address, and then delivers it to the correct port of the TCP layer.

Although the above idea is conceptually simple and clear, its implementation is quite involved, because we had to instrument the Linux kernel in order to fulfil the task. Specifically, we made the following changes to the RedHat 7.0 kernel.

- *Input connection information to the kernel.* Because the encapsulation process only takes place for those packets that belong to the connection involved in the traffic experiment, we need to inform the kernel the following parameters: the connection end points that we are interested in, and the remote IP address and port number that the packets are redirected to. A brute-force method is to hard wire these choices into the kernel, but then we need to recompile the kernel and reboot the machine each time we test a different destination site, making it too time-consuming. To save time, we modified the *socket* interface, so that the above parameters can be dynamically changed at run time by making the system call: *setsockopt()*.
- *Encapsulation of an TCP packet into an UDP packet.* This is the sender half of a thin layer between TCP and IP. To avoid extra buffer copying, we first reserved "enough" space while building the TCP packet; hence, the job simply involved preprending the UDP header to the TCP header.
- *Decapsulation of an UDP packet into an TCP packet.* This is the receiver half of the thin layer between TCP and IP. Normally IP de-multiplexes packets by protocols. Here, if the packet was an UDP packet coming from the designated IP address, we removed its UDP header, and delivered the packet to the local port based on its TCP header.
- *Maintaining correct socket status.* Although the OSI model clearly divides networking tasks into seven layers, the Linux networking implementation is far from modular, and has information "leakage" across various layers. The *sock* structure is used to maintain such information. In both the IP and transport layers, we had to

carefully set status variables in *sock* to prevent them from misinterpreted by other layers.

- *Checksum calculations.* In TCP and UDP headers, the calculation of checksums is based on the "pseudo header" that includes three fields from the IP header: length, source IP address, and destination IP address. To prevent packets from being silently discarded, we made sure that the checksums were computed over the correct information. In the sender side, each redirected packet has both TCP and UDP headers. To construct the TCP checksum, the source and destination IP addresses should be those of the local host. To construct the UDP checksum, the source IP address is the local IP address, and the destination IP is the remote server's IP address. In the receiver side, the UDP header checksum is verified by using the address of the echo server as its source IP address and its local address as its destination IP address, and the TCP header checksum is verified by using the local IP address as both its source and destination addresses.
- *Negotiation of maximum segment size (MSS).* During TCP's three-way handshake, the MSS of a connection is normally set to the value of the *path maximum transfer unit* (PMTU) that is supported by its network interface in order to avoid segmentation by the IP layer. In our experiments, the TCP connections involved in the study are between two local ports, not involving any network interfaces; therefore, their MSSs are set to very large values, resulting in fragmentation by the local IP, which is not desirable. Hence, we modified MSS negotiation in TCP so that the connections' MSSs are set to PMTU, although they appear to involve only local transfers from TCP's point of view.

- *Output of retransmission information.* To have a better understanding of TCP delays and jitters, we also modified TCP to generate information regarding retransmissions, such as the congestion window (*cwnd*), slow start threshold (*ssthresh*), and number of retransmitted packets per connection.

Using the above setup, we conducted traffic experiments to simulate both video and image transmissions. In simulating video transfers, we compared end-to-end packet delays and jitters of both TCP and UDP transmissions. In addition, we studied the loss behavior of UDP packets. In the case of image transfers, we compared the total response time of TCP and UDP, and also investigated the loss behavior of UDP packets.

3.2 Traffic Study Simulating Video Transfers

In this section, we describe the experiments that were done to emulate video transmissions. For this purpose, we periodically sent 500-byte packets to the echo port of each of the destinations in Table 3.1 at a rate of 50 packets per second, using both TCP and UDP. The aggregate data rate of 25 Kbytes per second is about that needed to send a slow-motion CIF (352×288) format H.263 video at 15 frames per second. From the packets echoed back, we then studied the end-to-end delays and jitters of both TCP and UDP transmissions, as well as the loss behavior of UDP packets.

3.2.1 Comparisons of TCP and UDP Packet Delays and Jitters

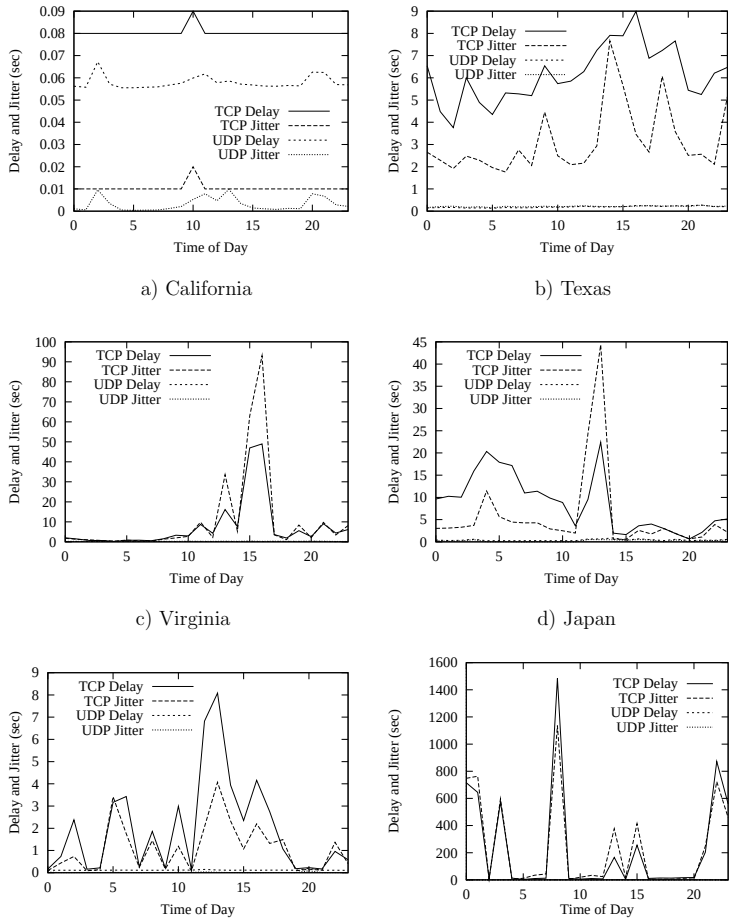
Video transmissions are always considered as "delay sensitive." Here, delay refers to not only network latency but also variations in latency, the latter defined as "jitter." Among the two measures, jitter plays a more important role in video playback quality. As a simple example, assume that each video frame can be placed in a single packet for

transmission. The receiver that needs to play the video at 30 frames per second expects packets to arrive every 33 millisecond. If a packet arrives too early, then it can simply be buffered by the receiver until its playback time. However, if a packet arrives too late, the receiver may be "starved," *i.e.,* it does not have the frame to update the screen. Therefore, the video playback becomes jerky and visually annoying.

There are two reasons that can explain why packets would have variation in delays. First, in a multihop packet-switched network, each packet may experience variable queuing time in each intermediate node. Second, transport protocols employed in end hosts can introduce additional variations in packet delays. Although we can do nothing about buffering delays at intermediate nodes, we can carefully choose the transport protocol at end hosts in order to reduce the adverse effects that jitters have on video quality.

In Figure 3.3, we compare the average end-to-end delay and jitter of packets transmitted by TCP and UDP, respectively. From the above, we have the following observations.

- In all cases, TCP packets have much greater jitters than UDP packets. For TCP, the large variance in packet delays is caused by its congestion and flow control mechanisms. For example, packets may arrive without retransmission, or with multiple retransmissions, and retransmissions can be triggered by timeouts or simply by fast retransmits, all these leading to large differences in TCP packet delays. On the other hand, UDP is simply responsible for multiplexing and de-multiplexing packets, introducing roughly the same amount of delay for each UDP packet.
- In the majority of cases, packet jitters vary greatly across different times of day, with the only exception of the Urbana-California connection. This makes the design of jitter buffers very difficult at receivers. If the design accomodates the worst



a) California b) Texas

c) Virginia d) Japan

e) UK f) China

Figure 3.3: Comparisons of packet delays and jitters of TCP and UDP transmissions for connections to various sites at 10 pm their local time on April 8, 2001.

scenario, it will introduce unnecessary initial buffering delays and unnecessary memory allocations at times when transmission jitters are not so bad. In addition, it is impossible to predict in advance what would be the largest jitter for a particular connection. The theoretical upper bound of the receiver buffer is the size of the video clip, but this is equivalent to downloading the entire video before its playback, which is overly pessimistic. On the contrary, UDP jitters are much smaller and also predictable, comparable to those of UDP packet delays.

One can conclude from the above analysis that TCP is not suitable for real-time video transmissions.

3.2.2 UDP Loss Behavior

From the above, we see that video transfers by TCP would give unacceptable quality since a majority of the packets would miss their deadlines and were discarded by applications. Therefore, one needs to rely on UDP for real-time video transmissions. However, UDP packets are subject to packet losses, leading to large degradations in playback quality if they cannot be recovered. In this section, we characterize the loss behavior of UDP transmissions on the Internet. Our goal is to find the interleaving factor that adjacent pixels should be separated by in order to make the probability of unrecoverable packet losses sufficiently small.

Figure 3.4 depicts typical CDFs of burst lengths of connections to the six test sites measured at 10 pm their local time on April 8, 2001. We can see that the number of consecutive packet losses is usually very small, of length 5 or less for the above six sites. In the graphs, we have stretched the y range to a value greater than one in order to allow clearer views of the CDF functions.

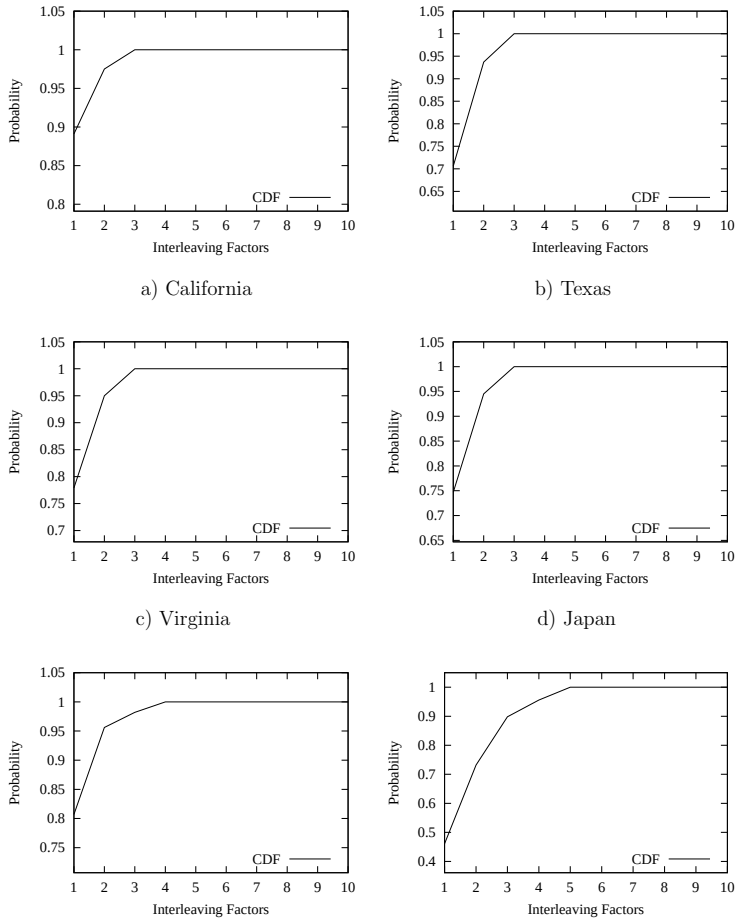


Figure 3.4: Cumulative distribution function (CDF) of consecutive packet losses in connections to various sites at 10 pm their local time on April 8, 2001.

The results imply that we can use small interleaving factors to convert bursty losses to random losses. However, the CDF of burst lengths alone is not sufficient to determine the interleaving factor because a bursty loss larger than the interleaving factor may span two interleaved sets and can be recovered from partial information received in each interleaved set. In general, an interleaving factor i allows reconstructions by interpolation of a bursty loss of $i-1$ packets or less if losses are from the same interleaved set, or of length in the range $[i,(2i-2)]$ when losses are from different interleaved sets. In order to conceal bursty losses most of the time, it is necessary to choose interleaving factor i so that the probability of packets that are not recoverable using i is small enough.

Let the total number of packets sent be n_p and the interleaving factor be i. Over all the interleaved sets, assuming that losses of j consecutive packets, $j \leq i$, happen m_j^i times, then the total number of packets lost is n_s (independent of i), where:

$$n_s = \sum_{j=1}^{i} j \times m_j^i. \tag{3.1}$$

Given that all the packets in an interleaved set are lost, $Pr(fail \mid loss, i)$, the conditional probability that the content of a packet cannot be recovered by reconstruction using interleaving factor i, can be derived from (3.1) as follows:

$$Pr(fail \mid loss, i) = \frac{i \times m_i^i}{n_s}. \tag{3.2}$$

$Pr(fail \mid i)$, the unconditional probability that a packet cannot be reconstructed in the stream received for interleaving factor i, can be computed as follows:

$$Pr(fail \mid i) = Pr(fail \mid loss, i) \times Pr(loss) = \frac{i \times m_i^i}{n_s} \times \frac{n_s}{n_p} = \frac{i \times m_i^i}{n_p}. \tag{3.3}$$

Figure 3.5 shows that $Pr(fail \mid loss, i)$ drops quickly with increasing interleaving factor i. For the Urbana-China connection, the loss rate can be as high as 50% for some part of the day, but the probability of not able to reconstruct a lost packet is held under 5% with an interleaving factor of 4. For connections to Texas, Virginia, Japan and UK, the loss rates are relatively lower, ranging from 0% to 20%, and the probability of failing to reconstruct a lost packet is less than 5% using an interleaving factor of 2. The connection to California is the best one, with all its lost packets recoverable using an interleaving factor of 2.

In summary, we can conclude that UDP packet losses happen in small bursts, and that they can be effectively recovered using small interleaving factors (2 or 4) for typical domestic and international connections.

3.3 Traffic Study Simulating Image Transfers

In this section, we describe the experiments that were conducted to emulate image transmissions. In contrast to video transmissions, image transfers involve sending a sequence of packets "continuously" to a receiver. While TCP packets are automatically spaced out by TCP's congestion and flow control algorithms, UDP packets are sent as fast as applications can. Now the question arises as to how "continuous" should UDP packets be sent in applications. Intuitively, if the number of packets exceeds either the sender or receiver buffer size, all the packets that cause buffer overflows will be discarded, leading to unnecessary losses observed by end users. Therefore, we need to space out the packets sent, without incurring unnecessary delays in the process. Our objective is to choose empirically a strategy to space UDP packets in order to eliminate sender or receiver buffer overflows, while incurring negligible end-to-end response times seen by users.

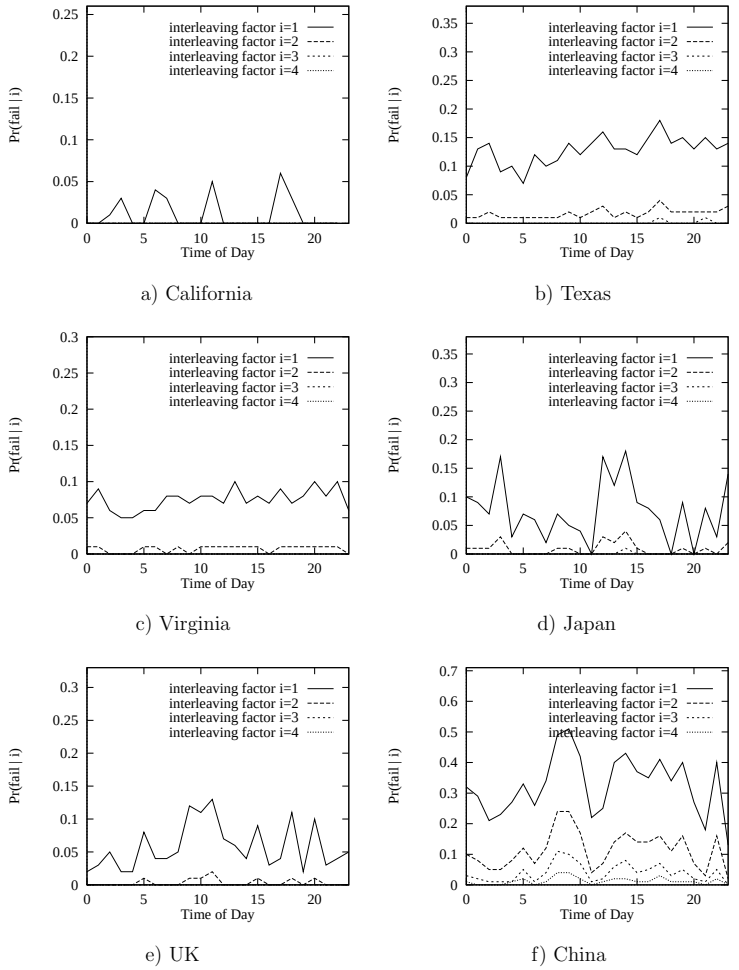
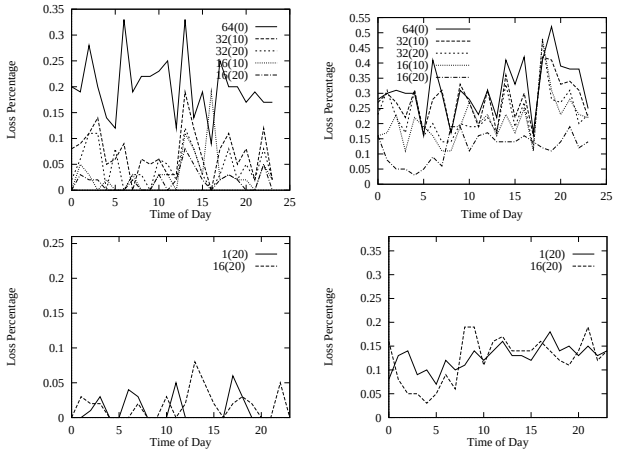
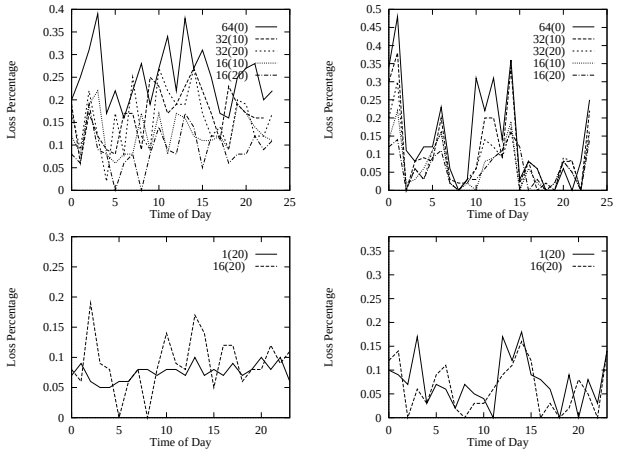


Figure 3.5: $Pr(fail|i)$, probability of bursty losses that cannot be recovered conditioned on interleaving factor i, at different times on April, 8, 2001.



a) California b) Texas

Figure 3.6: UDP loss percentages at various sending rates for connections to California and Texas, respectively.



a) Virginia b) Japan

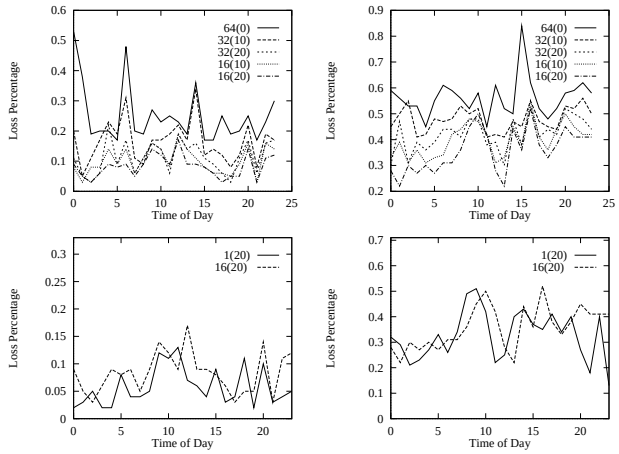
Figure 3.7: UDP loss percentages at various sending rates for connections to Virginia and Japan, respectively.

In the experiments, we sent 64 UDP packets of 512 bytes to remote echo servers, which corresponded to sending a 512×512 image coded at 1 bit per pixel. The transmissions were arranged in the following five ways.

1. 64(0) - All the 64 packets were sent continuously in a burst.
2. 32(10) - The packets were divided into two 32-packet groups, and the two groups were separated by a 10-millisecond interval.
3. 32(20) - The 32-packet groups were separated by a 20-millisecond interval.
4. 16(10) - Ajacent 16-packet groups were separated by a 10-millisecond interval.
5. 16(20) - Ajacent 16-packet groups were separated by a 20-millisecond interval.

Note that 10 milliseconds are the smallest timer interval supported by most UNIX operating systems. Here, we are trying to find a delivery strategy with the smallest overhead in response time.

Figures 3.6, 3.7 and 3.8 compare these five schemes for the six chosen sites. In each figure, the top two graphs compare UDP loss percentages of the five schemes. We can see that if we send packets in a burst, the packet loss rates can be rather high for even domestic sites. For example, the loss rate of the transmission to Texas at 7pm can reach as high as 50%, which is very unusual for short-distance connections. Most of the losses were caused by end-buffer overflows. After spacing out the packets sent by small intervals, packet losses are greatly reduced. In most cases, the 16(20) strategy produces the smallest loss percentages. The bottom two graphs in each figure compare the loss rates of the 16(20) stragety with those of periodic transmissions conducted in Section 3.2, in which packets were sent every 20 milliseconds. The graphs show that the two loss rates are comparable, which imply that the 16(20) scheme has effectively overcome the problem



a) UK b) China

Figure 3.8: UDP loss percentages at various sending rates for connections to UK and China, respectively.

of overruning sender and receiver buffers in transmission. Therefore, in the following experiments, we use the 16(20) scheme in arranging UDP transmissions.

3.3.1 Comparisons of TCP and UDP Response Times

As current image transfers are done primarily by TCP, we study the end-to-end delays of TCP transmissions, and compare them with those of UDP packets. For TCP, we used our modified Linux kernel to collect data, and for UDP, packets were sent with 20 millisecond intervals between every 16 packets.

Figure 3.9 compares TCP and UDP end-to-end delays for both international and domestic connections for a 24-hour period. From the graphs, we can draw two conclusions.

- The end-to-end delays of UDP transmission have far less variations than those of TCP transmission. They are almost constant compared to TCP times. For example, for transmission between Urbana and Texas (Figure 3.9b), UDP delays range from 0.3 to 0.5 seconds, whereas TCP delays take from 2 to 6 seconds. Therefore, UDP end-to-end response times are more predictable than TCP times.
- On average, TCP delivery is one to two orders slower than UDP delivery: TCP packets take from 4 to 8 times longer than UDP packets for domestic sites, and 5 to 184 times longer for international sites. The extremely long delay can be attributed to the coarse grained TCP timeouts and its congestion avoidance algorithms. Hence, using TCP to delivery images may result in long delays for end users.

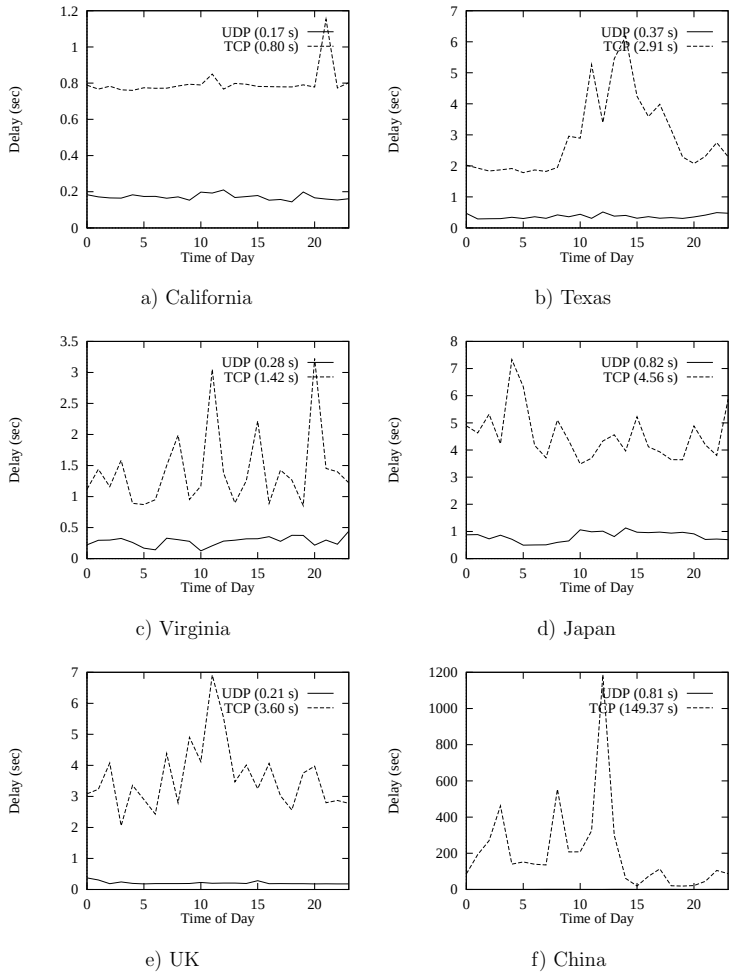


Figure 3.9: Comparisons of TCP and UDP end-to-end delays for the six test sites.

3.3.2 UDP Loss Behavior

In studying the packet loss characteristics of sending image data using UDP, again our goal is to find the interleaving factors that adjacent pixels should be separated by in order to make the probability of unrecoverable packet losses sufficiently small. Since in image transmissions, packets have been spaced out differently from those in video transmissions, we need to do similiar experiments as described in Section 3.2.2 to understand whether packet spacings would affect UDP loss behaviors. We computed the failure probability for various interleaving factors using (3.3) and plotted them in Figure 3.10. Because these experiments were done at different time instances from the video case, the graphs looked different from those in the video case (shown in Figure 3.5), but they still led us to draw a similar conclusion, *i.e.,* interleaving factors of 4 or less are sufficient for most domestic and international connections.

From the above study, we conclude that bursty losses in UDP delivery of images can be concealed effectively by interleaving and reconstruction using appropriate interleaving factors. Of course, it is expected that the reconstructed image will have lower quality. However, such trade-offs between quality and delay may be acceptable since users, when downloading an image from the Web, may not want to wait for minutes for a perfect image but may prefer to have a slightly degraded one in much shorter time.

3.4 Trace-Based Simulations

To make fair comparisons of different coding and error concealment algorithms, one needs to make sure the schemes are tested under the same network conditions. This is hard to satisfy in real-world transmissions because even two packets sent at almost the same

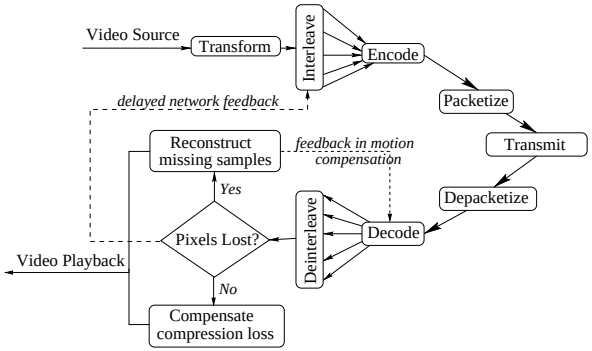


Figure 3.11: Components of our image and video transmission system.

3.4.1 System Prototype

As illustrated in Figure 3.11, our image and video transmission prototype consists of the following components. *Transformation* pre-processes the image and video signals in such a way that if image and video data suffer from losses during transmission, they can be better reconstructed using a prescribed method at the receiver. *Interleaving* is applied to convert bursty losses to random losses, and *de-interleaving* is simply its inverse

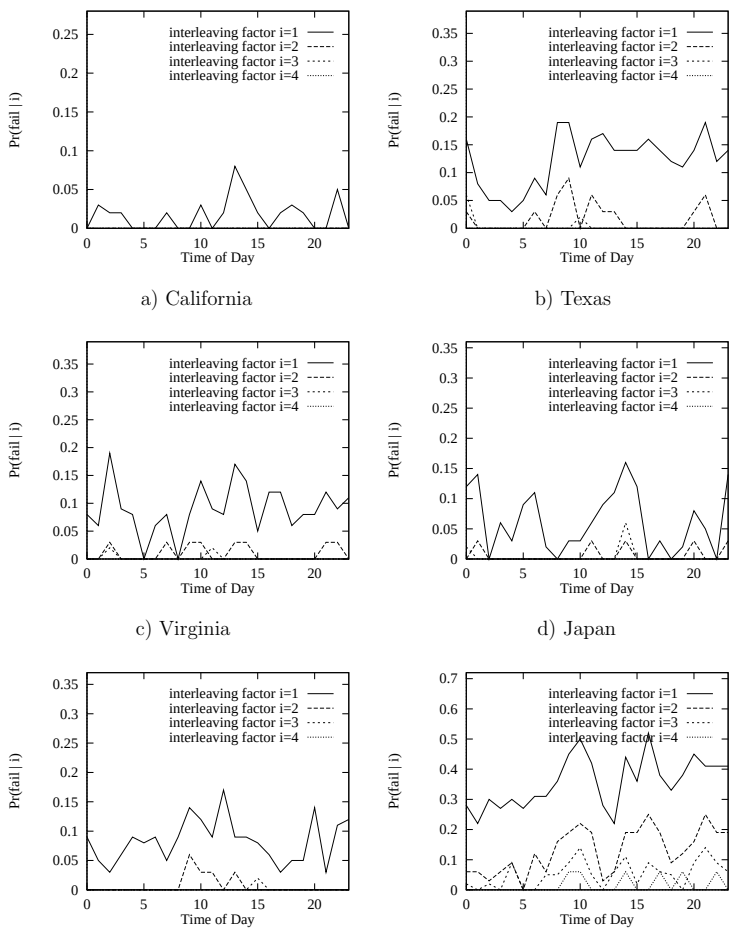


Figure 3.10: $Pr(fail|i)$, probability of bursty losses that cannot be recovered conditioned on interleaving factor i, at different times on April 8, 2001.

process. *Encoding* and *decoding* are needed to reduce the data rate to the level that can be sustained by the networks. *Packetization* divides the coded bit stream into smaller units for transmission, and *de-packetization* reassembles these units. The packets carrying image and video data are sent over the network for *transmission*. If they suffer from losses, *reconstruction* tries to estimate the lost signals by what have been received. If no losses happen, an optional *compensation* module is inserted to reduce coding loss.

There are no clear boundaries between some of the components, and some of them can be jointly designed to yield better performance. The relevant components are discussed in detail in the later chapters.

3.4.2 Sketch of Sender and Receiver Algorithms

Based on the prototype, our trace-based simulations consist of two processes, whose steps are outlined in Figure 3.12. Since every scheme we are about to study differs from others in some aspects, we only explain the common steps involved.

The *sender* process is responsible for transforming, properly interleaving and compressing the transformed and interleaved sub-frames. At the same time, it reads the traffic traces, maps packet losses to losses in image or video data, and selectively saves the contents that have been "received" to disk. For video transmissions, the interleaving process can be dynamic based on feedbacks from receivers; however, for image transmissions, this is impossible since image data could have been sent before feedbacks are received.

The *receiver* process is in charge of decompressing coded streams, deinterleaving them, and performing reconstructions. It reads from the files saved by the sender process, and groups them into interleaved sets. If some packets in an interleaved set are lost, then it

process *sender*

1. **foreach** (image or video frame) **do**
2. Transform the frame;
3. Decide on the proper interleaving factor based on loss statistics fed back from the receiver, and interleave the transformed frame (video only);
4. Compress the interleaved transformed sub-frames;
5. Packetize each sub-frame.
6. Based on the loss trace, selectively save the packets to disk.
7. **end-foreach**

end

process *receiver*

1. **foreach** (image or video frame) **do**
2. Read the packets saved by *sender* and group them into interleaved sets;
3. Decompress each sub-frame;
4. **if** all packets in an interleaved set are received **then**
5. Deinterleave the stream;
6. Compensate for coding loss (video only);
7. **else if** only some packets in an interleaved set are received **then**
8. Reconstruct the missing samples using simple interpolation;
9. **else if** all packets are lost **then**
10. copy from the last received frame or fill in the average value;
11. **end-if**
12. **end-foreach**

end

Figure 3.12: Outline of the sender and receiver algorithms in our trace-based simulations.

performs reconstruction using simple interpolation. If all the packets are lost, then in the video transmission case, samples are recovered by copying from a previously decoded frame, and in the image transmission case, they are recovered by simply using the average value of the image. The reconstruction process is kept simple in order to facilitate real-time playback.

Throughout the thesis, these two processes are used to evaluate different schemes, using the packet traces that were collected in the experiments previously described.

Chapter 4

Error Concealments for H.263-based Video Coding

Increases in bandwidth and computational speed lead to growing interests in real-time video transmissions over the Internet. In this chapter, we first review the H.263 video coding standard, which is specifically designed for low bit rate video transmissions at a target bandwidth below 64 kpbs. Next, we study various error concealment algorithms to combat three kinds of quality losses encountered in video streaming over unreliable IP networks. For countering bitstream losses, we propose to use a joint sender receiver-based multiple-description coding method to facilitate reconstruction of such losses. For countering propagation losses, we propose to modify the motion compensation part of decoders so that the reference frame is more accurate in case of packet losses. Further, we also adopt a syntax-based packetization strategy in order to reduce propagation losses among packets. For countering compression losses, we study an *artificial neural network-based reconstruction* to compensate for losses introduced in multiple-description compression. ANN is used because compression is a highly nonlinear complex process,

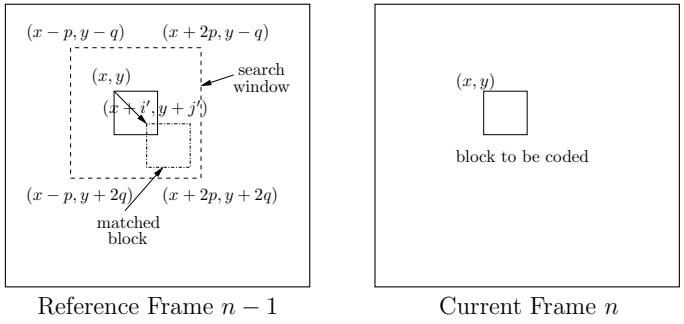


Figure 4.1: Illustration of the motion estimation process in H.263.

certain region in a frame, the best matching part in its previously coded frame is found. This process is called *motion estimation*. Second, although the matched regions are very similar, they are not identical; hence, the differences between matched regions are also computed. This is accomplished by *motion compensation*. The differences computed are generally of small magnitude, and most of them are zero. Compared with the information content of an entire frame, coding displacement vectors and a few small differences amounts to enormous savings. In H.263, the motion estimation is performed for each macroblock of size 16×16. Figure 4.1 illustrates this process: macroblock starting at (x, y) in frame n is compared against $2p \times 2q$ candidates within the big search window in the reference frame $n-1$ (already coded). For block starting at $(x+i, y+j)$ in frame $n-1$, the matching error is calculated as

$$MAE(i,j) = \frac{1}{MN} \sum_{k=0}^{M-1} \sum_{l=0}^{N-1} \|c(x+k, y+l) - r(x+i+k, y+j+l)\|. \qquad (4.1)$$

and ANNs have been shown effective in modeling nonlinear behavior. Finally, we describe experimental results to evaluate our proposed algorithms in real Internet tests.

4.1 Review of H.263 Coding Algorithm

The basic configuration of the H.263 standard is largely based on its forerunner, ITU H.261, and its ISO counterparts: MPEG-1 and MPEG-2. In addition, several enhancements have been made to achieve better compression performance, such as unrestricted motion vectors, syntax-based arithmetic coding, advanced prediction, and PB-frames. In this section, we describe the basic concepts underlying this codec, in order to prepare for discussions of error concealment in later sections.

The essential task of video compression is to remove redundancies in a video sequence. There are two forms of redundancies present in videos: spatial redundancy among pixels within a frame and temporal redundancies among adjacent frames. In H.263, spatial redundancy is exploited by Discrete Cosine Transform (DCT) that concentrates the image energy into few transform coefficients, and temporal redundancy is largely removed by a process called motion estimation and compensation. Besides these two important techniques, a lossy quantization step and a lossless entropy coding stage are also included to remove residual redundancies among transformed symbols. In the following subsections, these four schemes are discussed in detail.

4.1.1 Motion Estimation and Compensation

In a video sequence, most frames look very similar except at scene changes; therefore, one can get huge reduction in bit rate by not coding the same contents repeatedly from frame to frame. This temporal relationship is generally exploited in two steps. First, for

In (4.1), $c(u,v)$ and $r(u,v)$ represent the pixels in the current and the reference frames, respectively. The size of the macroblock is M by N, and typically both are set to 16. The macroblock that produces the smallest error ($MAE(i',j')$) among all the candidates is chosen as the reference block, and (i',j') is output as the integral motion vector for block (x,y). To further improve the accuracy of estimation, the same matching process is applied to eight more candidate macroblocks at half-pixel positions around $(x+i',y+j')$ to find the fractional motion vector that results in the smallest matching error. The final motion vector to be coded is the sum of the integral and fractional vectors.

After finding the motion vector for block (x,y), the next step, motion compensation, simply involves computing the changes between these two blocks and outputing the resulted difference block to the next stage: DCT transformation.

4.1.2 DCT Transformation

The basic idea of Discrete Cosine Transform (DCT) [57] is to decorrelate the input samples, thereby reducing the number of coefficients to be coded. There are a number of other transformations that serve the same purpose, such as Discrete Fourier Transform, Discrete Sine Transform, and Discrete Hadamard Transform, but DCT is chosen by most transform-based coders because of its low computational complexity, near optimal transform efficiency and orthonormal property [34, 57].

In H.263, each image is grouped into 8×8 blocks and input to the DCT module. There are two kinds of blocks to be processed, depending on their coding modes: in intra-coding, the original pixel block taken from the current frame is transformed; whereas in inter-coding, the residual block resulted from the previous motion compensation step is transformed. In the following equation, $t(i,j)$ $(i,j = 0,1,\ldots,7)$ denotes the pixel

(or residual) value at position (i,j) in a block, and $f(u,v)$ $(u,v = 0,1,\ldots,7)$ is its transformed coefficient, The forward and inverse transformations are expressed as:

$$f(u,v) = \alpha_u\alpha_v\frac{1}{4}\sum_{i=1}^{8}\cos\left(\frac{(2i-1)(u-1)\pi}{16}\right)\sum_{j=1}^{8}t(i,j)\cos\left(\frac{(2j-1)(v-1)\pi}{16}\right)$$

$$t(i,j) = \frac{1}{4}\sum_{u=1}^{8}\alpha_u\cos\left(\frac{(2i-1)(u-1)\pi}{16}\right)\sum_{v=1}^{8}f(u,v)\alpha_v\cos\left(\frac{(2j-1)(v-1)\pi}{16}\right),$$

where $\alpha_1 = 1/sqrt(2)$ and $\alpha_u = 1$ for $u = 2,3,\ldots,8$. The above equations show that the 2D DCT is essentially a separable transform, with transformations of rows followed by columns. This separable feature makes it possible to rewrite the inverse transforms in the following way:

$$t(i,j) = \sum_{u=1}^{8}\sum_{v=1}^{8}t(u,v)\mathbf{b}_u\mathbf{b}_v^T, \tag{4.2}$$

$$\text{where } \mathbf{b}_u = \left\{\frac{\alpha_u}{2}\cos\frac{(2k-1)(u-1)\pi}{16}\right\}_{k=1,2,\ldots,8}$$

$$\text{and } \mathbf{b}_v = \left\{\frac{\alpha_v}{2}\cos\frac{(2k-1)(v-1)\pi}{16}\right\}_{k=1,2,\ldots,8}.$$

$\mathbf{b}_u$ and $\mathbf{b}_v$ are the u^{th} and v^{th} basis vectors of DCT. The product $\mathbf{b}_u\mathbf{b}_v^T$ is normally called the basis image of DCT. A coefficient block can be considered as the projection of a pixel block into the correponding basis images, and conversely, the original block can be expressed as a superimposition of basis images.

The top portion of Figure 4.2 illustrates the effects of DCT transform. The input block (the first block), taken from a sample image, has pixel values ranging from 91 to

125	123	122	122	121	121	125	122
113	103	104	109	109	111	111	112
114	102	102	107	109	114	112	111
105	91	95	98	105	112	106	103
115	114	123	125	121	116	114	116
114	115	121	117	115	112	113	114
115	123	129	117	114	113	118	121
115	117	115	106	99	91	92	99

⇓ (DCT)

898	5	1	-1	3	4	4	2
4	-19	2	17	4	5	0	-2
11	16	8	-9	-3	-3	-4	0
45	-1	-6	-9	-3	0	-2	1
-5	7	-2	0	1	0	0	0
7	-8	3	3	-5	0	0	1
-1	5	1	1	1	0	-1	0
27	2	-2	-6	0	0	-1	0

⇓ (Quantization)

112	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0

Figure 4.2: An example of DCT and quantization processing of a block.

125. After the DCT transform, most of the energy is compacted into the component at position $(0, 0)$, and the rest of the coefficients are relatively small in magnitude. When combined with quantization, this energy compaction property can effectively reduce the number of coefficients to code. The coefficient at location $(0, 0)$ reflects the average value of the block and is identified as the "DC coefficient", and the remaining 63 coefficients are called "AC coefficients."

4.1.3 Quantization

Following the DCT transform, quantization is applied to reduce the magnitudes and increase the number of zero coefficients. The purpose of quantization is to achieve further compression by discarding information that is visually not important. Quantization can be either uniform or non-uniform. In uniform quantization, the same quantization level is applied to coefficients of all magnitudes, whereas in non-uniform quantization, different quantization levels are used for coefficients of different magnitudes. In theory, the optimal quantizer that minimizes the expected quantization error for a given input signal is generally non-uniform, but such quantizers are dependent on the statistical properties of input samples and, hence, are difficult to use in practice. That explains why many coders, including H.263, adopt the simple uniform quantization in their algorithms. Let $Q(u, v)$ denote the quantization factor, and $f(u, v)$ is the coefficient to be quantized, then the quantizer output $c(u, v)$ is obtained as follows:

$$c(u, v) = \frac{\left\lfloor f(u, v) \pm \frac{Q(u,v)}{2} \right\rfloor}{Q(u, v)}. \tag{4.3}$$

The symbol $\pm$ means it is an addition for positive $f(u, v)$, and subtraction for negative $f(u, v)$.

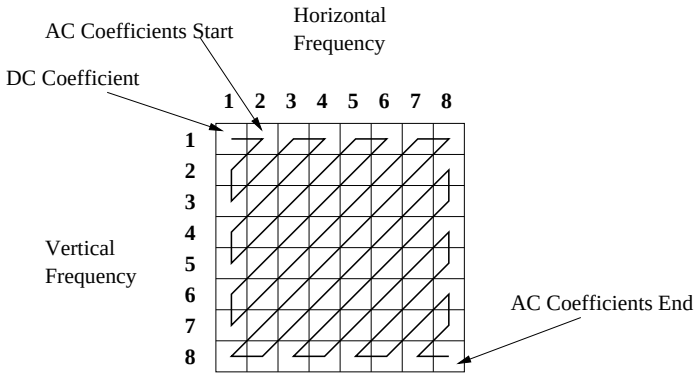


Figure 4.3: Zig-zag ordering of AC coefficients.

It is helpful to consider the entropy coding of AC coefficients as a 3-step process. The first step orders the quantized coefficients in the "zig-zag" pattern shown in Figure 4.3. This ordering facilitates later encodings by putting low-frequency coefficients before high-frequency ones. The second step converts the zig-zag sequence of AC coefficients into an intermediate representation. The last step actually encodes this intermediate representation into a sequence of binary bits by consulting a Huffman table.

The output of entropy coding is a coded bit stream, in which there is no recognizable boundaries of individual coefficients. This creates a problem when the coded bit stream needs to be divided and put into packets for transmission. In Section 4.4.2, we discuss how to overcome this difficulty.

The inverse quantization is simply done by:

$$f(u,v) = c(u,v)Q(u,v). \tag{4.4}$$

The bottom part of Figure 4.2 illustrates the quantization of a coefficient block (the second block) by a uniform quantizer of 10. Before quantization, there are 50 nonzero coefficients, but only 3 remain after quantization. Therefore, quantization can significantly reduce the number of coefficients to be coded.

Quantization is a many-to-one mapping and, therefore, is fundamentally lossy. It is the principle source of lossiness in H.263 coding. A nice property of DCT transform is that the energy of an error introduced by quantizing coefficients is equal to that calculated from the original pixels; hence, one can vary quantization factors in order to introduce controlled losses. If small (fine) quantization factors are used, less quantization errors are introduced, resulting in images of good fidelity. On the other hand, large (coarse) quantization factors will produce images of poor quality.

H.263 uses the same $Q(u,v) = Q$ for all coefficients $f(u,v)$, $u,v = 1,2,\ldots,8$. More generally, $Q(u,v)$ can take different values for different coefficients, in which quantization is characterized by a matrix. In Chapter 5, we discuss the design of quantization matrices to improve the reconstruction performance in lossy situations.

4.1.4 Entropy Coding

Entropy coding is the last step in an H.263 coder, which achieves additional reduction in bit rate by exploiting the statistical correlations among quantized coefficients.

Here, DC coefficients are treated differently from AC coefficients. They are encoded by fixed length coding, while AC coefficients are encoded by variable length coding methods, such as huffman coding [30] and arithmetic coding [52].

Table 4.1: Typical compression rations for videos of various formats.

Image Format	Compression Ratio			
	10f/s		20f/s	
	56 (kbps)	80 (kbps)	56 (kbps)	80 (kbps)
CIF (352×288)	212	149	424	297
QCIF (174×144)	53	37	106	74

4.1.5 Analysis

As described above, H.263 is targetted for a very low bit rate coding environment, normally in tens of kilo-bits per second. The typical compression ratios of H.263 coded videos at various frame and bit rates are listed in Table 4.1. In the following, we analyze how the different techniques we have described contribute to compression.

- *Transformation and quantization.* In Figure 4.2, we have seen that DCT can transform an 8 by 8 pixel block into a frequency representation that has just few dominant coefficients. After quantization, the number of non-zero coefficients are significantly reduced. Take *missa* sequence as an example, the intra-coded frame size is 4862 bytes using a quantization factor of 10. Comparing this with the original frame size of a CIF picture (101376 bytes), we have a compression ratio of 21. This reduction in bit rate is primarily contributed by transformation and quantization.
- *Temporal differnce coding.* For inter-coded frames, temporal redundancy is removed by predictive coding, which codes only the displacement vectors and differences between the current frame and its reference frame. This temporal difference coding is a major reason why video coding can achieve larger compression ratios than image coding. Using *missa* sequence as an example, the average P frame size of

this sequence is only 390 bytes using a quantization factor of 10, the same as that used for the intra-coded frame. Hence, using temporal difference coding leads to a bandwidth reduction of 80% for this sequence.

- *Entropy coding.* This last stage removes the residual redundancies by exploiting the statistical relationships among quantized symbols. Entropy coding can normally achieve a compression ratio of 2 to 3 [83].

From the above discussion, we understand that all the three techniques are important in achieving good compression performance in error free conditions. The question we need to address in our research is how they perform in lossy environment, such as the Internet.

4.2 Issues in Streaming H.263 Coded Videos

There are three kinds of losses to consider when H.263 coded videos are transmitted over the Internet. In Chapter 3, we have learned that packet losses are common in Internet transmissions. In general, domestic connections experience less than 10% packet losses, whereas international connections can suffer from over 50% losses. The losses of transmitted packets can cause losses in a coded bit stream, which is called *bitstream loss*.

Moreover, packet losses can have a compound effect on subsequent dependent frames due to temporal-difference coding and entropy coding, although they are lossless coding techniques themselves. The normal coding pattern of H.263 is to code a sequence of P frames following an I frame. Figure 4.4 illustrates this dependency relationship. This relationship implies that a packet loss happened to the I frame or a previous P frame can affect all subsequent P frames, until the next I frame comes. In addition, since variable length entropy coding includes state information into a stream, packet losses can cause decoders to go into an erroneous state, rendering subsequent packets useless even when

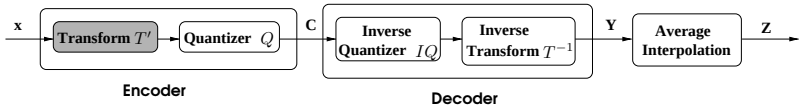


Figure 4.5: Basic building blocks of a modified codec. (The shaded block is our proposed ORB-DCT.)

each description, and assigns the multiple descriptions of the GOB to different packets in transmission to the destination.

In this process, a simple scheme that codes interleaved streams may not work well because the original DCT and quantizer are not necessarily the best for reconstructing lost streams. In this section we propose a new optimized reconstruction-based DCT (ORB-DCT) that takes into account the reconstruction process at the receiver.

Figure 4.5 shows a simplified diagram of the basic building blocks in our proposed transform-based system. It is based on existing state-of-the-art video codecs that have three stages: transformation, lossy quantization, and lossless entropy coding. The transformation stage concentrates the energy into the first few transform coefficients and decorrelates the coefficients; the quantizer causes a controlled loss of information; and the entropy coder removes residual redundancies among quantized symbols. Our goal is to find a new transform T' in order to minimize reconstruction error $\mathcal{E}_r$ after average interpolation, based on fixed quantization Q, inverse quantization IQ, and inverse DCT T^{-1}. (We do not indicate the entropy coder in Figure 4.5 as it is lossless.)

$$\mathcal{E}_r = \| \underbrace{Interpolate(T^{-1}(IQ(\mathbf{c})))}_{decompression + reconstruction} - \mathbf{x} \|^2 . \tag{4.5}$$

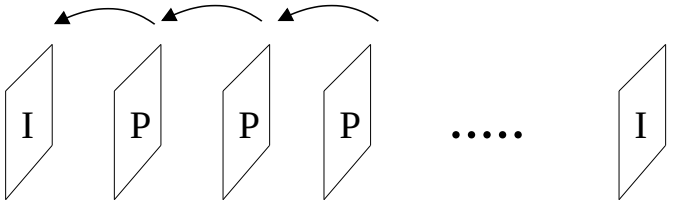


Figure 4.4: Dependency relationship of H.263 frames.

they are received correctly. The information loss in this form is called *propagation loss*. In Figure 1.3, we have given an example illustrating the effects of propagation loss.

The last sort of information losses, *quantization losses*, comes from compression itself. As discussed above, quantization factors can be adjusted to introduce losses of various degrees and to meet the target bit rate constraints. For example, for the *missa* sequence, after compression at a frame rate of 15f/s and a target bit rate of 80 kbps, the average distortion per pixel, measured in absolute error difference, is 8 gray levels.

In summary, we have shown that very low bit rate coding techniques, like H.263, that achieve large compression by aggressively removing redundancies in image sequences, can generate bit streams that are very vulnerable to information losses. We have identified three kinds of losses when streaming H.263 videos on the Internet. In the next section, we describe techniques to cope with these three kinds of losses.

4.3 Optimized Reconstruction-Based DCT to Cope with Bistream Loss

Our proposed system consists of an MDC-based strategy that *interleaves* adjacent pixels of a group of blocks (GOB) in the original video stream into multiple descriptions, codes

Putting (4.7) in matrix form gives:

$$\mathbf{Y}_1 = (\mathbf{p}_1 \quad \mathbf{p}_2 \quad \dots \quad \mathbf{p}_8)_{8\times 8}, \text{ where } \mathbf{p}_k = \sum_{i=1}^{8}\sum_{j=1}^{8} C_{i,j}\mathbf{b}_i b_{j,k} \qquad k = 1, 2, \dots, 8, \quad (4.8)$$

$b_{j,k}$ is the k^{th} component of basis vector $\mathbf{b}_j$. The set of interpolated pixels $\mathbf{Z}$ is obtained by inserting even-numbered columns as the average of columns from $\mathbf{Y}_1$, with the boundary column duplicated:

$$\mathbf{Z} = \left(\mathbf{p}_1 \quad \frac{\mathbf{p}_1 + \mathbf{p}_2}{2} \quad \mathbf{p}_2 \quad \frac{\mathbf{p}_2 + \mathbf{p}_3}{2} \quad \dots \quad \mathbf{p}_8 \quad \mathbf{p}_8\right)_{8\times 16}. \quad (4.9)$$

$\mathbf{Z}$ can also be expressed as:

$$\mathbf{Z} = \sum_{i=1}^{8}\sum_{j=1}^{8} C_{i,j}\mathbf{b}_i\mathbf{e}_j^T, \text{ where } \mathbf{e}_j = \left(b_{j1} \quad \frac{b_{j1} + b_{j2}}{2} \quad b_{j2} \quad \dots \quad b_{j8} \quad b_{j8}\right)^T. \quad (4.10)$$

We define $\mathbf{e}_j$ as an extended basis vector for reconstruction purpose. The distortion between the original and the received and reconstructed pixels is:

$$\mathcal{E}_r = \left\| \sum_{i=1}^{8}\sum_{j=1}^{8} C_{i,j}\mathbf{b}_i\mathbf{e}_j^T - \mathbf{X} \right\|^2. \quad (4.11)$$

To minimize $\mathcal{E}_r$ with respect to $\mathbf{C}$, we first linearize each matrix using operator $\mathcal{L}(\cdot)$ into a vector by a raster-scan order, *i.e.*, following the first row by the second row in a matrix, and so on. The following notations are defined after linearization:

$$\vec{\mathbf{u}} = \mathcal{L}\left\{(C_{i,j})_{(8\times 8)}\right\} \quad (4.12)$$

$$\vec{\mathbf{v}}_{8(i-1)+j} = \mathcal{L}\left\{\mathbf{b}_i\mathbf{e}_j^T{}_{(8\times 16)}\right\} \quad (4.13)$$

$$\vec{\mathbf{w}} = \mathcal{L}\left\{(X_{i,j})_{(8\times 16)}\right\}, \quad i, j = 1, \dots, 8. \quad (4.14)$$

In order to keep our decoders standard-compliant so that decoders at existing receivers are unchanged, we assume fixed inverse quantization IQ and inverse DCT T^{-1}.

With quantization in place, the minimization of $\mathcal{E}_r$ becomes an integer optimization problem, where $\mathbf{c}$ in (4.5) takes integer values. Such optimizations are computationally prohibitive in real time. In the following, we derive an approximate solution that does not take into account quantization effects. Specifically, the objective to be optimized in the following approximation is:

$$\mathcal{E}_r = \| \mathit{Interpolate}(T^{-1}(\mathbf{c})) - \mathbf{x} \|^2. \quad (4.6)$$

The resulting transform is called optimized reconstruction-based DCT (ORB-DCT). In the following, we first derive ORB-DCT based on partitioning video data into two descriptions and describe in Section 4.3.3 its extension to four descriptions.

4.3.1 ORB-DCT for Intra-Coded Blocks

Assume that the original frame is divided into blocks of 8×16 pixels. After ORB-DCT, block $\mathbf{X}$ is transformed into two blocks $\mathbf{C}_1$ and $\mathbf{C}_2$, each of size 8×8, corresponding to blocks of odd-numbered and even-numbered pixels, respectively. Since the derivations are similar, we only show the derivations for $\mathbf{C}_1$.

Our objective is to find $\mathbf{C}_1$ to minimize $\mathcal{E}_r$. After inverse DCT, output $\mathbf{Y}_1$ is calculated as:

$$\mathbf{Y}_1 = \sum_{i=1}^{8}\sum_{j=1}^{8} C_{i,j}\mathbf{b}_i\mathbf{b}_j^T, \text{ where } \mathbf{b}_i = \left\{\frac{\alpha_i}{2}\cos\frac{(2k-1)(i-1)\pi}{16}\right\}_{k=1,2,\dots,8}, \quad (4.7)$$

$C_{i,j}$ is the $(i,j)^{th}$ element of $\mathbf{C}_1$, $\mathbf{b}_i$ is the i^{th} basis vector of DCT, $\alpha_1 = \frac{1}{\sqrt{2}}$, and $\alpha_k = 1$ for $k = 2, 3, \dots, 8$.

4.3.2 ORB-DCT for Inter-Coded Blocks

For inter-coded blocks, output $\mathbf{Y}_1$ after inverse DCT, as shown in (4.7), is the residual block after motion prediction. Denote its corresponding reference block as:

$$\mathbf{R} = (\mathbf{r}_1 \quad \mathbf{r}_2 \quad \dots \quad \mathbf{r}_8)_{8\times 8}\,. \tag{4.20}$$

Then the interpolated data $\mathbf{Z}$ is the sum of two terms after motion compensation:

$$\begin{aligned}\mathbf{Z} &= \left(\mathbf{p}_1 \quad \frac{\mathbf{p}_1+\mathbf{p}_2}{2} \quad \mathbf{p}_2 \quad \frac{\mathbf{p}_2+\mathbf{p}_3}{2} \quad \dots \quad \mathbf{p}_8 \quad \mathbf{p}_8\right)\\ &\quad + \left(\mathbf{r}_1 \quad \frac{\mathbf{r}_1+\mathbf{r}_2}{2} \quad \mathbf{r}_2 \quad \frac{\mathbf{r}_2+\mathbf{r}_3}{2} \quad \dots \quad \mathbf{r}_8 \quad \mathbf{r}_8\right)\\ &= \sum_{i=1}^{8}\sum_{j=1}^{8} C_{i,j}\mathbf{b}_i\mathbf{e}_j^T + \mathbf{R}'. \end{aligned}\tag{4.21}$$

Substituting (4.21) into (4.11) yields the reconstruction error for inter-coded blocks:

$$\mathcal{E}_r = \left\| \sum_{i=1}^{8}\sum_{j=1}^{8} C_{i,j}\mathbf{b}_i\mathbf{e}_j^T - (\mathbf{X}-\mathbf{R}') \right\|^2. \tag{4.22}$$

To derive ORB-DCT in this case, we note that only vector $\vec{\mathbf{w}}$ is different as compared to the case of intra-coded blocks. From (4.18), it is obvious that the transform itself does not depend on $\vec{\mathbf{w}}$; therefore, ORB-DCT retains the same form.

In short, ORB-DCT can be applied uniformly for both intra- and inter-coded blocks. For intra-coded blocks, it is applied to an original block $\mathbf{X}$ to produce transform coefficients $\mathbf{C}_i, i = 1, 2$; for inter-coded blocks, it is applied to the interpolated motion-predicted block $(\mathbf{X} - \mathbf{R}')$.

Like DCT, ORB-DCT is also a row-column-separable transform. To compute a transform coefficient of ORB-DCT by a row-column approach, it takes 40 floating-point mul-

We further define matrix $\mathbf{V}$ as:

$$\mathbf{V} = (\vec{\mathbf{v}}_1 \quad \vec{\mathbf{v}}_2 \quad \vec{\mathbf{v}}_3 \quad \dots \quad \vec{\mathbf{v}}_{64})\,. \tag{4.15}$$

Then (4.11) can be rewritten as follows:

$$\mathcal{E}_r = \| \mathbf{V}\vec{\mathbf{u}} - \vec{\mathbf{w}} \|^2, \tag{4.16}$$

where $\mathbf{V}$ is a 128×64 matrix, $\vec{\mathbf{u}}$, a 64×1 vector, and $\vec{\mathbf{w}}$, a 128×1 vector. Since the linear system of equations $\mathbf{V}\vec{\mathbf{u}} = \vec{\mathbf{w}}$ is an over-determined one, there exists at least one least-square solution $\vec{\mathbf{u}}$ that minimizes (4.16) according to the theory of linear algebra [54]. Specifically, the solution $\vec{\mathbf{u}}$ with the smallest length $|\vec{\mathbf{u}}|^2$ can be found by first performing SVD decomposition of matrix $\mathbf{V}$:

$$\mathbf{V} = \mathbf{S}\,[diag(w_j)]\,\mathbf{D}^t, \qquad j = 1, 2, \dots, 64, \tag{4.17}$$

where $\mathbf{S}$ is a 128×64 column-orthogonal matrix, $[diag(w_j)]$, a 64×64 diagonal matrix with positive or zero elements (singular values), and $\mathbf{D}$, a 64×64 orthogonal matrix. Then the least-square solution can be expressed as:

$$\vec{\mathbf{u}} = \mathbf{D}\,[diag(1/w_j)]\,\mathbf{S}^T\,\vec{\mathbf{w}}. \tag{4.18}$$

In the above diagonal matrix $[diag(1/w_j)]$, the element $1/w_j$ is replaced by zero if w_j is zero. Therefore, ORB-DCT is a product of three matrices: $\mathbf{D}$, $[diag(1/w_j)]$, and $\mathbf{S}^T$.

To derive the ORB-DCT transform for $\mathbf{C}_2$, simply replace $\mathbf{e}_j, j = 1, 2, \dots, 8$, in (4.10) by:

$$\mathbf{e}_j = \left(b_{j1} \quad b_{j1} \quad \frac{b_{j1}+b_{j2}}{2} \quad b_{j2} \quad \dots \quad \frac{b_{j7}+b_{j8}}{2} \quad b_{j8}\right)^T.$$

The rest of the steps are similar.

where $z_{i,j}$ is the value of the pixel in row i and column j. The transformed values of Description 1 in order to achieve the optimal reconstruction in (4.23) can be derived as outlined in Sections 4.3.1 and 4.3.2. In a similar way, we need to derive transformations when one, two, or three descriptions are lost. Since it is impossible to know the specific loss pattern for an interleaved set until it is received at the receiver and it will be either overly optimistic or overly pessimistic if one loss pattern is selected *a priori*, the method is impractical for use on the Internet.

Second, we can partition video data into four descriptions by interleaving the original frame $\tilde{\mathbf{z}}$ in the horizontal direction into two streams, $\tilde{\mathbf{z}}_{\mathbf{h1}}$ and $\tilde{\mathbf{z}}_{\mathbf{h2}}$, and then by interleaving and transformations in the vertical direction to get two additional descriptions. The concept is illustrated in Figure 4.6. In a different way, we can also get four descriptions by first partitioning in the vertical direction and then in the horizontal direction. The four descriptions, $\tilde{\mathbf{z}}_{\mathbf{h1,v1}}$, $\tilde{\mathbf{z}}_{\mathbf{h1,v2}}$, $\tilde{\mathbf{z}}_{\mathbf{h2,v1}}$ and $\tilde{\mathbf{z}}_{\mathbf{h2,v2}}$, are then sent in distinct packets to the receiver, which carries out the following operations in order to reconstruct any missing descriptions.

a) If one out of the four interleaved descriptions is received, say $\tilde{\mathbf{z}}_{\mathbf{h1,v1}}$, then $\tilde{\mathbf{z}}_{\mathbf{h1,v2}}$ can be reconstructed optimally by taking averages along the vertical direction of pixels from $\tilde{\mathbf{z}}_{\mathbf{h1,v1}}$. By taking averages along the horizontal direction, $\tilde{\mathbf{z}}_{\mathbf{h2,v1}}$ and $\tilde{\mathbf{z}}_{\mathbf{h2,v2}}$ can then be recovered.

b) If two out of the four interleaved descriptions are received, then there are two possible scenarios. If the lost descriptions are from the same horizontally interleaved group, say $\tilde{\mathbf{z}}_{\mathbf{h1,v1}}$ and $\tilde{\mathbf{z}}_{\mathbf{h1,v2}}$, then they can be reconstructed by averaging of their horizontal neighbors. If the lost descriptions do not belong to the same horizontally interleaved

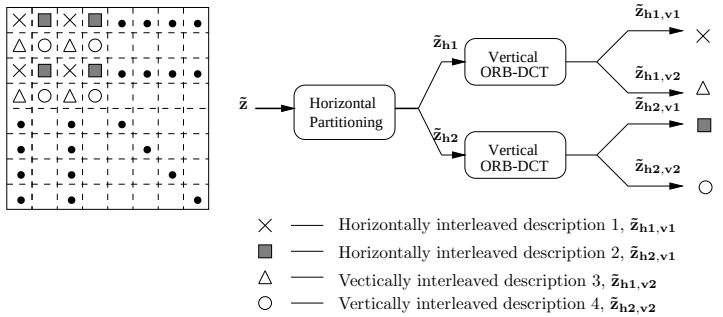


Figure 4.6: Four-way interleaving in the horizontal and vertical directions.

tiplications and 37 floating-point additions. In the future, we plan to study fast implementations of ORB-DCT, similar to what was done in deriving fast DCT.

4.3.3 Handling Burst Lengths of Four

As described in Chapter 3, the burst length of lost packets is likely to be greater than two for transcontinental connections. In this section we describe two ways to handle cases with a maximum burst length of four. We do not describe methods to handle burst lengths longer than four because such cases are infrequent and end-to-end delays will be intolerable.

First, assume that only one out of three interleaved descriptions, say Description 1 – $\tilde{\mathbf{z}}_{\mathbf{h1,v1}}$ – is received. The remaining three descriptions can be restored as follows:

$$\hat{z}_{i,j} = \begin{cases} \frac{(z_{i,j-1}+z_{i,j+1})}{2} & \hat{z}_{i,j} \in \tilde{\mathbf{z}}_{\mathbf{h2,v1}} \\ \frac{(z_{i-1,j}+z_{i+1,j})}{2} & \hat{z}_{i,j} \in \tilde{\mathbf{z}}_{\mathbf{h1,v2}} \\ \frac{(z_{i-1,j-1}+z_{i-1,j+1}+z_{i+1,j-1}+z_{i+1,j-1})}{4} & \hat{z}_{i,j} \in \tilde{\mathbf{z}}_{\mathbf{h2,v2}} \end{cases} \quad (4.23)$$

4.4 Schemes to Cope with Propagation Loss

In this section, we describe two schemes to cope with two kinds of propagation losses: *i.e.*, *decoder feedback with reconstructed frames* to decrease propagation losses due to temporal difference coding, and *syntax-based packetization and depacketization* to reduce losses due to variable length entropy coding.

4.4.1 Decoder Feedback with Reconstructed Frames

The decoder in the original H.263 system uses the last completely received frame as its reference to an inter-coded frame if a reference frame is lost during transmission. We can do better in our system because, if a subset of descriptions are received, the lost sub-frames can be reconstructed using the correctly received descriptions. We expect reconstructed sub-frames to be closer to lost ones than sub-frames in the last completely received frame due to motion in the video sequence. Therefore, we feed the reconstructed sub-frame back to the motion-compensation loop of the H.263 decoder. (see Figure 4.7 for the modified H.263 decoder). In the motion-compensation algorithm, we select the reference frame for a P-frame in the following order:

a) Use the reference sub-frame in the current description if it is received correctly;

b) Use the reconstructed sub-frame if it is lost and can be reconstructed from other descriptions;

c) Otherwise, use the previous reference sub-frame within the current description.

4.4.2 Syntax-Based Packetization and Depacketization

A good packetization strategy prevents the propagation of errors among packets so that the loss of one packet will not render subsequent packets in an erroneous state.

description, say $\tilde{\mathbf{z}}_{\mathbf{h1,v1}}$ and $\tilde{\mathbf{z}}_{\mathbf{h2,v2}}$, then they can be reconstructed optimally by taking averages of their respective vertical neighbors.

c) If three out of the four descriptions are received, then the lost description can be reconstructed by taking averages along the vertical direction.

In short, the second method to handle longer bursty losses follows the inverse flow of the interleaving process in Figure 4.6 and can be generalized easily to 2^m-way interleaving, for $m > 0$. It is flexible because the transformation at the sender does not depend on the loss pattern at the receiver. For this reason, we have adopted this approach in our experiments.

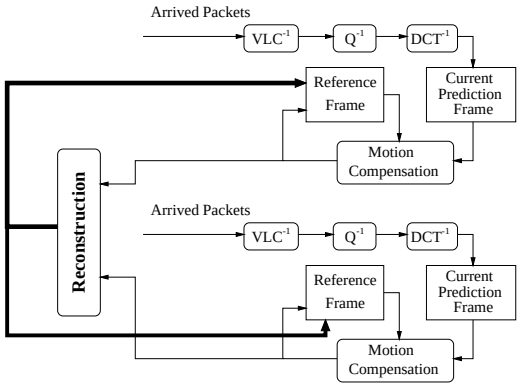


Figure 4.7: A modified H.263 decoder with feedback path (an expanded version of the decoder in Figure 3.11).

Our packetization strategy is based on the hierarchy that H.263 organizes its bit stream. The top level of the hierarchy is the *picture* layer that is divided into a sequence of *groups of blocks* (GOBs), each of which consists of a number of 16×16 *macroblocks* (MBs). Each MB consists of four 8×8 Y blocks, an 8×8 Cr block, and an 8×8 Cb block.

A GOB acts as the basic synchronization point in a coded stream. In most cases, when an error occurs within a GOB, the rest of the GOB will be undecodable, and the decoder has to resume synchronization at the start of the next GOB. As a result, we set our packet boundary corresponding to that of GOBs. Further, we choose our packet size of 512 bytes in order to avoid fragmentation in the Internet. Due to intra- and inter-coding and the use of variable-length coding, the size of a compressed GOB is variable. Hence, a packet may contain between a single coded GOB and several coded frames.

As GOBs serve as the basic syntactic units of packets, they also act as the basic unit for reconstruction purposes at the receiver.

4.5 Schemes to Cope with Compression Loss

In this section, we present our artificial neural network-based reconstruction method to deal with compression loss, the third kind of information loss that was identified.

4.5.1 Neural Networks For Compensating Quantization Errors

ORB-DCT described in Section 4.3 is used to minimize reconstruction errors when at least one description is lost during transmission and is reconstructed at the receiver. However, when all the descriptions are received at the receiver, perfect reconstruction is

still not possible. These errors are due to quantizations used when ORB-DCT coefficients are coded at the sender.

In this section, we first evaluate the compression-loss characteristics of MDC when all the descriptions are received at the receiver. Our evaluations show that compression and quantization errors are highly nonlinear and complex and cannot be compensated by a linear process. We then describe in detail our proposed neural-network architecture, input/output scaling, choice of activation functions, and learning algorithms.

4.5.2 Quantization Errors in MDC

The performance of a coding system can be characterized by its *coding gain*, which, in a first-order Markov source with adjacent sample correlation ρ, is upper bounded by [34]:

$$G(N) = \frac{1}{(1-\rho^2)^{\frac{N-1}{N}}}, \tag{4.24}$$

where N is the transform size. Since $G(N)$ in (4.24) is proportional to ρ of the video source and ρ is reduced after pixel-based interleaving, we expect the PSNR of an MDC system to be smaller than that of the original system at the same bit rate. This phenomenon is illustrated in Table 4.2 that compares ρ and PSNR of a horizontally 2-way deinterleaved stream and the original non-interleaved stream of two test image sequences.

In order to improve ρ or PSNR of the deinterleaved stream, we need better understanding of the effect of deinterleaving on individual pixels. Figure 4.8 illustrates the fluctuations observed in a smooth area and an area with a sharp transition after deinterleaving using pixel values from a horizontal line of an image in *missa*. (The curves showing the ANN-reconstructed stream are discussed in Section 4.6.2.) We observe that deinterleaved pixels incur spurious fluctuations in pixel values because different inter-

Using a learning algorithm, ANNs can be trained to learn complex tasks by progressively extracting meaningful features.

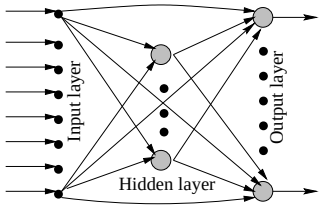


Figure 4.9: A three-layer feedforward ANN with full connections between the input-hidden and the hidden-output layers and shortcuts between the input-output layers.

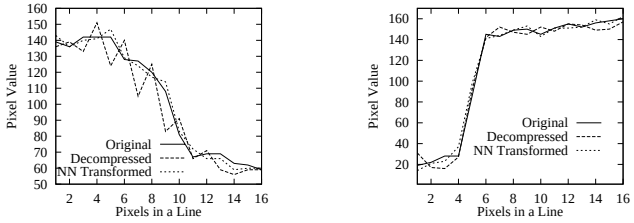
4.5.3 Neural-Network Architecture

Figure 4.9 illustrates a three-layer ANN architecture used in our study. It has full interconnections between the input-hidden layers, hidden-output layers, and input-output layers, the latter implementing *shortcuts*. This architecture provides two alternative mappings: a *nonlinear* mapping between the input and output layers if the synaptic weights from the input to the hidden layer have nonzero weights, and a *linear* mapping between the input and output layers if all the input-to-hidden weights are zero and some of the input-to-output weights are nonzero. It adapts to either mapping depending on which mapping leads to smaller training errors.

To facilitate real-time playback, ANN weights are trained in advance by the back-propagation algorithm [25] using a training set. Pixels taken from *deinterleaved and decompressed frames* serve as inputs to the ANN, and those taken from the *original frames* (before compression) serve as desired outputs. It is expected that weights adapted

Table 4.2: Adjacent sample correlations and PSNRs of a horizontally 2-way deinterleaved stream and the original non-interleaved stream.

Video Sequence	adjacent correlation ρ		PSNR (dB)	
	original	interleaved	original	interleaved
missa	0.9964	0.9829	37.48	36.74
football	0.9964	0.9819	31.13	30.16



a) Smooth area b) Sharp-transition area

Figure 4.8: Comparison of pixel values in the original stream, the deinterleaved stream, and the ANN-reconstructed stream. Pixel values are taken from a horizontal line of an image in *missa*.

leaved streams have different quantization losses. Such spurious fluctuations may be removed by suitable filtering.

In the following we propose to train a feedforward multilayer *artificial neural network* (ANN) to perform such filtering in real time. We use ANNs rather than adaptive linear filters because compression and decompression are highly nonlinear and complex and ANNs have been proven effective in finding nonlinear mappings between inputs and outputs. The nonlinearity of an ANN lies in a nonlinear activation function in each neuron and the one or more hidden layers that are not part of the input or output layers.

to the training tuples can capture the common characteristics of compression loss, and thus can be generalized to frames other than those in the training set.

To allow faster convergence in training, we have picked suitable initial weights in our experiments based on the fact that the decompressed pixel $\hat{g}(i,j)$ is very close to the original pixel $g(i,j)$. This relationship can be realized by initializing the weights between the input and output layers to values close to one and the other weights to values close to zero.

Previous studies [11, 40] show that training will be faster if the activation function of the hidden layer is centered around 0. Hence, we choose the *tanh* function in the range $[-1,1]$ as the activation function f_h of the hidden layer:

$$f_h(x) = \frac{2}{1+\exp(-2x)} - 1; \qquad f_o(x) = x. \tag{4.25}$$

To allow our ANN realize a linear mapping between the input and output layers, we choose the activation function of the output layer f_o to be linear and use a simple identity function.

To use the *tanh* and identity activation functions, image pixel values originally in the range $[0,255]$ must be scaled properly in order to avoid slow convergence when pixel values are too large. In our experiments, we scale input pixel values to $[-1,1)$ by $f_s(x) = (x-128)/128$ and rescale them back to $[0,255]$ at the output layer.

4.6 Experimental Results

We have evaluated our ORB-DCT and ANN reconstruction schemes using six test videos in two scenarios: a synthetic scenario under controlled losses and real Internet tests. In

the following, we only show the PSNR of the Y component, the dominant component in human perception.

4.6.1 Reconstruction Quality under Controlled Losses

In this section, we show the reconstruction quality under controlled loss scenarios for the ORB-DCT and DCT transforms. To isolate the effects due to transformations, we first eliminate quantization loss by removing quantization and dequantization in the coding process.

The left half of Table 4.3 compares the reconstruction quality of frames transformed by DCT and by ORB-DCT, assuming that video data is divided into two descriptions along horizontal directions, that only one of the descriptions is received, and that quantization effects are ignored. Results along the vertical direction are similar and are not shown. When either the odd-numbered description or the even-numbered description is received, the ORB-DCT transformed frames have consistently better performance ($1.4-1.9$ dB or $65\%-72\%$ of the reconstruction error) than the DCT transformed frames.

The top half of Table 4.4 presents the results of dividing video data into four descriptions based on recursive 2-way interleaving without quantization effects. It shows that ORB-DCT transformed frames have better quality in all cases except in Case IV. However, Case IV corresponds to losses of burst length one and is not a frequently occurring case when four descriptions are used.

Next, we show results on reconstruction quality after including quantization in the coding process. To this end, we modified the H.263 codec obtained from TenetRD (http://www.nta.no/brukere/DVC/) to use either ORB-DCT or DCT in the transforma-

tion stage. H.263 was chosen as the compression algorithm in order to keep the resulting bit rate comparable to sustained Internet bandwidth.

The right half of Table 4.3 and the lower half of Table 4.4 show the reconstruction quality for two-description and four-description coding after incorporating quantization in the coding process. As expected, the quality of frames transformed by ORB-DCT is consistently better than frames transformed by DCT, when some of the interleaved descriptions are lost. However, the gain is not as high as cases without quantization. These degradations are caused by the lossy quantization process, in which it made certain changes to the transformed pixels that are not invertible.

4.6.2 Reconstruction Quality with All Descriptions Received

As explained in Section 4.5.2, the playback quality in MDC is not as good as that in single-description coding when all the descriptions are received without errors. In this section, we show that ANN transformations can effectively compensate for compression losses in MDC.

The ANN used has eight inputs, eight outputs, and three hidden units. The weights were initialized and trained offline using back-propagation based on training tuples taken from the *missa* sequence. Table 4.5 compares the performance of the following four cases:

i) The system interleaves the original video sequence into two (resp. four) descriptions, compresses each using DCT, decompresses each, and deinterleaves the two (resp. four) descriptions.

ii) The same as above, except that the trained ANN is applied after deinterleaving the descriptions. The same ANN is applied to all the video sequences.

Table 4.3: Reconstruction quality of frames in terms of PSNR (dB) when transformed by ORB-DCT and DCT and only one of the descriptions is received under two-descriptions coding. Gain is defined as the difference in PSNR of ORB-DCT and that of DCT.

Video Sequence	No quantization effects						With quantization effects					
	Odd received			Even received			Odd received			Even received		
	DCT	ORB-DCT	Gain	DCT	ORB-DCT	Gain	DCT	ORB-DCT	Gain	DCT	ORB-DCT	Gain
missa	39.44	41.31	1.87	39.51	41.45	1.94	36.20	36.61	0.41	36.14	36.59	0.45
boxing	35.86	37.37	1.51	35.80	37.26	1.46	28.54	28.96	0.42	28.50	28.98	0.48
football	36.05	37.48	1.43	36.01	37.47	1.46	29.43	29.82	0.39	29.40	29.83	0.43
akiyo	37.12	38.50	1.38	36.95	38.53	1.58	30.27	30.60	0.33	29.97	30.59	0.62
coastguard	36.13	37.89	1.76	35.39	37.35	1.96	27.20	27.94	0.74	26.88	27.53	0.65
river	36.43	38.00	1.57	36.23	37.71	1.48	27.45	28.00	0.55	27.49	28.01	0.52

Table 4.4: Reconstruction quality in terms of PSNR (dB) in dividing video data into four description based on recursive 2-way interleaving. Case I represents the case in which three out of the four interleaved descriptions were lost; II represents the case in which two descriptions, both from the same horizontal group, were lost; III represents the case in which two descriptions, each from a different horizontal group, were lost; IV represents the case in which one out of the four interleaved descriptions was lost.

Video Sequence	Quant. Effects	Case I			Case II			Case III			Case IV		
		DCT	ORBDCT	gain	DCT	ORBDCT	gain	DCT	ORBDCT	gain	DCT	ORBDCT	gain
missa	No	35.84	37.27	1.43	39.35	39.88	0.53	39.38	41.25	1.87	42.82	42.74	-0.08
boxing		35.43	36.87	1.44	37.14	38.02	0.88	36.35	37.56	1.21	39.76	39.81	0.05
football		34.92	35.97	1.05	35.72	36.15	0.43	35.99	37.35	1.36	40.15	40.00	-0.15
akiyo		35.21	36.52	1.31	36.78	37.80	1.02	36.69	37.84	1.15	39.93	39.84	-0.09
coastguard		34.87	35.46	0.59	34.98	36.05	1.07	36.31	37.69	1.38	39.01	39.11	0.10
river		35.12	36.33	1.21	35.87	36.58	0.71	35.64	36.66	1.02	39.25	39.19	-0.06
missa	Yes	33.58	33.93	0.35	34.01	34.23	0.22	34.47	34.89	0.42	35.07	35.16	0.09
boxing		25.51	26.27	0.76	27.59	28.09	0.50	26.22	26.62	0.40	28.01	28.25	0.24
football		24.32	24.68	0.36	27.76	27.96	0.20	28.43	28.83	0.40	29.24	29.37	0.13
akiyo		28.79	29.31	0.52	29.22	29.35	0.13	29.11	29.39	0.28	29.41	29.43	0.02
coastguard		24.42	24.62	0.20	24.77	25.09	0.32	26.90	26.27	0.37	27.35	27.45	0.10
river		26.62	26.97	0.35	27.07	27.13	0.06	26.62	26.92	0.30	27.00	27.15	0.15

iii) The system interleaves the original sequence into two (resp. four) descriptions, compresses each using ORB-DCT, decompresses each, and deinterleaves the two (resp. four) descriptions.

iv) The same as above, except that the trained ANN is applied after deinterleaving the descriptions. The same ANN as in Case (ii) is applied.

Table 4.5 illustrates an average improvement of $0.26 - 0.66$ dB due to ANN-based reconstruction in comparing between Cases (i) and (ii) and between Cases (iii) and (iv). Figure 4.8 also shows that, after ANN transformation, the pixel values approach closer to the values in the original frame.

The results indicate that the pre-trained ANN based on *missa* generalizes very well to other sequences. Note that the reconstruction quality after ANN reconstruction is still worse than that of single-description coding. This is the price paid for improved error resilience and robustness.

4.6.3 Tests on the Internet

Using the prototype shown in Figure 3.11, we have tested the following strategies for transmissions on the Internet.

S1) Average reconstruction of frames transformed by ORB-DCT, if any of the interleaved descriptions is lost, or ANN-based reconstruction if all the descriptions are received;

S2) Average reconstruction of frames transformed by DCT;

S3) Decoding of frames that are single-description coded.

In the three strategies, S1 and S2 reflect MDC with error concealment, whereas S3 is the original codec. For a fair comparison under the same traffic conditions, we did trace-based

Table 4.5: Reconstruction quality when all the descriptions are correctly received in both 2-way and 4-way interleaving. Gain is defined as the difference in PSNR of ORB-DCT&NN and that of DCT. Cases (i) and (iii) show the loss introduced by compression, whereas Cases (ii) and (iv) demonstrate the added effect of incorporating an ANN transformation.

Video Sequence	Inter. Degree	Case i: DCT	Case ii: DCT&NN	Case iii: ORB-DCT	Case iv: ORB-DCT&NN	Gain in PSNR (dB)
missa	2	36.74	37.06	36.70	37.05	0.31
boxing		29.74	30.21	29.59	30.15	0.41
football		30.16	30.69	30.09	30.67	0.51
akiyo		32.60	32.99	32.54	32.92	0.32
coastguard		28.48	28.78	28.27	28.74	0.26
river		28.29	28.61	28.10	28.63	0.34
missa	4	35.53	36.09	35.43	36.02	0.49
boxing		28.45	28.96	28.50	29.11	0.66
football		29.73	30.35	29.72	30.31	0.58
akiyo		30.96	31.35	30.99	31.42	0.46
coastguard		27.78	28.14	27.62	28.05	0.27
river		27.29	27.64	27.15	27.59	0.30

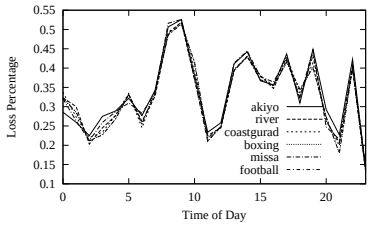
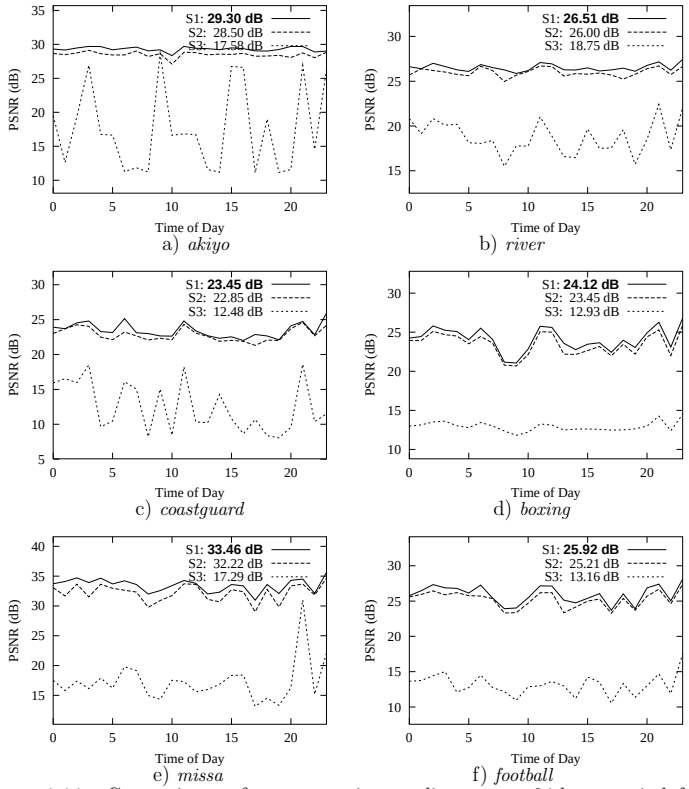


Figure 4.10: Loss rates over a 24-hour period for the Urbana-China connection.



a) *akiyo* b) *river*
c) *coastguard* d) *boxing*
e) *missa* f) *football*

Figure 4.11: Comparisons of reconstruction quality over a 24-hour period for the Urbana-China connection.

simulations by applying each of the strategies on the same trace of packets collected in real Internet transmissions (see Section 3.1).

Our experiments to apply traffic traces consist of a sender process and a receiver process. The sender process was responsible for compressing and packetizing video frames, and mapping packet losses to GOB losses of each frame. The number of descriptions (2 or 4) was set periodically every 0.5 sec at the sender according to feedback information on GOB losses of frames from the receiver. In our simulations, we assume that the receiver collected GOB loss information every 0.5 sec before sending the information to the sender, and that the network delay was constant at 0.5 sec. The receiver process was in charge of decompressing coded streams, deinterleaving them, and performing reconstruction. For every GOB of each frame, any missing information was reconstructed by average interpolation using adjacent pixels. The reconstructed frame was sent back to the decoder as a reference for future inter-coded frames. If the entire GOB was lost, it was reconstructed by copying the corresponding GOB from the last received frame.

Figures 4.10 and 4.11 compare the reconstruction quality and the corresponding loss rates over a 24-hour period for the Urbana-China connection. The loss rates of this connection range between 20% and 50% in most cases. Note that the loss rates are slightly different for the six sequences because the coded bit streams were generally mapped to different number of packets. For all the sequences, ORB-DCT, when coupled with ANN reconstruction, performs the best at all times. This strategy (S1) gives 0.5 to 1.2 dB improvement in quality when compared to average reconstruction of the original DCT streams (S2), and 8 to 16 dB improvement when compared to transmission of single-description coded streams (S3).

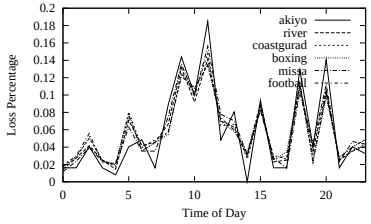


Figure 4.12: Loss rates over a 24-hour period for the Urbana-UK connection.

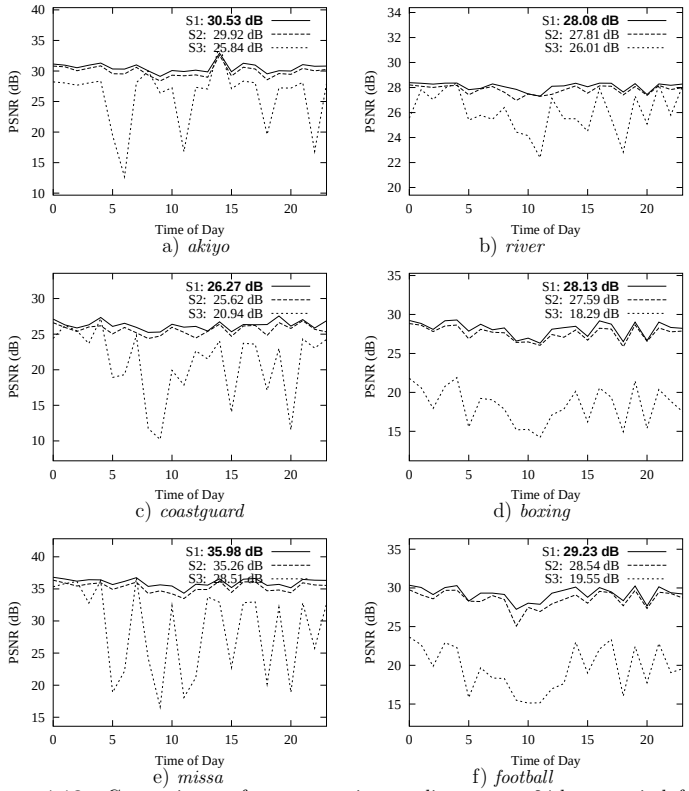


Figure 4.13: Comparisons of reconstruction quality over a 24-hour period for the Urbana-UK connection.

Figures 4.12 and 4.13 show the corresponding results on Urbana-UK connection. As expected, ORB-DCT, together with ANN reconstruction, yields the best playback quality at almost all times. The only exception is the transmission of the *akiyo* sequence at Hour 14, in which no packet losses happened, and the delivery of single-description coded streams outperforms the other two MDC-based schemes. On average, Strategy S1 gives 0.3 to 0.7 dB gain when compared to Strategy S2, and 5 to 10 dB gain when compared to Strategy S3.

Figures 4.14 and 4.15 show the corresponding comparisons of the Urbana-California connection. The loss rates of this connection are generally very low, and transmissions are even perfect at a few occasions. In these scenarios, delivery of single-description coded videos, of course, gives the best playback quality. However, the quality of such strategy drops sharply even with very low percentage of packet losses. Therefore, on average, Strategy S1 still performs the best and gives 0.2 to 0.5 dB gain when compared to Strategy S2, and 0.05 to 2.5 dB gain when compared to Strategy S3.

The above graphs show that the playback quality of single-description coded streams is very poor, although they have high PSNRs (see Table 4.2) in an error-free environment. Therefore, single-description coded streams using the original H.263 codec are not suitable for transmission in a lossy environment like the Internet.

It is interesting to note that under real loss situations, the gain of S1 and S2 for the test sequences are higher than that in the synthetic scenario in Sections 4.6.1 and 4.6.2. This is not surprising because in real tests, we always fed the reconstructed frames that were lost back to the motion-compensation loop, and the improvement of reconstruction quality due to these feedbacks accrued as the video sequence was played. In contrast, in synthetic scenarios, feedbacks were not possible since one or more streams were consistently lost.

The fraction of interleaved sets that were correctly received (those for which ANN transformations were applied), can roughly be calculated as one minus the loss rate. Our experimental results show that, for those interleaved sets that were received correctly, ANN reconstruction was performed on between 50% and 98% of the interleaved sets. These indicate that ORB-DCT performed at the sender and ANN reconstruction performed at the receiver are complementary methods that work jointly in improving playback quality.

4.7 Summary

In this chapter, we have first discussed the basic components of a very low bit rate video coding standard, H.263, which is designed to achieve maximum coding gain in lossless conditions. In transmissions over the lossy Internet, H.263 coded videos produce very poor quality because of three kinds of information loss: bitstream losses resulted from packet losses, propagation losses resulted from bitstream losses in previous frames due to temporal-difference coding, and compression losses resulted from lossy quantizations.

In this chapter, we have proposed error concealment algorithms to handle all three kinds of losses. To conceal bitstream losses, we have proposed optimized reconstruction-based DCT to optimize the final reconstruction quality when such losses happen. Our proposed ORB-DCT distinguishes from previous MDC-based approaches in that we have adopted a joint sender receiver-based approach in our design, while previous schemes only rely on inadequate capabilities of either senders or receivers. To conceal propagation losses, we have proposed to feed back the reconstructed frames to the motion compensation part of decoders. This scheme, when compared with the original H.263 decoding, effectively reduces propagation losses because the feedback in decoders enables

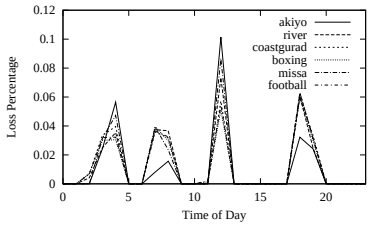


Figure 4.14: Loss rates over a 24-hour period for the Urbana-California connection.

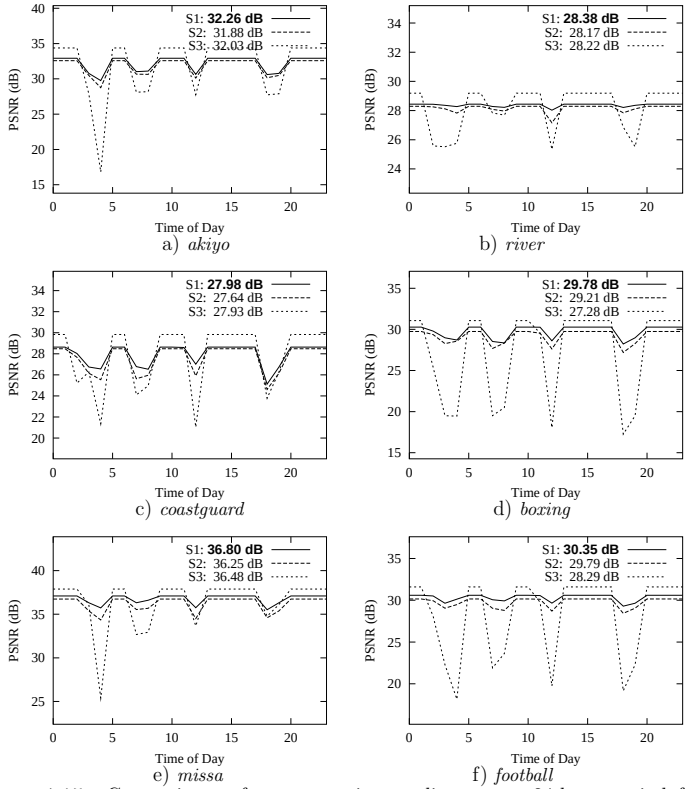


Figure 4.15: Comparisons of reconstruction quality over a 24-hour period for the Urbana-California connection.

more accurate frames to be used for the decoding of dependent frames when reference frames are lost. To conceal compression losses, we have trained an artificial neural network to model the nonlinear behavior of such losses. The training of ANNs does not need *a priori* assumption about video signals and has been shown to generalize to new video sequences.

Up to now, we have assumed that the bandwidth for video transmissions is sufficient. Clearly, this assumption is not valid for the current Internet. Therefore, in the next chapter, we discuss our modifications in quantizer modules to address the problem of bandwidth limitations.

Chapter 5

Reconstruction-Based Rate Control in H.263 Video Coding

For packet video, information loss and bandwidth limitation are two factors that affect its playback quality. In Chapter 4, we have focused on dealing with information loss alone, assuming that the desired bandwidth can be supported by underlying networks. Since this assumption is certainly too restricted, we study in this chapter rate control and allocation problems for lossy networks.

5.1 Analysis of the Problem

The general problem to be considered in this chapter is as follows: *given the available bandwidth R, how do we design an MDC in order to minimize reconstruction $\mathcal{E}_r$, subject to the rate constraint: $r \leq R$?*

As discussed before, Transform T and Quantizer Q are two very important components that can greatly affect video playback quality. However, in lossy situations, the original Transform T and Quantizer Q are not designed for optimal *reconstruction* per-

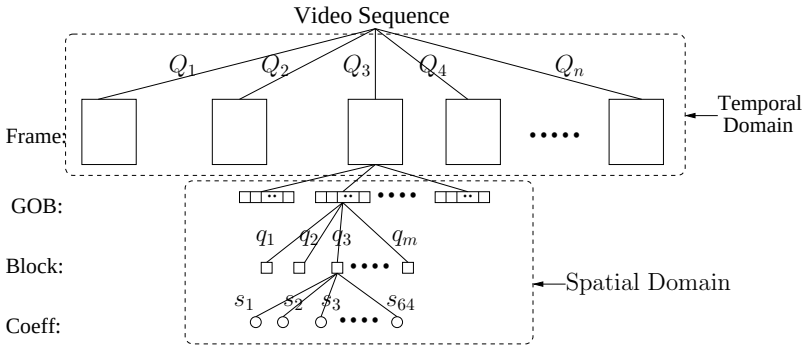
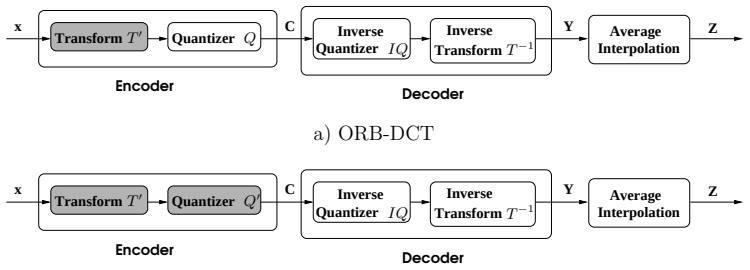


Figure 5.2: Rate allocation and control problems in H.263.

Different from existing schemes [18, 29, 43, 49, 60, 61, 66, 72] that deal with rate allocations under lossless conditions, we study rate allocations for lossy transmissions in which parts of a bit stream may get lost and need to be reconstructed. Therefore, the design of rate allocation schemes is closely related to those of multiple-description coding at a sender and the reconstruction algorithm employed at a receiver.

In the following, we focus on spatial-domain rate control approaches, because we perform reconstruction in the spatial domain in our video streaming system. We do not perform reconstruction in the temporal domain due to two reasons. First, simple interpolations of pixels at the same spatial location, (x, y), from different frames yields very poor reconstruction quality because frames carry motion information. On the other hand, performing interpolations of pixels along a motion trajectory yields better reconstruction quality, although it requires accurate knowledge of motion vectors that is not readily available at decoders. Second, reconstructions involving future frames introduce



b) H.263 codec with modified Transform T and Quantizer Q

Figure 5.1: Modified H.263 codec for reconstruction purpose.

formance. In Chapter 4, we have proposed ORB-DCT that modified only Transform T, but not Quantizer Q, as illustrated in Figure 5.1a. To add rate constraints, we need to modify both T and Q, as illustrated in Figure 5.1b. However, such a formulation involving quantization module Q is a constrained integer optimization problem and is not solvable in a closed-form. Therefore, in this chapter, we discuss heuristic approaches to address this problem.

Modifying Quantizer Q results in different rate allocations in the frame, GOB, and block levels. Figure 5.2 shows how rate control and allocations can be done in each layer. At the top frame level, rate allocations can be achieved by assigning distinct Q_is to frames. At the GOB level, rate allocations can be done by assigning different q_is to blocks within the GOB. The assignment of q_is overrides the default quantization choice set at the frame level. At the block level, rate allocations can be done by applying different s_is to coefficients within a block. Again, the value of s_i overrides the quantization choice set at the GOB level. Frame-level rate control is considered an approach in the temporal domain, whereas the other two approaches are in the spatial domain.

additional delays in playback. Based on above, we do not consider reconstruction-based rate control and allocation in the temporal-domain. Instead, we study such problems in the spatial domain, *i.e.,* both in the GOB and the block levels, while taking into account the MDC and reconstruction processes.

5.2 Reconstruction-Based Rate Allocation among Blocks in a GOB

As a GOB consists of a sequence of macroblocks, and if the total rate allocated to this GOB is constrained by a budget R, the question is how to choose quantization factors among macroblocks within the GOB in order to maximize reconstruction performance, subject to the rate constraint.

Let us start by reviewing the solution to this problem, without considering reconstruction issues. The classical solution to this problem is based on the following theorem.

Theorem 5.1 *[49] Given that the rate-distortion functions of macroblocks, $D_i(x_i), i = 1, 2, \ldots, n$, are convex, the rate allocation vector $(r_1, r_2, \ldots, r_n)$ is the solution to:*

$$\min \sum_i D_i(x_i)$$

$$s.t. \sum_i x_i \leq R$$

if and only if the following condition satisfies. $\left(\frac{\partial D_1}{\partial x_1}\right)_{r_1} = \left(\frac{\partial D_2}{\partial x_2}\right)_{r_2} = \ldots = \left(\frac{\partial D_n}{\partial x_n}\right)_{r_n}$

The proof can be found in [49], and the discrete version of the theorem can be found in [66]. Essentially, the derivatives $\left(\frac{\partial D_i}{\partial x_i}\right)_{r_i}, i = 1, 2, \ldots, n$, are the slopes of lines tangent to the R-D curves of the macroblocks at rates $r_i, i = 1, 2, \ldots, n$. For this reason,

the algorithm implementing the theorem is normally referred to as "constant slope optimization." The intuitive idea behind the algorithm is very simple. At those points with constant slope, all the macroblocks operate at the same marginal return for an extra bit in the rate-distortion trade-off. In other words, If we reduce one bit for macroblock i, and spend it on another macroblock j (to maintain the same bit rate), then the reduction in distortion of macroblock j would be equal to the increase in distortion of macroblock i. For this reason, there is no allocation that is more efficient than this rate budget.

This theorem establishes the necessray and sufficient conditions for optimal rate allocations among macroblocks. In practice, a bisection search [56, 66] can be used to find the correct slope at the required rate. The detailed algorithm is outlined in Figure 5.3.

This algorithm assumes that the R-D characteristics of all the macroblocks are available. Normally, they are obtained offline by computing all the operating points from a block or by interpolating from existing ones.

For SDC, the rate-distortion functions $D_i(x_i)$ in Theorem 5.1 is computed based on the distortion between the decompressed and the original signals. Its R-D curves are well known to be convex; hence, the above algorithm can be applied to achieve optimal rate allocations. For MDC, if we incorporate a reconstruction process in the end, the distortion is calculated between the decompressed and reconstructed signals and the original signals. The R-D curves defined in this way have not been studied before. Next, we establish empirically the properties of the R-D curves for MDC coders with reconstruction.

To this end, we first modified the MDC-based H.263 codec in such a way that the reconstruction quality after interpolation and the corresponding bit rate spent on each macroblock were saved for each description, for a given quantization choice. Then we iterated through all possible quantization choices, *i.e.,* $2, 3, \ldots, 31$, and obtained 30 rate-

1. Find the maximum slope, λ_{max}.
2. Find the minimum slope, λ_{min}.
3. Choose an intial slope, λ_0.
4. Calculate the total consumed rate r at slope λ_0.
5. **while** $(r \neq R)$ **do**
6. **if** $(r < R)$ **then**
7. $\lambda_{i+1} = (\lambda_i + \lambda_{min})/2$.
8. $\lambda_{max} = \lambda_i$.
9. **elseif** $(r > R)$ **then**
10. $\lambda_{i+1} = (\lambda_i + \lambda_{max})/2$.
11. $\lambda_{min} = \lambda_i$.
12. **else**
13. Desired λ is found, exit.
14. **end-if**
15. Calculate the total consumed rate r, at slope λ_i.
16. **end-while**

Figure 5.3: Bisection search to find the correct λ for a specified rate.

distortion pairs, that resulted in a rate-distortion (R-D) curve for each macroblock. From the experiments, we have found that all the intra-coded macroblocks and a majority of the inter-coded macroblocks have convex R-D curves. Some inter-coded macroblocks have non-convex R-D curves due to their complex dependencies on the R-D curves of their reference macroblocks. To save space, we only show the R-D curves of four randomly chosen intra-coded macroblocks and four inter-coded macroblocks from each video sequence. Figure 5.4 (resp. 5.5), 5.6 (resp. 5.7), 5.8 (resp. 5.9), 5.10 (resp. 5.11), 5.12 (resp. 5.13), and 5.14 (resp. 5.15) plot the R-D curves of intra-blocks (resp. inter-

blocks) taken from *missa*, *football*, *boxing*, *akiyo*, *coastguard* and *river*, respectively. In these graphs, rate is measured in bytes, and distortion is calculated in terms of mean squared error. Although for some video sequences, their curves are not convex in certian small local regions, convexity is still observed in most parts of all R-D curves. As a result, we conclude that the R-D relationship for reconstructed macroblocks in MDC is approximately convex; therefore, previous approaches that address optimal allocations among macroblocks [49, 56, 66] can still be applied in MDC with reconstruction.

5.3 Design of Quantization Matrices for MDC

H.263 uniformly quantizes every coefficient in a block by applying the same quantization factor q. Intuitively, this simple scheme is not optimal because it does not exploit the characteristics of individual coefficients. The objective of our work is to improve its performance for MDC by assigning proper quantization factors to different coefficients.

As quantization is done in the coefficient domain after DCT transform, we need to first relate errors introduced in the coefficient domain to those observed in the pixel domain. Let X and X' denote the original and the reconstructed blocks of pixels, and Y and Y' be the corresponding original and reconstructed blocks of transformed coefficients, we have the following relationship between the errors in these two domains.

$$\begin{aligned} \| Y - Y' \|^2 &= \| TX - TX' \|^2 && (5.1)\\ &= (X - X')^T T^T T (X - X') && (5.2)\\ &= (X - X')^T (X - X') \quad \text{if and only if} \quad T^T T = I && (5.3)\\ &= \| X - X' \|^2 . && (5.4) \end{aligned}$$

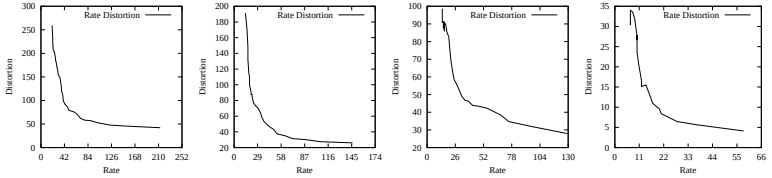


Figure 5.8: Rate-distortion curves of four randomly chosen macroblocks from an I-frame of *boxing*.

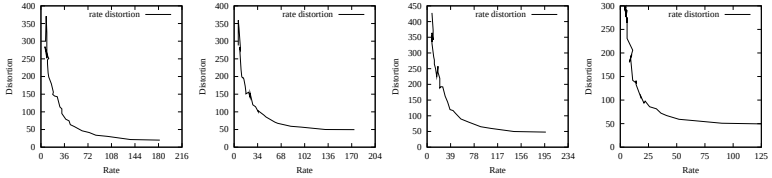


Figure 5.9: Rate-distortion curves of four randomly chosen macroblocks from a P-frame of *boxing*.

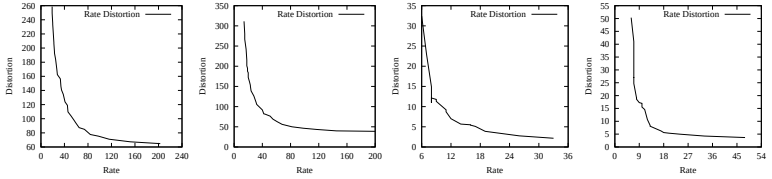


Figure 5.10: Rate-distortion curves of four randomly chosen macroblocks from an I-frame of *akiyo*.

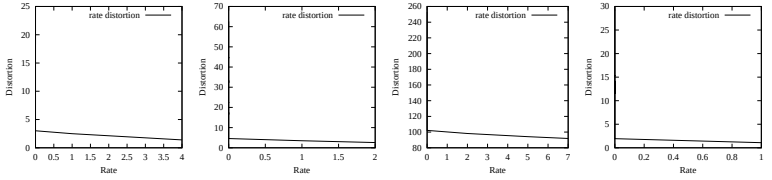


Figure 5.11: Rate-distortion curves of four randomly chosen macroblocks from a P-frame of *akiyo*.

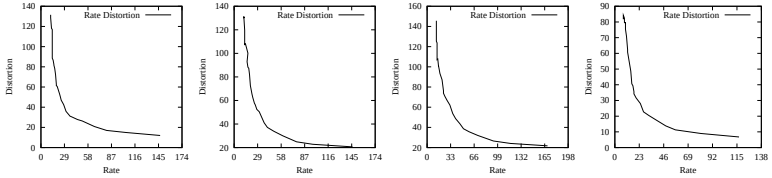


Figure 5.4: Rate-distortion curves of four randomly chosen macroblocks from an I-frame of *missa*.

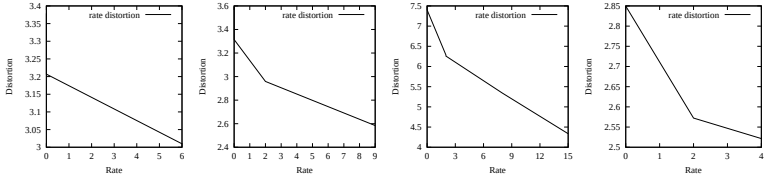


Figure 5.5: Rate-distortion curves of four randomly chosen macroblocks from a P-frame of *missa*.

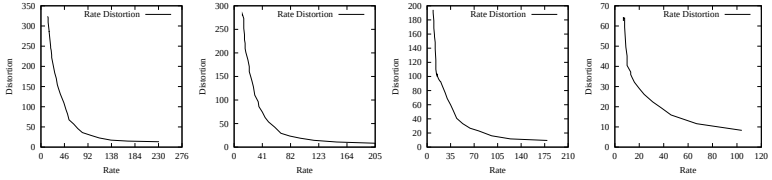


Figure 5.6: Rate-distortion curves of four randomly chosen macroblocks from an I-frame of *football*.

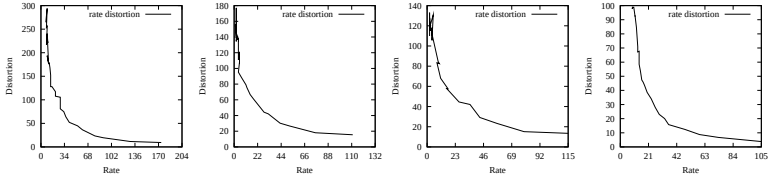


Figure 5.7: Rate-distortion curves of four randomly chosen macroblocks from a P-frame of *football*.

(5.2) and (5.3) hold if and only if T is an orthonormal matrix. It is easy to verify that DCT is an orthonormal transform; therefore, the energy of quantization errors in DCT transform coefficients is equal to that of image pixels. This is a useful property because it implies that our efforts to reduce quantization errors are equally reflected in the pixel domain as well.

Next, we need to identify within a coefficient block, the kinds of coefficients that should be quantized more finely than others. As said before, this is a bit-allocation problem in the block level. Essentially, we need to solve a constrained minimization problem formulated as follows [34]:

$$\mathcal{E}_r = \frac{1}{n}\sum_{i=1}^{n} {d_i}^2 \tag{5.5}$$

$$s.t. \quad \sum_{i=1}^{n} R_i = R. \tag{5.6}$$

In the above formulation, we assume that i^{th} coefficient is quantized to R_i bits, and the resulting quantization error is d_i. This problem can be solved by Lagrangian optimization. First, we define the Lagrangian function

$$L(R_1, R_2, \ldots, R_n, \lambda) = \frac{1}{n}\sum_{i=1}^{n} {d_i}^2 + \lambda(\sum_{i=1}^{n} R_i - R). \tag{5.7}$$

By employing an empirical formula found in DPCM system [34] that relates distortion, d_i, to bit rate, R_i, and coefficient variance, σ_i, the Lagrangian function can be rewritten as follows:

$$L(R_1, R_2, \ldots, R_n, \lambda) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{2}\alpha^2 2^{-2R_i}\sigma_i^2 + \lambda(\sum_{i=1}^{n} R_i - R), \tag{5.8}$$

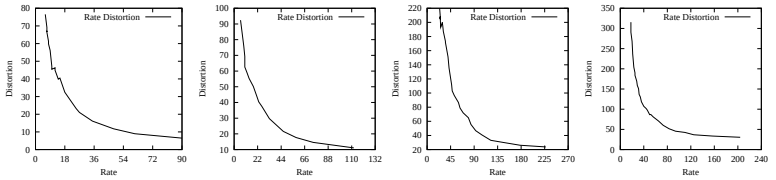


Figure 5.12: Rate-distortion curves of four randomly chosen macroblocks from an I-frame of *coastguard*.

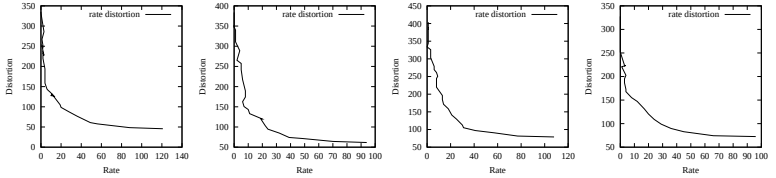


Figure 5.13: Rate-distortion curves of four randomly chosen macroblocks from a P-frame of *coastguard*.

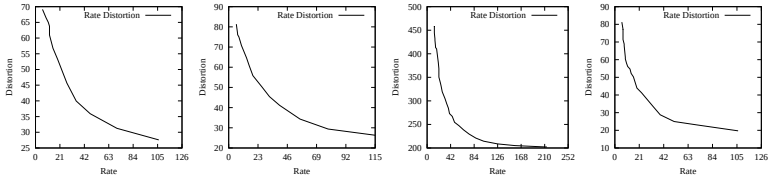


Figure 5.14: Rate-distortion curves of four randomly chosen macroblocks from an I-frame of *river*.

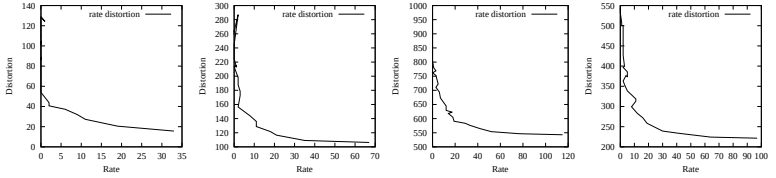


Figure 5.15: Rate-distortion curves of four randomly chosen macroblocks from a P-frame of *river*.

where α is a constant. The optimal bit allocation vector $(R_1, R_2, \ldots, R_n)$ can be derived by setting the derivatives of the Lagrangian function to zero.

$$\frac{\partial L(R_1, R_2, \ldots, R_n, \lambda)}{\partial R_i} = 0 \implies R_i = \frac{1}{2}\log_2 \frac{\alpha\sigma_i^2}{n\lambda} \tag{5.9}$$

$$\frac{\partial L(R_1, R_2, \ldots, R_n, \lambda)}{\partial \lambda} = 0 \implies \sum_i^n R_i = R. \tag{5.10}$$

Substituting (5.9) in (5.10), we can eliminate λ and arrive at the final solution for R_i [34].

$$R_i = \frac{R}{n} + \frac{1}{2}\log_2 \frac{\sigma_i^2}{\sqrt[n]{\sigma_1^2\sigma_2^2\ldots\sigma_n^2}} \qquad i = 1, 2, \ldots n. \tag{5.11}$$

The equation indicates that bit allocation should be done based on coefficient variances. If all the coefficients have equal variances, *i.e.*, $\sigma_1^2 = \sigma_2^2 = \ldots = \sigma_n^2$, then the best way is to assign $\frac{R}{n}$ bits to each coefficient. On the other hand, if the variance of a certain coefficient, σ_i^2, is greater (or smaller) than the geometrical average of the variances, then the number of bits allocated to this coefficient should be greater (or smaller) than $\frac{R}{n}$, the average number of bits for each coefficient.

Although mathematically elegant, it is difficult to directly apply this closed-form solution in practice. First, real-time estimations of the variance for each coeffcient is very expensive, since video coding is already computationally intensive. Second, due to the non-stationarity of video frames, the variance of a coeffcient changes from frame to frame; hence, a quantization matrix needs to be sent for each frame, leading to additional bit overhead that may not justify the bit savings resulted from this approach. Third, the largest obstacle to the application of this formula is that the relationship between rate and quantization factors cannot be derived in advance due to the zigzag ordering and variable length coding employed. As they have large impact on the resulted bit rate but

cannot be formulated in closed-form, it is hard to find the optimal quantization choices given the knowledge of this optimal bit allocation vector.

Solution 5.11, however, still provides guidelines for us to develop practical heuristics in designing efficient quantization schemes for MDC. Since we know that coefficients should be quantized according to their variances, our first step is to study how the MDC process changes the variances of individual coeffcients. For this purpose, we group coefficients from a video frame into 64 bands by putting coefficients with the same coordinate (i, j) in a transformed block, $i, j = 1, 2, \ldots, 8$, into the same band, and calculate the variances of coefficients within each band. We do this separately for intra-coded and inter-coded frames because they have different inputs: intra-coded frames code the original pixel values, whereas inter-coded frames code the residual signals computed from the current and its reference frames.

Table 5.1 shows the ratio of coefficient variances of a horizontally-interleaved MDC system as compared to those of a SDC system, for CIF and QCIF sequences, respectively. The coefficients having smaller variances after MDC are circled in ovalboxes.

The results tell us that the variances in the upper right part of a coefficient block tend to increase after MDC, and those in the lower left tend to decrease after MDC. This is not surprising because horizontal (resp. vertical) frequency components are likely to increase (resp. decrease) after horizontal partitioning, and the coefficients in the upper right (resp. lower left) triangle are the ones that capture horizontal (resp. vertical) frequencies. As we know that coefficients with large variances need to be quantized more finely than those with smaller variances, our observation natually leads to the following

Table 5.1: Ratio of coefficient variances of a horizontally-interleaved MDC system compared to those of a SDC system, for intra-coded and inter-coded blocks from *missa*, *football*, *boxing*, *akiyo*, *coastguard*, and *river*, respectively.

a) *missa*: intra block

	1	2	3	4	5	6	7	8
1	0.95	2.01	3.35	4.14	4.09	2.85	2.56	1.61
2	0.84	1.15	2.43	3.39	2.62	2.62	2.59	1.38
3	0.68	1.50	1.16	1.58	2.08	1.99	2.87	1.36
4	0.60	1.27	1.11	1.91	1.34	1.75	1.89	1.22
5	0.58	1.12	1.18	1.22	1.34	1.13	2.32	1.05
6	0.51	1.46	1.26	1.34	1.24	1.22	1.73	1.02
7	0.06	0.87	1.18	0.78	1.27	0.33	0.80	1.16
8	0.08	0.86	1.48	1.03	1.14	0.46	1.07	1.23

b) *missa*: inter block

	1	2	3	4	5	6	7	8
1	1.88	1.51	1.44	2.00	2.59	1.54	2.60	1.39
2	1.25	1.30	1.46	1.65	2.31	2.07	1.51	1.21
3	1.20	1.14	1.37	1.80	2.16	1.63	1.87	1.29
4	0.87	1.03	1.20	2.05	1.19	1.73	1.66	1.41
5	0.73	1.50	1.45	1.38	1.03	1.39	1.79	1.06
6	0.71	1.05	1.10	1.29	0.97	0.93	1.62	1.30
7	0.05	0.91	0.88	1.04	1.46	0.41	0.89	5.23
8	0.06	0.90	1.35	1.33	1.30	0.49	1.01	3.27

c) *football*: intra block

	1	2	3	4	5	6	7	8
1	0.92	2.50	2.64	1.91	2.16	2.62	5.37	9.41
2	0.79	1.21	2.07	1.77	2.85	3.61	5.19	8.26
3	0.66	1.02	1.45	2.25	2.93	3.94	4.63	7.67
4	0.59	0.81	1.45	1.98	3.51	4.42	5.16	6.56
5	0.66	0.91	1.76	1.91	3.10	5.09	6.15	7.47
6	0.66	0.99	2.01	2.52	3.83	5.27	6.94	7.87
7	0.65	1.11	1.49	2.48	4.15	6.36	7.11	12.12
8	0.59	0.96	1.41	1.62	3.47	8.09	11.64	13.29

d) *football*: inter block

	1	2	3	4	5	6	7	8
1	1.50	1.69	2.12	2.25	1.88	4.06	5.56	9.51
2	1.00	1.13	1.32	1.68	2.34	4.36	8.81	13.85
3	0.83	1.02	1.06	1.64	2.65	4.20	9.01	14.18
4	0.73	0.91	1.14	1.64	2.17	3.73	9.43	14.76
5	0.85	0.88	1.35	1.36	2.85	6.12	8.68	13.24
6	0.67	1.24	1.64	1.83	2.94	5.86	8.64	14.54
7	0.62	1.38	1.35	1.22	3.62	9.52	10.97	10.20
8	0.69	0.87	1.12	1.16	2.22	5.35	10.18	17.76

e) *boxing*: intra block

	1	2	3	4	5	6	7	8
1	0.93	1.71	1.81	2.38	4.14	4.29	6.87	12.06
2	0.86	1.33	1.41	1.48	2.19	2.93	12.53	26.26
3	0.85	1.41	1.14	1.23	2.11	3.43	9.65	21.26
4	0.75	1.26	1.21	1.29	1.81	4.32	7.51	20.18
5	0.77	1.37	0.88	0.81	2.16	3.51	10.31	22.09
6	0.74	1.09	1.06	1.29	1.73	3.28	6.38	12.34
7	0.67	0.93	0.80	1.58	1.69	4.07	3.00	7.12
8	0.71	0.88	0.97	0.76	0.52	2.97	3.09	6.60

f) *boxing*: inter block

	1	2	3	4	5	6	7	8
1	1.03	2.40	2.46	2.00	1.95	4.84	5.80	14.38
2	0.80	1.39	1.67	1.54	2.03	4.70	7.55	14.99
3	0.76	0.57	1.04	1.76	1.79	3.70	6.79	19.15
4	1.13	0.41	0.88	1.06	1.48	3.13	5.24	23.38
5	1.05	0.98	1.28	0.86	1.49	2.79	6.34	15.59
6	0.71	0.93	1.02	1.14	1.37	2.36	5.40	11.11
7	0.79	0.85	1.04	1.18	1.20	2.51	4.21	9.12
8	0.73	0.16	0.22	0.36	0.55	2.12	4.59	7.69

Table 5.1: (Cont'd)

g) *akiyo*: intra block

	1	2	3	4	5	6	7	8
1	0.94	1.62	2.15	1.65	2.28	3.65	6.57	16.70
2	0.76	1.38	0.81	0.65	2.06	2.45	4.91	20.84
3	0.69	1.12	0.98	0.75	1.03	1.87	4.45	16.49
4	0.62	0.74	1.30	1.28	1.88	1.30	2.39	6.22
5	0.38	0.82	0.95	1.92	1.25	2.00	2.97	4.93
6	0.55	0.31	0.74	1.25	3.12	2.69	3.54	7.67
7	0.56	0.40	0.30	1.09	2.58	6.88	2.87	5.98
8	0.43	0.55	0.40	1.21	0.91	4.64	0.93	19.92

h) *akiyo*: inter block

	1	2	3	4	5	6	7	8
1	0.95	1.06	0.85	1.18	1.72	1.73	2.01	3.97
2	1.14	1.08	1.13	1.37	1.69	1.70	1.81	2.66
3	0.99	1.04	1.06	1.16	1.19	1.44	1.58	3.53
4	1.12	1.41	0.96	0.79	1.23	1.45	1.50	2.93
5	1.10	1.13	1.05	0.82	1.10	1.48	1.85	2.91
6	1.03	0.88	0.76	0.91	2.10	1.78	2.54	4.73
7	0.96	0.84	1.23	0.80	1.57	2.84	2.61	6.63
8	0.88	0.55	0.49	1.08	1.63	5.81	3.38	16.56

i) *coastguard*: intra block

	1	2	3	4	5	6	7	8
1	0.92	2.43	1.24	2.67	5.63	1.31	15.62	20.19
2	0.90	1.35	1.36	1.60	1.60	4.19	5.64	14.89
3	0.79	1.37	1.71	1.52	2.52	2.94	5.89	13.51
4	0.86	1.07	0.97	1.04	1.64	3.07	9.42	21.87
5	0.81	1.03	0.92	2.06	1.39	2.99	4.22	12.83
6	0.70	1.10	1.29	1.14	1.16	2.76	6.43	23.45
7	0.73	0.86	1.18	1.29	2.12	2.31	6.22	13.45
8	0.70	1.05	0.95	1.57	1.70	2.66	3.60	19.49

j) *coastguard*: inter block

	1	2	3	4	5	6	7	8
1	0.87	1.51	1.17	1.40	1.85	3.23	2.37	7.20
2	0.88	1.07	1.12	1.42	1.88	2.72	6.18	18.12
3	0.91	1.04	1.28	1.38	1.99	1.89	4.36	9.09
4	0.96	1.19	0.97	1.38	1.85	2.35	6.27	13.06
5	0.91	0.88	1.25	1.42	1.50	2.03	4.24	19.45
6	0.83	1.12	1.30	0.93	1.79	2.48	5.66	19.75
7	0.81	0.94	1.30	1.10	2.09	2.37	4.99	17.76
8	0.81	1.26	1.08	1.67	1.59	2.45	5.55	15.39

k) *river*: intra block

	1	2	3	4	5	6	7	8
1	0.97	2.35	1.08	1.16	1.21	1.28	1.97	2.73
2	0.87	1.21	1.92	1.07	0.94	0.90	1.36	2.02
3	0.79	1.18	0.93	1.09	1.22	2.12	1.20	2.05
4	0.75	0.94	0.91	1.02	0.92	1.58	2.10	2.41
5	0.71	1.72	1.02	0.83	0.56	1.14	2.39	2.47
6	0.69	1.16	1.17	0.62	0.86	0.78	1.17	4.36
7	0.63	1.32	1.10	1.16	0.84	0.84	1.33	1.55
8	0.83	1.34	0.82	0.84	1.19	1.02	1.59	0.90

l) *river*: inter block

	1	2	3	4	5	6	7	8
1	1.07	1.37	1.60	1.10	1.42	1.09	1.09	1.18
2	0.88	0.93	1.48	1.42	1.14	1.02	1.06	1.16
3	0.96	1.04	1.30	1.22	1.43	1.05	1.23	1.61
4	1.14	0.97	1.20	1.32	1.23	1.25	1.52	1.68
5	0.93	1.15	1.21	1.12	1.35	1.27	1.63	1.60
6	0.95	1.08	1.21	0.95	1.12	0.91	1.48	1.73
7	0.85	0.86	0.87	1.28	0.84	0.87	1.00	1.70
8	0.80	1.10	0.96	0.99	1.14	1.18	1.20	1.44

quantization scheme for MDC:

$$Q_{i,j} = \begin{cases} \alpha Q & i \geq j \\ \beta Q & i < j \end{cases} \qquad \alpha \geq 1,\ \beta \leq 1,$$

where $Q_{i,j}$ is the quantization factor to be used for the coefficient of row i and column j, α and β are scaling parameters, and Q is the original quantization choice for this block. To choose suitable α and β, we have evaluated the following combination of choices: $\alpha = 1.0, 1.05, \ldots, 1.2$ and $\beta = 0.7, 0.75, \ldots, 1$, for each video sequence.

The best results along with the parameters and the comparisons with the original quantization scheme can be found in Table 5.2. Here, R_s (resp. R_o) represents the bit rate resulted from the scaled (resp. original) quantization, measured in bytes, and $PSNR_s$ (resp. $PSNR_o$) denotes PSNR values for the scaled (resp. original) approach. From the results on $\Delta PSNR$ ($= PSNR_s - PSNR_o$) and $\frac{\Delta R}{R}$ ($= \frac{R_s - R_o}{R_o}$), we can see that the modified quantization scheme lead to better PSNRs and $1\% - 10\%$ savings in bit rates for *missa*, *football*, *boxing*, *akiyo*, and *river*, and comparable R-D results for *coastguard*.

In our approach, since the same scaling factors are used throughout a video sequence, there is no overhead in bit rate when compared to approaches that need to send frame-based quantization matrices to decoders. Furthermore, the estimations of variances and scaling factors, α and β, do not add much extra complexity in real-time encoding because they can be done offline. However, moderately more compuations are still needed in encoding because the quantization and de-quantization processes become floating point operations after scaling.

Table 5.2: Comparisons of bit rates and PSNRs of scaled quantization and original quantization for *missa*, *football*, *boxing*, *akiyo*, *coastguard* and *river*, respectively.

a) *missa*: one description received ($\alpha = 0.9$, $\beta = 1.0$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	520142	39.30	505814	39.37	-2.75%	0.07
8	155992	37.87	140943	37.93	-9.65%	0.06
12	81494	36.74	79162	36.89	-4.01%	0.15
16	52638	35.96	51285	36.01	-1.27%	0.05
20	38244	35.22	37814	35.26	-1.09%	0.04

b) *missa*: two descriptions received ($\alpha = 0.9$, $\beta = 1.0$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	1041431	39.70	1007594	39.83	-3.25%	0.13
8	312739	37.94	283405	38.04	-9.38%	0.10
12	161445	36.79	158298	36.93	-1.95%	0.14
16	104896	35.94	102589	36.00	-2.20%	0.06
20	75707	35.21	75190	35.22	-0.68%	0.01

c) *football*: one description received ($\alpha = 0.95$, $\beta = 1.05$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	1370410	34.02	1313080	34.08	-4.18%	0.06
8	686309	31.80	664489	31.83	-3.18%	0.03
12	429069	30.14	417638	30.15	-2.66%	0.01
16	297435	28.94	292659	28.96	-1.61%	0.02
20	222290	28.02	219458	28.05	-1.27%	0.03

d) *football*: two descriptions received ($\alpha = 0.95$, $\beta = 1.05$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	2739181	35.60	2624672	35.73	-4.18%	0.13
8	1371836	32.25	1328216	32.32	-3.18%	0.07
12	857213	30.23	834977	30.28	-2.59%	0.05
16	594469	28.88	585938	28.93	-1.44%	0.05
20	443592	27.91	438298	27.96	-1.19%	0.05

Table 5.2: (Cont'd)

e) *boxing*: one description received ($\alpha = 0.95$, $\beta = 1.05$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	6433505	32.96	6202908	32.99	-3.58%	0.03
8	3372707	31.37	3299831	31.40	-2.16%	0.03
12	2248014	29.99	2215691	30.05	-1.44%	0.06
16	1660040	28.86	1653551	28.90	-0.39%	0.04
20	1302048	27.93	1301098	27.96	-0.07%	0.03

f) *boxing*: two descriptions received ($\alpha = 0.95$, $\beta = 1.05$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	12873887	35.16	12408811	35.32	-3.61%	0.16
8	6744048	32.30	6599903	32.42	-2.14%	0.12
12	4493241	30.37	4428943	30.47	-1.43%	0.10
16	3316086	28.98	3303632	29.07	-0.38%	0.09
20	2600602	27.90	2598681	27.97	-0.07%	0.07

g) *akiyo*: one description received ($\alpha = 0.9$, $\beta = 1.0$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	166766	33.18	148173	33.24	-11.2%	0.06
8	84907	32.08	81653	32.23	-3.83%	0.15
12	49612	31.09	47693	31.26	-3.87%	0.17
16	39536	30.25	35720	30.46	-9.65%	0.21
20	29138	29.48	27075	29.58	-7.08%	0.10

h) *akiyo*: two descriptions received ($\alpha = 0.9$, $\beta = 1.0$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	338150	36.09	297852	36.26	-11.9%	0.17
8	166909	33.78	164726	34.04	-1.31%	0.26
12	99810	32.10	94733	32.38	-5.09%	0.28
16	78012	30.87	70867	31.20	-9.16%	0.33
20	58506	29.87	52890	30.05	-9.60%	0.18

Table 5.2: (Cont'd)

i) *coastguard*: one description received ($\alpha = 0.95$, $\beta = 1.05$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	870368	32.74	838909	32.71	-3.61%	-0.03
8	434397	30.85	421569	30.89	-2.95%	0.04
12	268977	29.34	269558	29.40	0.02%	0.06
16	182715	28.22	180423	28.17	-1.25%	-0.05
20	133379	27.34	131247	27.31	-1.60%	-0.03

j) *coastguard*: two descriptions received ($\alpha = 0.95$, $\beta = 1.05$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	1732269	34.70	1670505	34.71	-3.57%	0.01
8	864098	31.62	837878	31.62	-3.03%	0.00
12	534677	29.65	523749	29.63	-2.04%	-0.02
16	363121	28.35	358719	28.31	-1.21%	-0.04
20	264748	27.37	261419	27.35	-1.26%	-0.02

k) *river*: one description received ($\alpha = 0.95$, $\beta = 1.0$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	891271	32.58	886953	32.63	-0.05%	0.05
8	422859	31.08	419573	31.15	-0.08%	0.07
12	252516	29.97	250526	30.04	-0.08%	0.07
16	170231	29.14	168197	29.17	-0.12%	0.03
20	125177	28.50	124459	28.52	-0.06%	0.02

l) *river*: two descriptions received ($\alpha = 0.95$, $\beta = 1.0$)

Quant Factor	R_o	$PSNR_o$	R_s	$PSNR_s$	$\Delta R/R_o$	$\Delta PSNR$
4	1781375	33.99	1773424	34.10	-0.04%	0.11
8	845460	31.63	838596	31.75	-0.08%	0.12
12	505124	30.18	500135	30.29	-0.10%	0.11
16	339817	29.21	336550	29.27	-0.10%	0.06
20	250665	28.50	248950	28.54	-0.07%	0.04

5.4 Summary

In this chapter, we have studied reconstruction-based rate control schemes to address the problem of bandwidth limitations. Again, the objective here is to optimize reconstruction quality when losses happen.

In general, rate control can be formulated as integer programming problems. Since it is difficult to derive signal-independent closed-form solutions to such problems, we have developed heuristic approaches to do rate control in two levels. First, for rate control among blocks within a GOB, we have studied schemes based on the "constant slope theorem," which basically states that the optimal rate allocation vector can be found at points with constant slopes in rate-distortion curves. To apply this theorem, one needs to verify an important assumption, *i.e.,* the R-D relationship for each individual block is convex. It has been found that conventional SDC generates convex R-D curves, but no results have been reported about MDC coders with reconstructions. Our work has filled this gap by verifying empirically the convexity of R-D curves for MDC coders with reconstructions. As a result, conventional approaches based on the "constant slope theorem" can still be used for MDC coders. Second, for rate control among coefficients within a block, we have first investigated the property of coefficient variances for MDC coders. Then, based on the observations about the change of variances, we have proposed a scaled quantization scheme that can produce videos with higher PSNRs using smaller bandwith.

The schemes proposed in this chapter, together with those in the last chapter, complete our study on transmissions of H.263 coded videos over the low-bandwith Internet.

Chapter 6

Coding and Transmissions of Subband-Coded Images

As subband image coding emerges as the core technology in the JPEG2000 standard [48], more images transmitted in the World Wide Web will be coded using this technique. Quality-delay trade-offs can be made in transmitting subband-coded images in the Internet by using either the TCP or the UDP protocol. Delivery by TCP gives superior decoding quality but with very long delays when the network is unreliable, whereas delivery by UDP has negligible delays but with degraded quality when packets are lost. Although images are delivered primarily by TCP today, we study in this chapter the use of UDP to deliver multi-description reconstruction-based subband-coded images and the reconstruction of missing information at the receiver based on information received. First, we give a brief overview of subband-based image coding. Next, we propose a joint sender-receiver approach for designing *optimized reconstruction-based subband transform* (ORB-ST) in multi-description subband coding, similar to the design of ORB-DCT in H.263 coding. Finally, we study the delay and quality trade-offs involved in the TCP delivery of single-description coded (SDC) images and the UDP delivery of multiple-

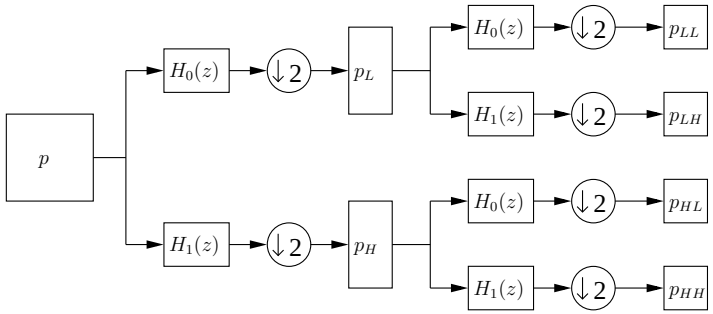


Figure 6.1: 2D forward wavelet transform.

elegant framework in which the properties of both smooth and active image data can be analyzed and represented with equal importance. This is the reason why wavelet transform has attracted much attention in the image processing community in the past decade.

Both forward and inverse wavelet transform can be efficiently implemented by a pair of symmetric quadrature mirror filters (QMFs), as described in [89]. Each QMF pair consists of a lowpass filter ($H_0(z)$) and a highpass filter ($H_1(z)$) that split a signal's bandwidth in half.

Figure 6.1 illustrates a 2-D forward wavelet transform of image p, which is composed of two consecutive 1-D transforms. Image p is first filtered and downsampled along the x dimension, resulting in a lowpass image p_L and a highpass image p_H. In order to keep the same data rate, the downsampling operation is accomplished by dropping every other filtered value. Next, both p_L and p_H are filtered and downsampled along the y direction, producing four subimages, p_{LL}, p_{LH}, p_{HL}, and p_{HH}. Among the four subimages, p_{LL} represents the average of the image signal, p_{LH}, the image details with large vertical

description coded images. The key observations made in our evaluations motivate us to further design a hybrid TCP/UDP delivery approach that performs better than previous approaches by either TCP or UDP alone.

6.1 Review of Subband-Based Image Coding

Wavelet-based image coding is chosen as the JPEG2000 standard [48], not only because of its superior coding performance, but also because of its capability to generate an embedded bit stream in which encoding can be stopped at any point in order to allow a target bit rate to be met exactly.

Similar to DCT-based coding, the encoding process of JPEG2000 also consists of three steps: wavelet-based image transformation, quantization, and entropy coding. Wavelet-based image coding, usually implemented via subband decomposition, is also called subband image coding. This is the term that we will use throughout this chapter. Next, we describe these three steps in detail.

6.1.1 Wavelet-Based Image Transformation

One drawback with block-based transform coding, like H.263, is that a fixed block size is not always good for different kinds of image areas. For example, in a smooth area, signals have a wide band in the spatial domain and localized in the frequency domain; thus, a large block size is desirable to capture trends. In contrast, in an edge area, signals are more localized in the spatial domain and tend to have a wide band in the frequency domain; thus, a small block size is desirable to capture transient information. The choice of a fixed block size has to trade the capability to represent "trends" for the ability to capture "transients." On the other hand, the theory of wavelet transform provides an

frequency, p_{HL}, the image details with large horizontal frequency, and p_{HH}, the diagonal features.

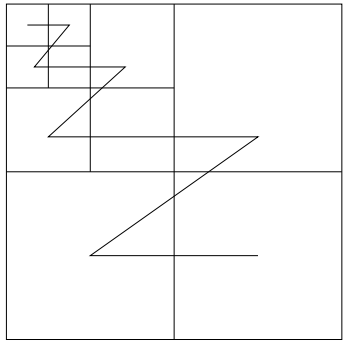


Figure 6.2: The layout of an image after three level decomposition.

To improve transform efficiency, it is necessary to recursively apply the same 2-D decomposition process to the lowest frequency band. In general, the higher the desired compression ratio, the more times the transform is applied. But at the same time, the number of transformations is also limited by factors such as residual correlations, computing resources, and available time. In practice, an image is normally decomposed for 3 to 5 times before presented to the quantization stage. Figure 6.2 shows the layout of an image after three levels of 2-D decompositions and the zig-zag ordering of subbands from low to high frequency. This scanning order is commonly used in the encoding of coefficients.

Each coefficient in level one subbands, p_{LL}, p_{LH}, p_{HL}, and p_{HH}, represents approximately a spatial area of size 2×2. In each successive level, the coefficients carry information of a larger spatial area: an area of 4×4 for coefficients in level two, an area of

8×8 for coefficients in level three, and so on. Therefore, the coefficients of low frequency bands (*i.e.*, the bands at the beginning of the zig-zag order) characterize "trends" from a large spatial area, and the coefficients of high frequency bands (*i.e.*, the bands at the end of the zig-zag order) capture "transient" information from a small area. Hence, a wavelet transform is able to achieve a delicate balance between the representation of both "trends" and "transients."

6.1.2 Quantization

As stated before, the purpose of quantization is to discard insignificant coefficients that are not important to our visual perception. We can exploit the energy invariance property of orthonormal subband transforms in order to achieve high-quality lossy compressions. The energy invariance property states that any change in the energy of subband coefficients is equal to the change of energy in pixel values. In other words, we can zero out those coefficients with small magnitude without creating large distortions in the reconstructed image.

To achieve the best results, a separate quantizer needs to be designed for each individual subband, based on the properties of the human visual system [33] and the statistical distributions of coefficients. In general, we shall apply low quantization factors to low-frequency bands, and large quantization factors to high-frequency bands. This is needed because low-frequency coefficients, when compared to with high-frequency ones, make larger contributions to the overall energy and, thus, are more important to human perception.

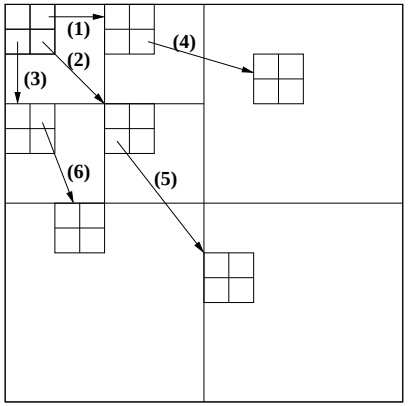


Figure 6.3: Relationship of coefficients in a spatial orientation tree.

Since a large percentage of coefficients must be zero after quantization in low-bit rate coding, a large portion of bit budget is spent on coding the significance map, *i.e.*, the locations of significant coeffcients. To efficiently generate and code the significance map, a special data structure called "spatial orientation tree" is introduced. Figure 6.3 illustrates a spatial orientation tree in the layout of a 2-level subband decomposition. In the figure, links represent parent-child relationships in a spatial orientation tree. To construct a spatial orientation tree, the coefficients in the lowest frequency band at the upper left corner of Figure 6.3 are linked to the coefficients of the other three high-pass subbands at the same level as their parent node, as indicated by arrows (1), (2) and (3). Next, the coefficient at location (i, j) from subband level l are linked to four pixels at $(2i, 2j)$, $(2i + 1, 2j)$, $(2i, 2j + 1)$, and $(2i + 1, 2j + 1)$ of subband level $l + 1$ as their parent node, as indicated by arrows (4), (5) and (6). This process is repeated to link coeffcients from the lowest to the highest level in the subband hierarchy. SPIHT

Currently, there does not exist a standard quantization algorithm in subband-based image coding. Different algorithms use specialized approaches specifically tuned in their settings [24, 44, 62, 65, 76, 90, 91].

6.1.3 Entropy Coding

The task of entropy coding is to compress a sparse array of subband coefficients losslessly. One can consider entropy coding being carried out in two steps. First, it needs to convert the quantizer output to an intermediate representation, with the objective that the entropy of this intermediate representation is smaller than that of the original quantizer output. Second, it codes the intermediate symbols using conventional entropy coding techniques, such as Huffman coding and arithmetic coding.

While the second step has nothing new to discuss, the first step has spurred a lot of research interests, and a wide variety of schemes exist [42]. In fact, advanced techniques developed in this step are among the key reasons why wavelet-based coders outperform block DCT-based coders [92]. Of particular interest is a scheme called "set partitioning in hierarchical trees" (SPIHT) developed by Said and Pearlman [62]. This algorithm generates an embedded coded bitstream that allows both encoding and decoding to terminate at any point for a target bit rate to be met exactly. Intuitively, a non-embedded coder should have better coding gains because it is not constrained by conditions that a bitstream needs to be embedded. However, this is not true in practice, and embedded coders [62, 65] are currently the best coders for subband-based image coding. Because of its superior coding quality and implementation efficiency, SPIHT is used in our experiments. In the following, we describe how SPIHT processes its quantized coefficients in order to generates an embedded bitstream.

tests significance information based on the spatial orientation trees, which effectively explore the self-similarity of coefficients across subband levels. Self-similarity states that if a coefficient is insignificant with respect to a threshold, then its descendants in the spatial orientation tree are highly likely to be insignificant as well. Therefore, in most situations, one bit suffices to convey the information when the coefficients constituting a spatial orientation tree are all insignificant. This effectively reduces the cost for encoding significance information.

After constructing the spatial orientation tree, SPIHT encodes coefficients that are significant with respect to a threshold, and the threshold value is reduced by half at every step. For each threshold value, the coding process consists of two passes: the sorting pass and the refinement pass. The sorting pass involves selecting the coefficients that lie in the range $2^m \leq |c(i,j)| \leq 2^{m+1}$, where m is an integer. This pass divides coefficients into significant sets and insignificant sets. The refinement pass involves coding the m^{th} most significant bit of all the coefficients that are greater than threshold 2^{m+1}. Since m decreases at each successive step in the coding process, the coefficients that contribute to the largest reduction in error energy are selected to code before those that have smaller contributions, thereby generating an embedded bitstream. It is interesting to note that SPIHT can also be used as a lossless coder because m can be arbitrarily small.

6.2 Issues in the Delivery of Subband Coded Images

Quality and delay are two key performance measures to evaluate the delivery of images. Previously, high quality in delivery is considered more important because image data is not real time in nature and is generally sent using a reliable transport protocol like TCP.

With the advent of the World Wide Web, trade-offs between quality and delay in transferring image data may need to be changed. Oftentimes, when there are multiple images to be transferred from a Web server, users may prefer to see (slightly) degraded images in (much) faster turnaround time than to wait for a long time to see high-quality images. TCP delivery in such cases is not desirable because it incurs intolerable long delays by using coarse grained timeout periods (500 ms) and exponential backoffs of sending rates when congestion happens. This is evident in Web surfing, as one frequently experiences long stalls in downloading images. On the other hand, UDP delivery incurs shorter end-to-end delays but cannot be used for sending conventionally coded images because dependencies may render these images not decode-able when losses happen.

Based on the above analysis, one can see that there exist two key issues in studying the delivery of subband coded images. The first problem is how to design an efficient multiple-description coding method in order to facilitate the reconstruction of lost streams when they are encapsulated in UDP packets. We address this issue by mathematically deriving an *optimized reconstruction-based subband transform* that maximizes reconstruction performance when some of the descriptions are lost and reconstructed using average interpolation. The second issue involves the choice of suitable delivery strategies for image transmissions on the Internet. To address this problem, we evaluate experimentally quality-delay trade-offs of various delivery schemes. In the next section, we describe our approaches in detail.

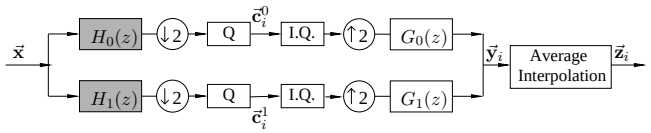


Figure 6.4: Basic building blocks of a modified codec. (The shaded block is our proposed ORB-ST.)

6.3 UDP Delivery of Multi-Description Coded Images

In our MDC system, we first *interleave* adjacent pixels of an image into multiple descriptions, decompose each description into segments so that each segment fits in a packet, code each segment, place the coded streams into packets, and then transmit them to the destination. Again, in this process, a simple scheme that codes interleaved streams may not work well because the original subband transforms and quantizer are not necessarily the best for reconstructing lost streams. In this section we propose a new optimized reconstruction-based subband transform (ORB-ST) that takes into account the reconstruction process at the receiver.

Figure 6.4 shows the basic building blocks in our proposed subband image coding system. It is based on existing state-of-the-art images codecs that consist of several stages: the subband transformation stage that divides image data into components with different frequency contents, the quantizer that causes a controlled loss of information based on frequency information, and an optional entropy coder that removes residual redundancies among quantized symbols.

In a subband transform system, one can represent the filtering operations as equivalent linear transformations in spatial domain. Here, we use $\hat{\mathsf{H}}$ to denote $H_0(z)$ and $H_1(z)$ with down-sampling, and similarly $\hat{\mathsf{G}}$ for $G_0(z)$ and $G_1(z)$ with up-sampling. Our goal

is to find a new analysis system $\hat{\mathsf{H}}'$ ($H_0'(z)$ and $H_1'(z)$ with down-sampling) in order to minimize reconstruction error $\mathcal{E}_r$ after average interpolation, based on fixed quantization Q, inverse quantization IQ, and the original synthesis system, $\hat{\mathsf{G}}$ ($G_0(z)$ and $G_1(z)$ with up-sampling), where

$$\mathcal{E}_r \;=\; \| \underbrace{Interpolate(\hat{\mathsf{G}}(IQ(\vec{\mathbf{c}})))}_{decompression+reconstruction} - \vec{\mathbf{x}} \|^2 . \tag{6.1}$$

In order to keep our decoders standard-compliant so that existing decoders at receivers can be used, we assume fixed inverse quantization IQ and synthesis system $\hat{\mathsf{G}}$.

With quantization in place, the minimization of $\mathcal{E}_r$ becomes an integer optimization problem, where $\vec{\mathbf{c}}$ in (6.1) takes integer values. Such optimizations are too expensive to be done in real time. Similar to the derivation of ORB-DCT, we derive an approximate solution that does not take into account quantization effects. Specifically, the objective to be optimized in the approximation is:

$$\mathcal{E}_r \;=\; \| Interpolate(\hat{\mathsf{G}}(\vec{\mathbf{c}})) - \vec{\mathbf{x}} \|^2 . \tag{6.2}$$

The resulting transform is called optimized reconstruction-based subband trasform (ORB-ST). In the following, we derive ORB-ST based on partitioning image data into two descriptions; extensions to four descriptions can be done similarly as described in Section 4.3.3.

6.3.1 Deriving ORB-ST

Assume that each row of the original image, $\vec{\mathbf{x}}$ of size n, is transformed into $\vec{\mathbf{c}}_1$ of size $\frac{n}{2}$ and $\vec{\mathbf{c}}_2$ of size $\frac{n}{2}$, corresponding to the descriptions of odd-numbered and even-numbered

pixels. Here, $\vec{\mathbf{c}}_i$, $i = 1, 2$, is an interleaved vector of components from $\vec{\mathbf{c}}_i^0$ and $\vec{\mathbf{c}}_i^1$, for $i = 1, 2$, where $\vec{\mathbf{c}}_i^j$ is the output from subband j, and subbands are ordered from low to high frequencies. Our objective is to find $\vec{\mathbf{c}}_1$ and $\vec{\mathbf{c}}_2$ in order to minimize $\mathcal{E}_r$. Since the derivations are similar, we only show those for $\vec{\mathbf{c}}_1$.

As explained above, the synthesis system, consisting of up-sampling, $G_0(z)$ and $G_1(z)$, is equivalent to a linear transform, $\hat{\mathsf{G}}$, in spatial domain. Let $g_0(i)$ $(i = -N_1, -N_1 + 1, \ldots, M_1 - 1, M_1)$ and $g_1(j)$ $(j = -N_2, -N_2 + 1, \ldots, M_2 - 1, M_2)$ denote the filter coefficients of G_0 and G_1, respectively. Then $\hat{\mathsf{G}}$ can be written in the following form:

$$\hat{\mathsf{G}} = \begin{pmatrix} & & & \vdots & & \vdots & & \vdots & & \\ \cdots & g_0(-2) & g_1(-2) & g_0(0) & g_1(0) & g_0(2) & g_1(2) & g_0(4) & g_1(4) & \cdots \\ \cdots & g_0(-3) & g_1(-3) & g_0(-1) & g_1(-1) & g_0(1) & g_1(1) & g_0(3) & g_1(3) & \cdots \\ \cdots & g_0(-4) & g_1(-4) & g_0(-2) & g_1(-2) & g_0(0) & g_1(0) & g_0(2) & g_1(2) & \cdots \\ \cdots & g_0(-5) & g_1(-5) & g_0(-3) & g_1(-3) & g_0(-1) & g_1(-1) & g_0(1) & g_1(1) & \cdots \\ & & & \vdots & & \vdots & & \vdots & & \end{pmatrix}$$

$$= \begin{pmatrix} & \vdots & \vdots & \vdots & \vdots & \\ \cdots & S_{-1} & S_0 & S_1 & S_2 & \cdots \\ \cdots & S_{-2} & S_{-1} & S_0 & S_1 & \cdots \\ & \vdots & \vdots & \vdots & \vdots & \end{pmatrix}, \tag{6.3}$$

where $S_i = \begin{pmatrix} g_0(2i) & g_1(2i) \\ g_0(2i-1) & g_1(2i-1) \end{pmatrix}$.

Then after synthesis filtering, output $\vec{\mathbf{y}}_1$ is calculated as:

$$\vec{\mathbf{y}}_1 = \hat{\mathsf{G}}\vec{\mathbf{c}}_1. \tag{6.4}$$

The set of interpolated pixels $\vec{\mathbf{z}}_1$ is obtained by inserting even-numbered columns as the average of columns from $\vec{\mathbf{y}}_1$, with the boundary column duplicated:

$$\vec{\mathbf{z}}_1 = \begin{pmatrix} & \vdots & \vdots & \vdots & \vdots & \\ \cdots & 1 & 0 & 0 & 0 & \cdots \\ \cdots & 0.5 & 0.5 & 0 & 0 & \cdots \\ \cdots & 0 & 1 & 0 & 0 & \cdots \\ \cdots & 0 & 0.5 & 0.5 & 0 & \cdots \\ & \vdots & \vdots & \vdots & \vdots & \end{pmatrix} \vec{\mathbf{y}}_1$$

$$= \begin{pmatrix} & & \vdots & \vdots & & \\ \cdots & B_0 & B_1 & & & \cdots \\ \cdots & & B_0 & B_1 & & \cdots \\ \cdots & & & B_0 & B_1 & \cdots \\ & & \vdots & \vdots & & \end{pmatrix} \vec{\mathbf{y}}_1 \tag{6.5}$$

$$= \mathsf{U}\vec{\mathbf{y}}_1, \tag{6.6}$$

where $B_0 = \begin{pmatrix} 1 & 0 \\ 0.5 & 0.5 \\ 0 & 1 \\ 0 & 0.5 \end{pmatrix}$ and $B_1 = \begin{pmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0.5 & 0 \end{pmatrix}$.

Then distortion between the original and the received and reconstructed pixels becomes:

$$\mathcal{E}_r = \left\| \mathsf{U}\hat{\mathsf{G}}\vec{\mathbf{c}}_1 - \vec{\mathbf{x}} \right\|^2 = \left\| \mathsf{P}\vec{\mathbf{c}}_1 - \vec{\mathbf{x}} \right\|^2. \tag{6.7}$$

Since the linear system of equations $\mathsf{P}\vec{\mathbf{c}}_1 = \vec{\mathbf{x}}$ is an over-determined one, there exists at least one least-square solution $\vec{\mathbf{c}}_1$ that minimizes (6.7), according to the theory of linear algebra [54]. Specifically, the solution $\vec{\mathbf{c}}_1$ with the smallest length $|\vec{\mathbf{c}}_1|^2$ can be found by

respectively.

$$\mathsf{T}_1 = (\vec{\mathbf{a}}_1 \quad \vec{\mathbf{a}}_2 \quad \ldots \quad \vec{\mathbf{a}}_{\frac{n}{2}})^T$$

$$\mathsf{T}_2 = (\vec{\mathbf{b}}_1 \quad \vec{\mathbf{b}}_2 \quad \ldots \quad \vec{\mathbf{b}}_{\frac{n}{2}})^T,$$

where $\vec{\mathbf{a}}_i$ and $\vec{\mathbf{b}}_i$ are the transpose of row vectors in each transform. First, we interleave row vectors of T_1 and T_2 to give a combined transform, T, as follows:

$$\mathsf{T} = (\vec{\mathbf{a}}_1 \quad \vec{\mathbf{b}}_1 \quad \vec{\mathbf{a}}_2 \quad \vec{\mathbf{b}}_2 \quad \ldots \quad \vec{\mathbf{a}}_{\frac{n}{2}} \quad \vec{\mathbf{b}}_{\frac{n}{2}})^T.$$

The inverse of T is the transform that will be applied at the receiver when both descriptions are available. In practice, perfect reconstruction is not always achievable due to errors introduced by truncations of floating point numbers as well as quantization effects.

6.3.2 UDP Delivery of ORB-ST Coded Images

As ORB-ST coded images are resilient to packet losses, they can be delivered by unreliable transport protocol like UDP.

An important consideration here is how to packetize coded images to limit loss propagations. In our coding schemes, each of the coded descriptions is too large to fit into a single packet and must be decomposed into distinct packets. If we simply divide a single coded description into multiple packets, then the loss of a packet can render subsequent packets useless because the decoder has a mismatch in its control paths when decoding significance maps. Further, there is no synchronization units, such as GOBs of H.263, in a coded image.

Hence, we need to decompose heuristically an image into segments in such a way that a coded segment can fit in a packet, that each description from the coded segment can be

first performing SVD decomposition of matrix P:

$$\mathsf{P} = \mathbf{S}\,[diag(w_j)]\,\mathbf{D}^T, \qquad j = 1, 2, \ldots, \frac{n}{2}, \tag{6.8}$$

where $\mathbf{S}$ is an $n \times \frac{n}{2}$ column-orthogonal matrix, $[diag(w_j)]$, an $\frac{n}{2} \times \frac{n}{2}$ diagonal matrix with positive or zero elements (singular values), and $\mathbf{D}$, an $\frac{n}{2} \times \frac{n}{2}$ orthogonal matrix. Then the least-square solution can be expressed as:

$$\vec{\mathbf{c}}_1 = \mathbf{D}\,[diag(1/w_j)]\,\mathbf{S}^T\,\vec{\mathbf{x}}. \tag{6.9}$$

In the above diagonal matrix $[diag(1/w_j)]$, element $1/w_j$ is replaced by zero if w_j is zero. Therefore, ORB-ST is a product of three matrices: $\mathbf{D}$, $[diag(1/w_j)]$, and $\mathbf{S}^T$.

To derive the ORB-ST transform for $\vec{\mathbf{c}}_2$, simply replace B_0 and B_1 in (6.5) by:

$$B_0 = \begin{pmatrix} 0 & 0.5 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \qquad B_1 = \begin{pmatrix} 0.5 & 0 \\ 1 & 0 \\ 0.5 & 0.5 \\ 0 & 1 \end{pmatrix}.$$

The rest of the steps are similar.

In our derived ORB-ST, P in (4.11) is a product of two matrices, where U and $\hat{\mathsf{G}}$ are both block circulant matrices, as evident in (6.3) and (6.5). Hence, P is also block circulant. It is easy to verify that the least-square solution of a block circulant matrix is still block circulant [17]. This implies that our derived ORB-ST still takes the form of a linear filter.

When both $\vec{c}_1$ and $\vec{c}_2$ are available at the receiver, we can derive an inverse transform to achieve perfect reconstruction. Let T_1 and T_2 represent ORB-ST for $\vec{c}_1$ and $\vec{c}_2$,

decoded independent of other coded segments, and that the total bit rate is unchanged from that of SDC. As a simple solution, we divide an image into equal-size segments in such a way that each coded segment fits in a single packet, and that each segment is coded using the same bit rate. Since our strategies of using equal-size segments and the coding of each using the same bit rate are suboptimal, we expect losses in image quality when compared to MDC without segmentation. We plan to study in the future how to better segment images to reduce the quality degradations.

6.3.3 Experimental Results

In this section, we compare the performance of ORB-ST and the original subband transform (ST) in two scenarios: a synthetic scenario under controlled losses and real Internet tests.

We carried out our experiments using four test images: *barbara*, *goldhill*, *peppers*, and *lena*, and evaluated the reconstruction quality by PSNR.

6.3.3.1 Reconstruction Quality under Controlled Losses

In this section, we study the reconstruction quality under controlled-loss scenarios for ORB-ST and the original ST. To isolate the effects due to ST and ORB-ST, we first eliminate quantization losses by removing quantization and dequantization in the coding process.

The first four rows of Table 6.1 compare the reconstruction quality of frames transformed by ST and by ORB-ST, assuming that image data is divided into two descriptions along horizontal directions, and that quantization effects are ignored. Under these controlled conditions, data in one description is assumed to be received without loss; there-

fore, there is no need to perform segmentation, since segmentation is only introduced to reduce loss propagations within a description by creating additional synchronization points. Results along the vertical direction are similar and are not shown. When either the odd-numbered or even-numbered description is received, the ORB-ST transformed frames have consistently better quality ($1.46-1.74$ dB or $81\%-85\%$ of the reconstruction error) than that of the ST transformed frames. When both descriptions are available, we omit the results in the table because perfect reconstruction is achieved and the PSNR values for both ST and ORB-ST are infinite.

Similarly, the top four rows of Table 6.2 present the results of dividing image data into four descriptions by recursive 2-way interleaving without quantization. It shows that ORB-ST transformed frames have better quality in all cases except in Case IV. However, Case IV corresponds to losses of burst length one and should be infrequent when using four descriptions.

Next, we show results on reconstruction quality after including quantization effects.

The image codec used is an implementation based on SPIHT [62] that was downloaded from `http://qccpack.sourceforge.net` (see a description in Section 6.1.3). Throughout the experiments, we have used the Daubechies 9/7 filters [8] in ST and the ORB-ST derived from them.

The bottom parts of Tables 6.1 and 6.2 show the reconstruction quality for two-description and four-description coding after incorporating quantization in the coding process. The three bit rates tested correspond to transmitting 16, 32 and 64 512-byte packets. When some descriptions are lost, the quality of reconstructed frames transformed by ORB-ST is better than that by ST for all cases under 2-description coding, and for most cases under 4-description coding, with a few exceptions for *barbara* and

Table 6.1: Reconstruction quality of frames in PSNR (dB) when transformed by ORB-ST and ST along the horizontal direction and only one of the descriptions is received under two-descriptions coding. Gain is defined as the difference in PSNR between ORB-ST and ST, and boxed numbers represent positive gains.

Image	Quant. Effects	bit rate	Odd Received			Even Received			Both Received		
			ST	ORB-ST	Gain	ST	ORB-ST	Gain	ST	ORB-ST	Gain
barbara	No	-	25.21	26.67	1.46	25.16	26.62	1.46	perfect reconstruction		
goldhill	No	-	32.58	34.05	1.47	32.64	34.12	1.48	perfect reconstruction		
peppers	No	-	31.60	33.24	1.64	31.23	32.69	1.46	perfect reconstruction		
lena	No	-	34.11	35.83	1.72	34.19	35.93	1.74	perfect reconstruction		
barbara	Yes	0.25	23.41	24.25	0.84	23.38	24.24	0.86	25.78	25.67	-0.11
		0.50	24.35	25.31	0.96	24.30	25.26	0.96	27.94	27.82	-0.12
		1.0	24.92	25.98	1.06	24.87	25.93	1.06	32.88	32.80	-0.08
goldhill	Yes	0.25	27.88	28.33	0.45	27.90	28.34	0.44	28.70	28.64	-0.06
		0.50	29.55	30.21	0.66	29.54	30.22	0.66	31.33	31.17	-0.16
		1.0	31.09	32.14	1.05	31.09	32.16	1.07	34.89	34.65	-0.24
peppers	Yes	0.25	29.61	29.92	0.31	29.39	29.62	0.23	30.50	30.51	0.01
		0.50	30.53	31.23	0.70	30.22	30.89	0.67	31.67	31.68	0.01
		1.0	31.06	32.17	1.11	30.70	31.73	1.03	34.49	34.30	-0.19
lena	Yes	0.25	29.91	30.63	0.72	29.97	30.73	0.76	30.91	30.99	0.08
		0.50	32.12	33.18	1.06	32.19	33.27	1.08	34.59	34.44	-0.15
		1.0	33.33	34.70	1.37	33.41	34.79	1.38	37.71	37.50	-0.21

Table 6.2: Reconstruction quality in PSNR (dB) when image data is divided into four descriptions by recursive 2-way interleaving. Case I represents the case in which three out of the four interleaved descriptions were lost; II, the case in which two descriptions, both from the same horizontal group, were lost; III, the case in which two descriptions, each from a different horizontal group, were lost; IV, the case in which one out of the four interleaved descriptions was lost; and V, the case in which none of the four descriptions was lost. Boxed numbers represent positive gains.

Image	Bit Rate	Case I			Case II			Case III			Case IV			Case V		
		ST	ORB-ST	Gain	ST	ORB-ST	Gain	ST	ORB-ST	Gain	ST	ORB-ST	Gain	ST	ORB-ST	Gain
barbara	-	25.05	25.82	0.77	25.21	26.67	1.46	27.12	27.93	0.81	29.66	29.31	-0.35	perfect reconstruction		
goldhill	-	30.01	31.33	1.32	32.58	34.05	1.47	31.90	32.45	0.55	34.34	35.41	1.07	perfect reconstruction		
peppers	-	30.07	31.27	1.20	31.60	33.24	1.64	30.83	31.92	1.09	33.04	34.10	1.06	perfect reconstruction		
lena	-	32.86	34.25	1.39	34.10	35.83	1.73	36.12	37.43	1.31	36.23	36.41	0.18	perfect reconstruction		
barbara	0.25	22.70	23.39	0.69	22.66	23.45	0.79	23.34	23.66	0.32	23.95	23.78	-0.17	24.34	24.15	-0.19
	0.50	23.61	24.46	0.85	23.57	24.62	1.05	24.66	25.32	0.66	25.90	25.71	-0.19	26.89	26.91	0.02
	1.0	24.48	25.34	0.86	24.52	25.64	1.12	26.06	26.84	0.78	28.01	27.45	-0.56	31.31	31.29	-0.02
goldhill	0.25	26.23	26.79	0.56	26.55	26.91	0.36	26.60	26.91	0.31	26.75	26.97	0.22	28.02	27.96	-0.06
	0.50	27.63	28.42	0.79	28.31	28.89	0.58	28.45	28.68	0.23	28.84	29.04	0.20	30.42	30.25	-0.17
	1.0	28.86	29.92	1.06	30.22	31.14	0.92	30.14	30.46	0.32	31.15	31.54	0.39	32.83	33.62	-0.21
peppers	0.25	27.70	28.20	0.50	28.08	28.53	0.45	28.12	28.42	0.30	28.44	28.65	0.21	28.81	28.78	-0.03
	0.50	29.06	29.71	0.65	29.71	30.36	0.65	29.63	30.05	0.42	30.31	30.61	0.30	30.93	30.85	-0.08
	1.0	29.64	30.59	0.95	30.67	31.78	1.11	30.27	31.02	0.75	31.57	32.19	0.62	34.14	33.96	-0.18
lena	0.25	27.94	28.46	0.52	28.02	28.51	0.49	28.22	28.55	0.33	28.33	28.58	0.25	28.41	28.60	0.19
	0.50	30.24	31.05	0.81	30.51	31.29	0.78	31.29	31.55	0.26	31.58	31.61	0.03	32.03	31.94	-0.09
	1.0	31.98	33.10	1.12	32.67	33.86	1.19	33.96	34.58	0.62	34.32	34.38	0.06	35.73	35.42	-0.31

interpolation. When an entire interleaved set is lost, it is filled by the average of image pixels.

Figure 6.5 (resp. Figures 6.7 and 6.9) plots the loss rates of traces over a 24-hour period when sending 16, 32, and 64 packets at the beginning of each hour to the remote UDP echo ports of China (resp. UK and California). Figure 6.6 (resp. Figures 6.8 and 6.10) compares the reconstruction quality of sending four test images using the traces obtained, when each image was coded at, respectively, 0.25 bpp, 0.5 bpp and 1 bpp and put into 16, 32 and 64 packets for transmission.

For the Urbana-China connection, ORB-ST outperforms ST at all bit rates for all four images, with an average of 0.31 to 0.38 dB better for the 0.25-bpp case, 0.25 to 0.48 dB better for the 0.5-bpp case, and 0.54 to 0.86 dB better for the 1-bpp case. Quality gain improves with increasing bit rates when there is less quantization noise. When entire interleaved sets were lost at certain hours, quality degraded significantly (such as hours 9, 11, 17 and 19 at 1 bpp).

For the Urbana-UK connection, the average reconstruction quality based on ORB-ST is better than that of ST most of the time, with only two exceptions: *goldhill* and *lena* at 0.5 bpp. For the Urbana-California connection, the reconstruction quality of the two schemes are comparable. In these two cases, the gain of performing ORB-ST is, in general, not as much as in the Urbana-China connection because the gain of performing ORB-ST is offset by degradations when all the descriptions are received under low loss rates.

These results lead us to conclude that ORB-ST is more suitable for the delivery of images over unreliable channels than the original ST.

goldhill. However, when all the descriptions are received, the quality of ORB-ST transformed frames is, in general, not as good as that of original ST transformed ones. As explained before, cases with no loss and those in which three out of the four descriptions were received should occur infrequently when MDC was used. Moreover, since degradations are not as high as gains when losses happen, we should expect an improvement in average quality.

The tables show less gain for cases with quantization when compared to those without quantization. As pointed out in Section 6.3, these degradations were caused by the lossy quantization process in which it made certain changes to the transformed pixels that were not invertible. Although an inverse process exists for ORB-ST, quantization errors in the coded bit stream make it hard to achieve perfect reconstruction in practice. The tables also show improved gains in quality with increasing bit rates.

6.3.3.2 Tests in the Internet

To further evaluate our proposed schemes, we built a prototype (see Figure 3.11) and tested the quality of frames reconstructed by linear interpolation of adjacent pixels received when the original frame was either ST transformed or ORB-ST transformed. For a fair comparison under the same traffic conditions, we did trace-driven simulations by applying reconstructions on the trace of packets received in real Internet transmissions (see Chapter 3).

Our trace-driven simulations involved a sender process and a receiver process. The sender process was responsible for coding an image, packetizing it, and mapping packet losses to the losses of coded descriptions. The receiver process was in charge of decompressing coded streams, deinterleaving them, and performing reconstruction by linear

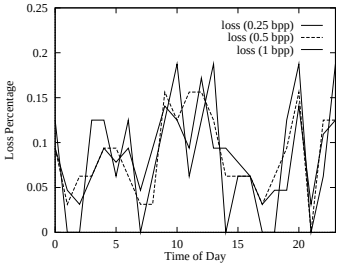
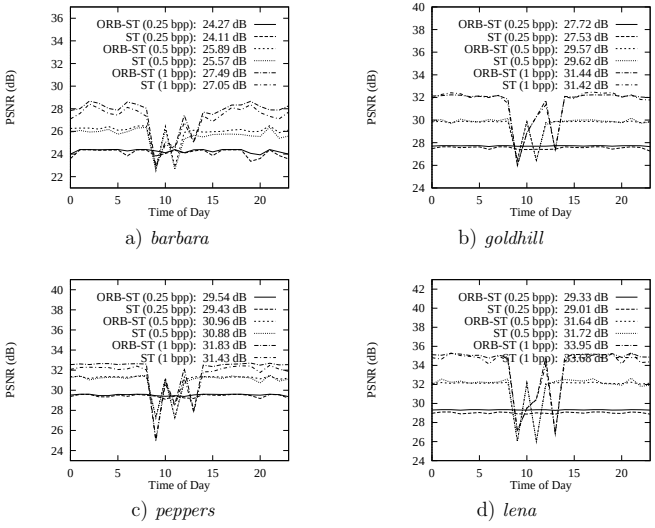


Figure 6.7: Loss rates of 16-, 32- and 64-packet transmissions between Urbana and UK.



a) *barbara* b) *goldhill*

c) *peppers* d) *lena*

Figure 6.8: Comparisons of reconstruction quality over a 24-hour period for the Urbana to UK connection, when each image was coded at, respectively, 0.25 bpp, 0.5 bpp and 1 bpp, and placed into 16, 32 and 64 packets for transmission.

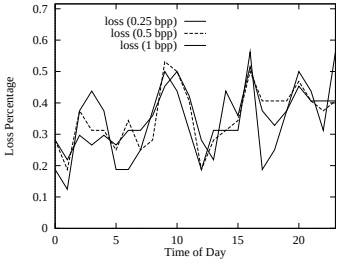
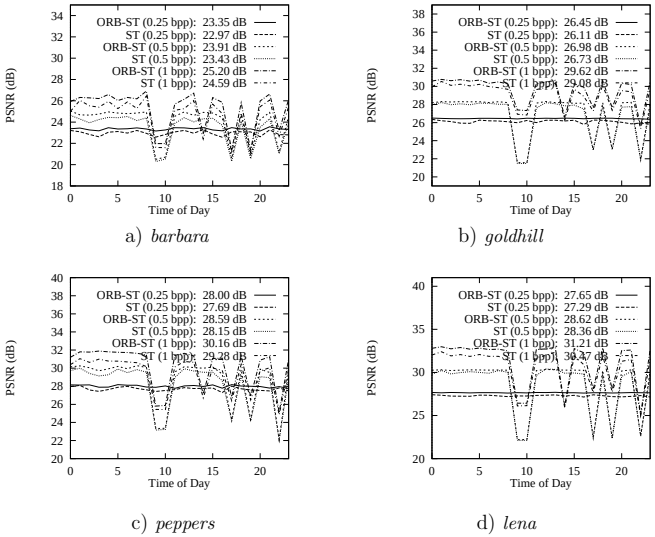


Figure 6.5: Loss rates of 16-, 32- and 64-packet transmissions between Urbana and China.



a) *barbara* b) *goldhill*

c) *peppers* d) *lena*

Figure 6.6: Comparisons of reconstruction quality over a 24-hour period for the Urbana to China connection, when each image was coded at respectively, 0.25 bpp, 0.5 bpp and 1 bpp, and placed into 16, 32 and 64 packets for transmission.

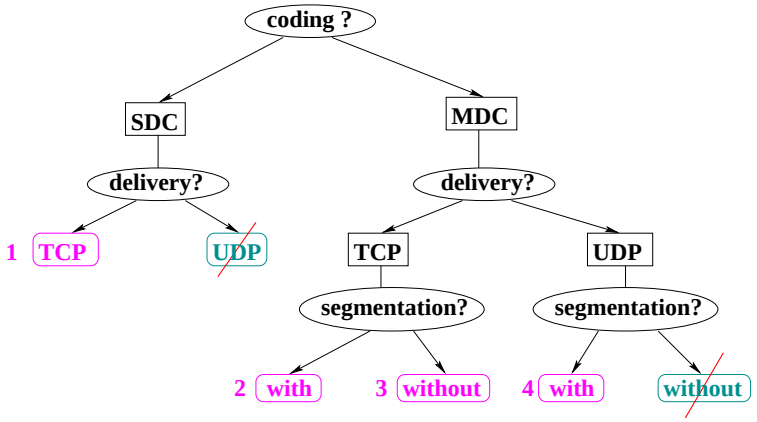


Figure 6.11: Alternative coding and delivery schemes.

6.4 Quality-Delay Trade-offs between TCP and UDP Delivery of Images

In this section, we examine in detail the quality-delay trade-offs of several alternative coding and delivery schemes for image data.

6.4.1 Quality-Delay Trade-offs of Alternative Coding and Delivery Schemes

Figure 6.11 organizes possible image coding and delivery schemes in hierarchies, depending on if an image is coded in SDC or MDC, delivered by TCP or UDP, and segmented or not. As discussed in Section 6.3.2, segmentation is introduced to isolate the effects of packet losses by dividing images into independent blocks, so that the loss of a packet

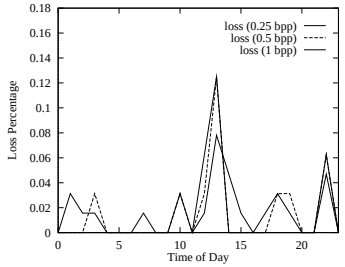
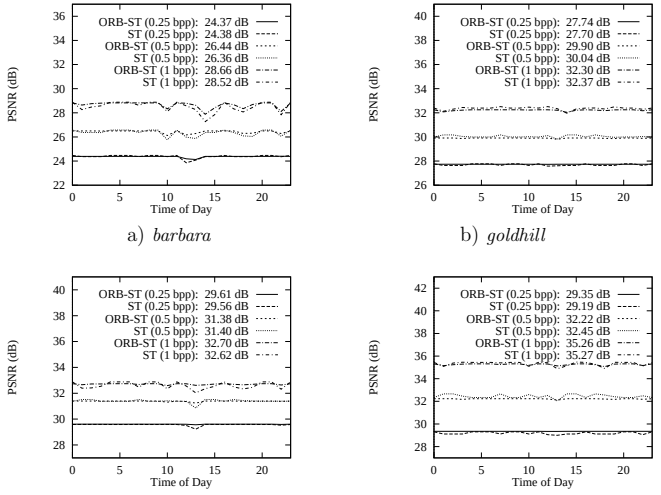


Figure 6.9: Loss rates of 16-, 32- and 64-packet transmissions between Urbana and California.



a) *barbara* b) *goldhill*

c) *peppers* d) *lena*

Figure 6.10: Comparisons of reconstruction quality over a 24-hour period for the Urbana to California connection, when each image was coded at, respectively, 0.25 bpp, 0.5 bpp and 1 bpp, and placed into 16, 32 and 64 packets for transmission.

does not propagate to other packets. Next, we walk through different branches of the hierarchy to identify coding and delivery schemes to be studied.

For TCP delivery of SDC images, every packet is reliably transmitted, so segmentation is not included further down in this branch. For UDP delivery of SDC images, the quality of transmitted images is likely to be very poor because the SDC images are very vulnerable to UDP packet losses; hence, this scheme is not further considered. For TCP delivery of MDC images, since all packets containing MDC data are also reliably transmitted, it would be unnecessary to include segmentation down in this branch. However, in order to isolate the effects of segmentation to MDC coded data, we still consider the two approaches with and without segmentation. For UDP delivery of MDC images, we study the scheme with segmentation because it is necessary to isolate packet losses; otherwise, the decoded images would be of poor quality due to propagation effects.

In summary, we have chosen the following four modes of coding and delivery in our experiments: 1) TCP delivery of SDC image data, 2) TCP delivery of MDC data in which the image is not segmented, 3) TCP delivery of MDC data in which the image is segmented, and 4) UDP delivery of MDC segmented image data. To further illustrate the difference between ORB-ST and ST, we have included the quality evaluations of UDP delivery of both ORB-ST-coded and ST-coded segmented image data in mode 4.

Figures 6.12 to 6.23 show delay-quality trade-offs at midnight, 6am, 12 noon and 6pm, local time in the remote servers in China, UK and California, using the above four modes of coding and delivery. Results at other times are similar and are not shown.

The two curves and one point related to TCP delivery were obtained by assuming that each image was coded in 1 bpp and transmitted in 64 packets. Based on the statistics collected, we calculated the average arrival times of the first i, $i = 1, 2, \ldots, 64$, packets

and then evaluated the quality of the corresponding packets after decoding them by the SPIHT decoder. The times in each curve include both end-to-end delays and decoding times.

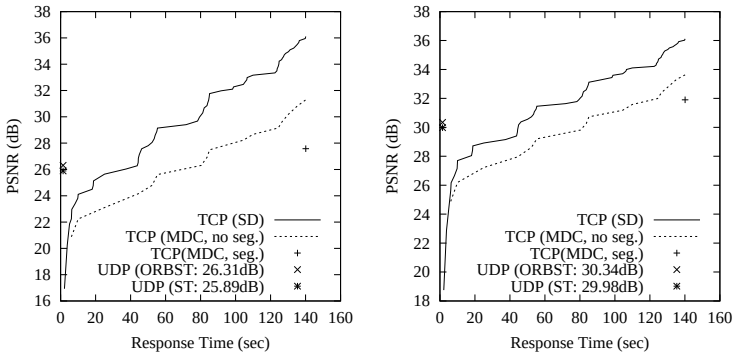
The two points related to UDP delivery were obtained under 1 bpp and included end-to-end delays, decoding time, and reconstruction time when losses happened. Since packets may arrive out of order in UDP delivery and the algorithm needs to wait for all packets to arrive before decoding, it is not possible to generate the sequence of quality-delay points as in TCP.

The graphs show that the UDP delivery of MDC images is an attractive alternative to the TCP delivery of SDC images when the delay that an end user can tolerate is small and when absolute quality is not critical. The graphs show that, without exception, TCP delivery leads to poorer quality using the same amount of time required by UDP delivery. Hence, UDP delivery is beneficial when one is interested to preview a coarser image, rather than waiting impatiently for the arrival of a perfect TCP transmitted image.

As an illustration, Figure 6.24 depicts *barbara* (resp. *goldhill*, *lena* and *peppers*) when it was delivered to UK (resp. China, China and California) by UDP and by TCP. Although they differ by 3-8 dB in PSNR, subjective quality differences are not significant.

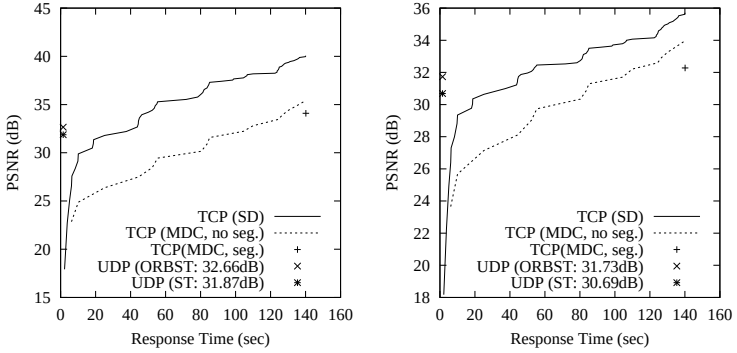
6.4.2 Analysis of Quality-Delay Trade-offs

As described in Chapter 3, we have reduced the probability of unrecoverable UDP packet losses to less than 5% by properly choosing interleaving factors according to network conditions. Yet we still have degradations in quality when compared to the TCP delivery of SDC data due to the reconstruction process and MDC. Figures 6.12 to 6.23 also



a) *barbara*

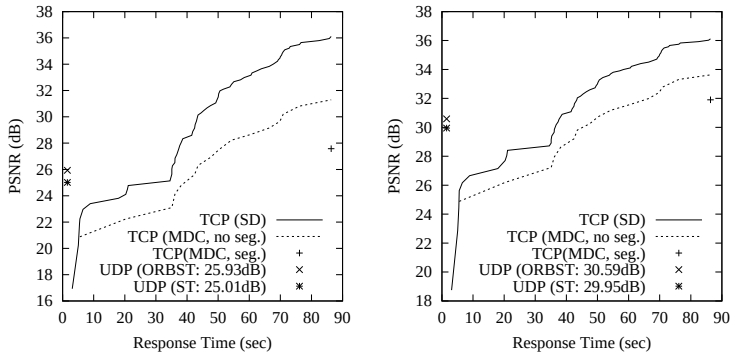
b) *goldhill*



c) *lena*

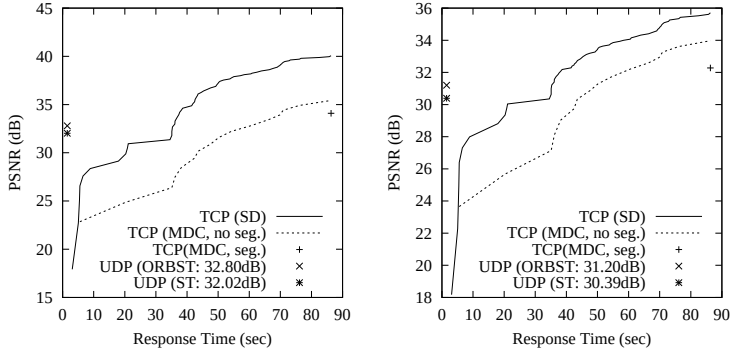
d) *peppers*

Figure 6.13: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-China connection at 6am Beijing Standard Time.



a) *barbara*

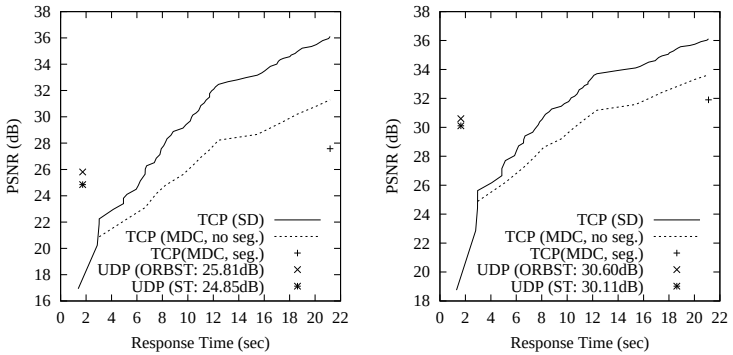
b) *goldhill*



c) *lena*

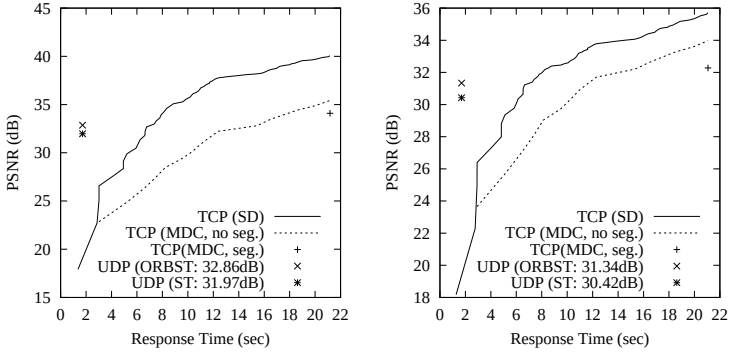
d) *peppers*

Figure 6.12: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-China connection at 0 midnight Beijing Standard Time.



a) *barbara*

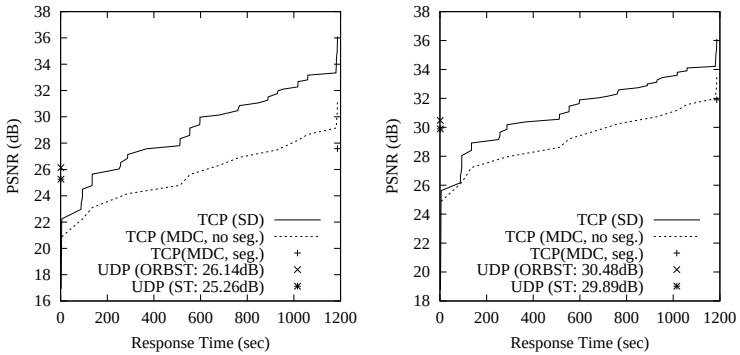
b) *goldhill*



c) *lena*

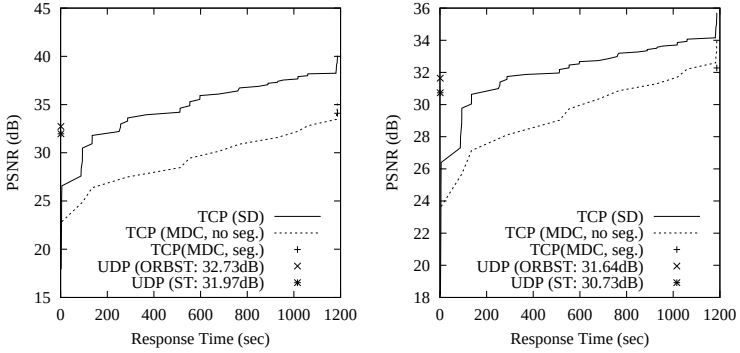
d) *peppers*

Figure 6.15: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-China connection at 6pm Beijing Standard Time.



a) *barbara*

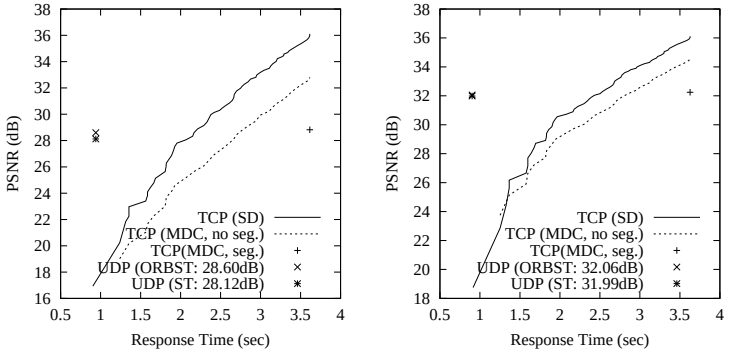
b) *goldhill*



c) *lena*

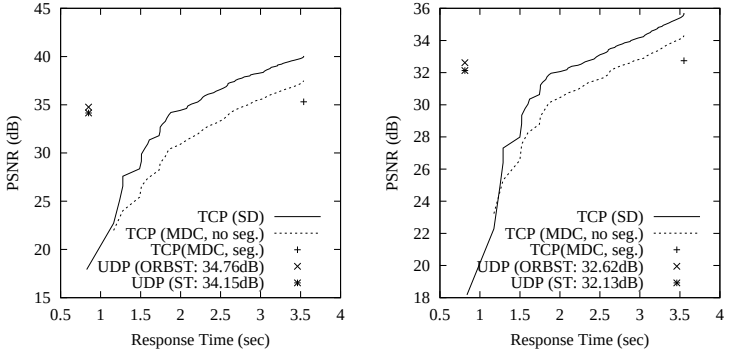
d) *peppers*

Figure 6.14: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-China connection at 12 noon Beijing Standard Time.



a) *barbara*

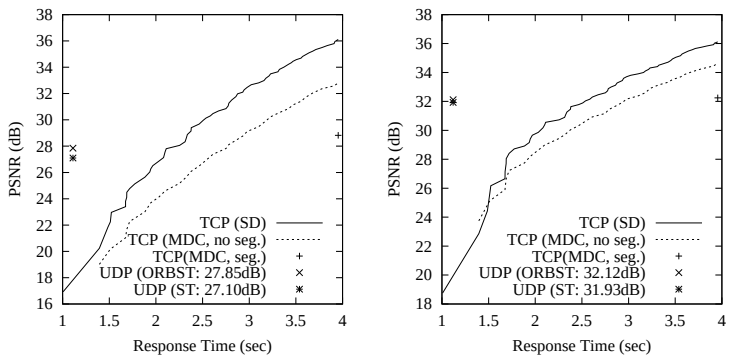
b) *goldhill*



c) *lena*

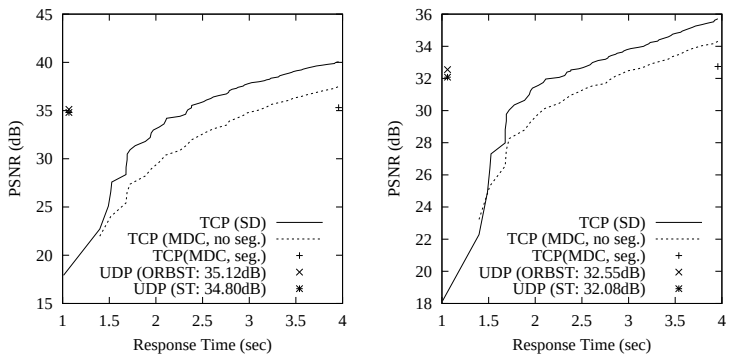
d) *peppers*

Figure 6.17: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-UK connection at 6am GMT.



a) *barbara*

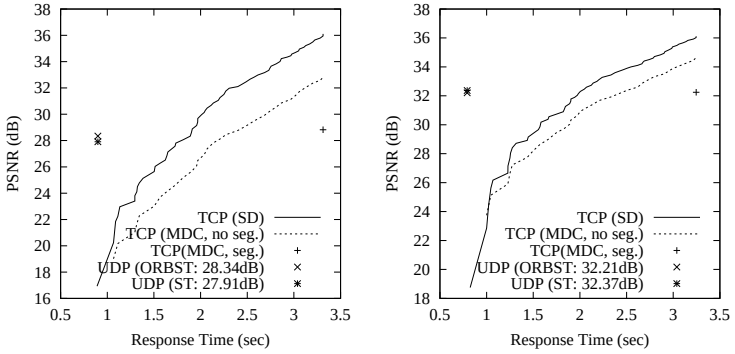
b) *goldhill*



c) *lena*

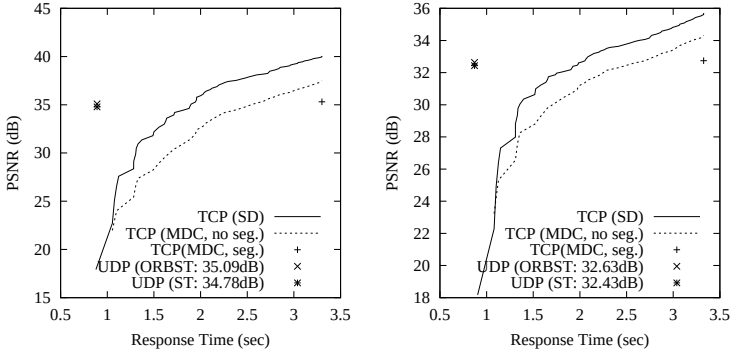
d) *peppers*

Figure 6.16: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-UK connection at 0 midnight GMT.



a) *barbara*

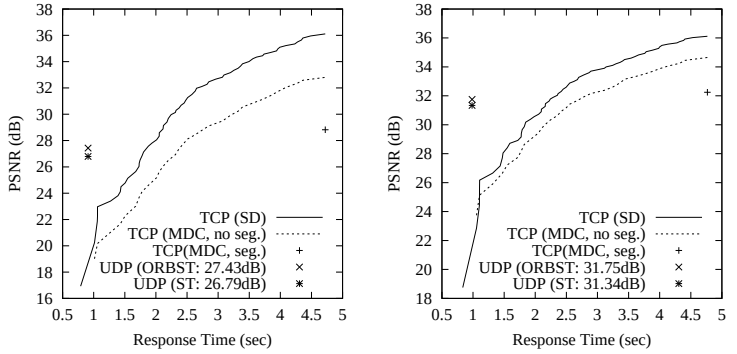
b) *goldhill*



c) *lena*

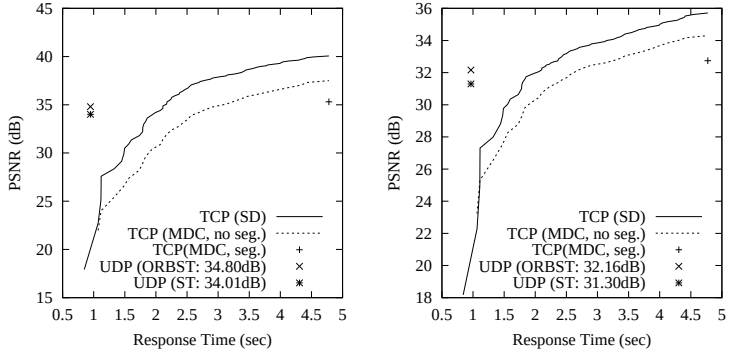
d) *peppers*

Figure 6.19: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-UK connection at 6pm GMT.



a) *barbara*

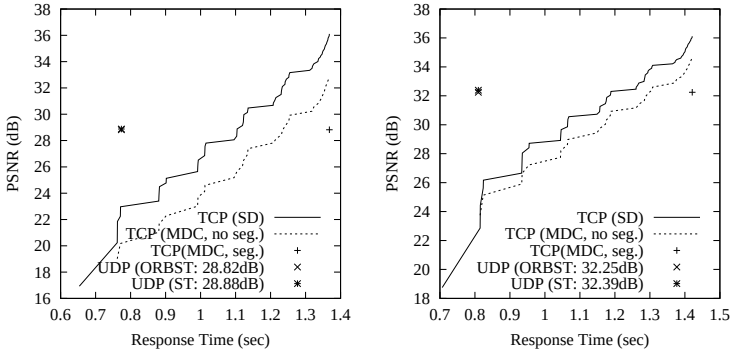
b) *goldhill*



c) *lena*

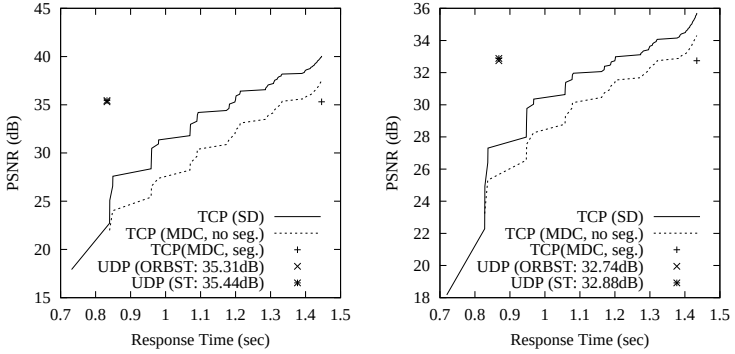
d) *peppers*

Figure 6.18: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-UK connection at 12 noon GMT.



a) *barbara*

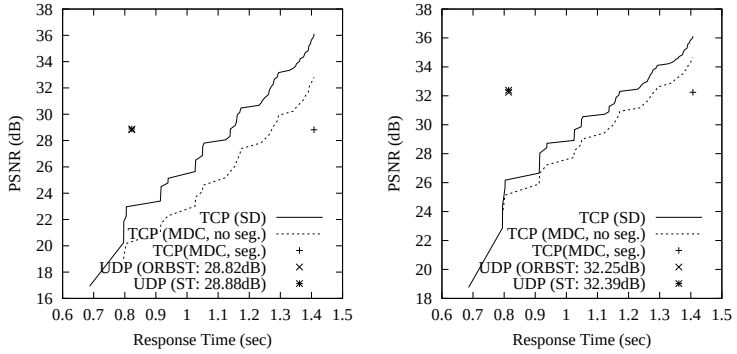
b) *goldhill*



c) *lena*

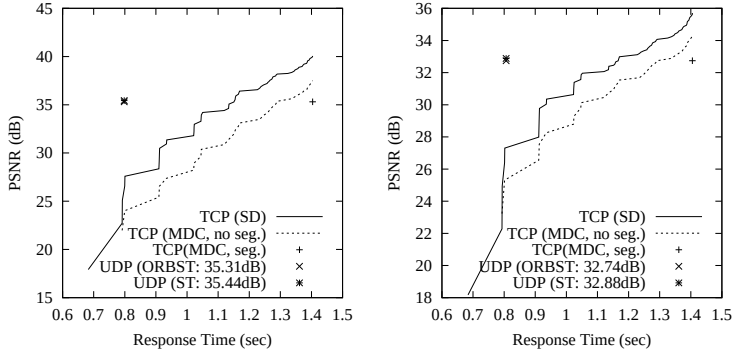
d) *peppers*

Figure 6.21: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-California connection at 6am PDT.



a) *barbara*

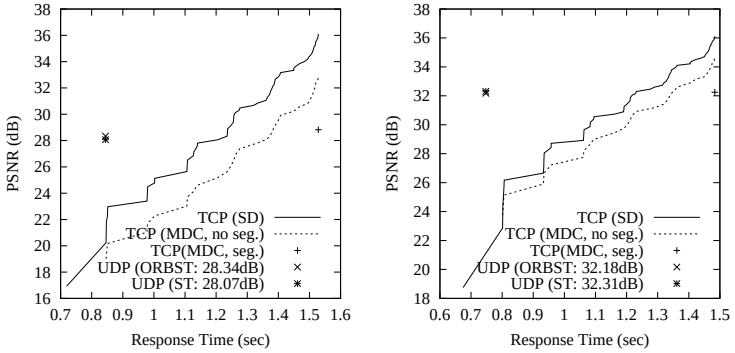
b) *goldhill*



c) *lena*

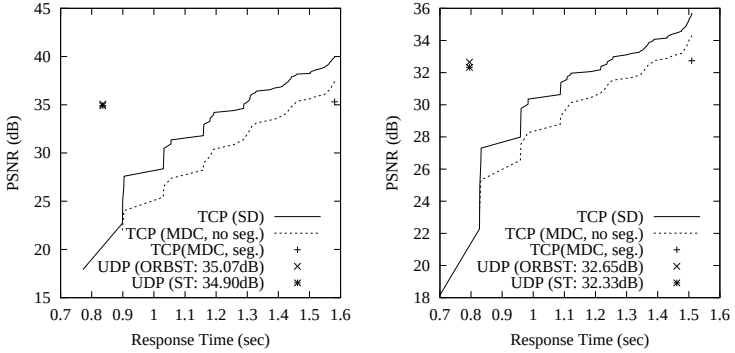
d) *peppers*

Figure 6.20: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-California connection at 0 midnight PDT.



a) *barbara*

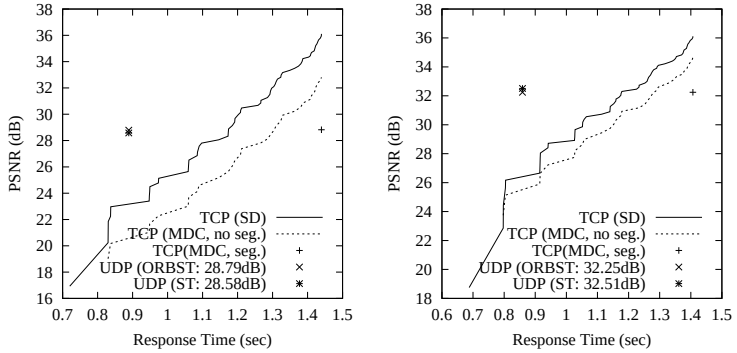
b) *goldhill*



c) *lena*

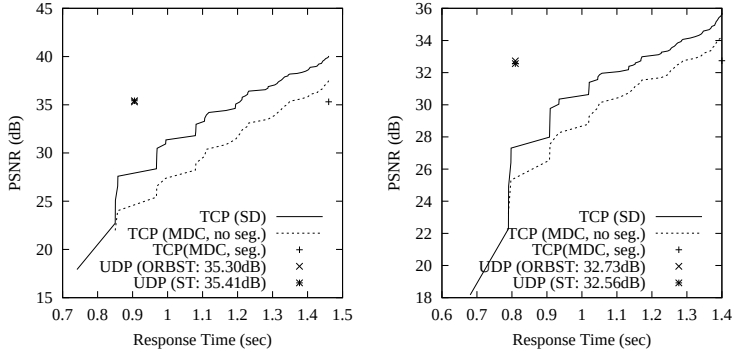
d) *peppers*

Figure 6.23: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-California connection at 6pm PDT.



a) *barbara*

b) *goldhill*



c) *lena*

d) *peppers*

Figure 6.22: Quality-delay trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-California connection at 12 noon PDT.

a) *barbara* transferred by UDP to UK using segmented MDC (27.85 dB)

b) *barbara* transferred by TCP using SDC (36.10 dB)

c) *goldhill* transferred by UDP to China using segmented MDC (30.59 dB)

d) *goldhill* transferred by TCP using SDC (36.11 dB)

Figure 6.24: Images *barbara*, *goldhill*, *lena* and *peppers* when transferred, respectively, by UDP using segmented MDC and by TCP using SDC.

e) *lena* transferred by UDP to China using segmented MDC (32.80 dB)

f) *lena* transferred by TCP using SDC (40.06 dB)

g) *peppers* transferred by UDP to California using segmented MDC (32.88 dB)

h) *peppers* transferred by TCP using SDC (35.71 dB)

Figure 6.24: (Cont'd)

illustrate three factors that cause the degradations in quality by several dBs between the TCP delivery of SDC images and the UDP delivery of MDC images.

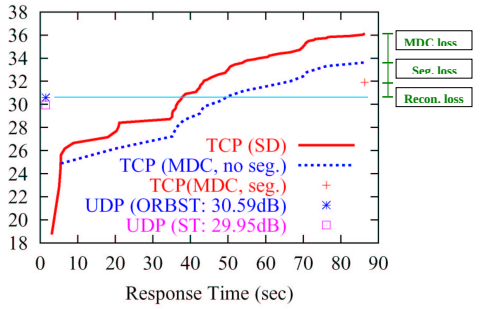


Figure 6.25: Analysis of quality-delay trade-offs when delivering *goldhill* to China (an annotated version of Figure 6.12b).

As an illustration, we indicate the three degradation factors in Figure 6.25, which is an annotated version of Figure 6.12b.

First, MDC alone causes between 1 to 3.5 dB loss in PSNR and is the price paid for improved error resilience. This is illustrated by the difference between the top two curves in the graph that show the quality of TCP delivery of SDC images and that of MDC images. Such degradations happen because of reduced correlations when partitioning an image into multiple descriptions and the suboptimal fixed coding rate for each description.

Second, another 2 to 3.5 dB loss in PSNR is caused by the suboptimal strategies of using fixed-size segments in the segmentation of image data in each description and of using a fixed coding rate for each segment in order for the coded segment to fit in a 512-byte packet (the difference between the point on the right of the blue dotted line and the

cross on the right of the graph). Note that in the TCP delivery of segmented MDC data, decoding cannot be done until all the packets in both descriptions have been received. The segmentation of an image before coding and packetization is necessary because image data in each description does not contain synchronization points and cannot be decoded when some packets are lost. We plan to study in the future better strategies for allocating coding rates to segments.

Third, packet losses and reconstructions in the UDP delivery of segmented ST-MDC data lead to further degradations. These degradations are between 1 to 2 dB for the Urbana-China connection (the difference between the 'x' point on the left of the graph and the cross on the right of the graph). Large degradations may also be caused by the loss of all the packets in an interleaved set. Degradations for the connection to UK are less than 1 dB, and those for the connection to California are negligible. Note that improvements due to ORB-ST when compared to ST in the UDP delivery of segmented MDC data are less than 1 dB (the two points on the left of the graph), as evaluated in the last subsection.

6.5 Hybrid TCP/UDP Delivery of SDC/MDC Images

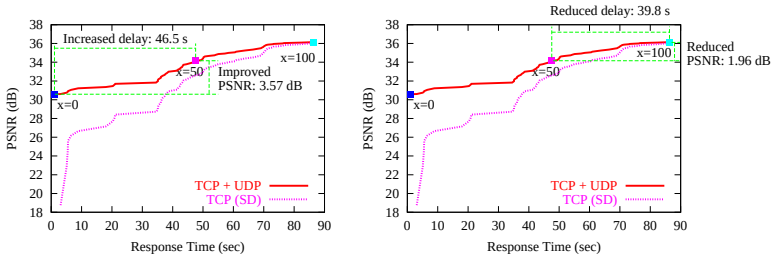
6.5.1 Motivations for Hybrid Coding and Delivery

The quality-delay trade-offs studied previously only show two extreme cases of image transmissions, either by TCP or by UDP. By inspecting the trade-off graphs, we see a promising hybrid approach by using combined TCP and UDP. For TCP delivery, quality improves very quickly in the beginning but saturates gradually when more packets are

6.5.2 Experimental Results on Quality-Delay-Bandwidth Trade-offs

To evaluate the quality-bandwidth-delay trade-offs involved, we have compared the following approaches for three sites and four test images:

- TCP delivery of SDC images.
- Hybrid TCP/UDP delivery of SDC and MDC images, with α set to $1, 1.2, 1.5, 1.8$ and 2, respectively.
- Redundant UDP transmission of MDC images by sending y copies of UDP packets that contain MDC data, with y set to 2 and 3.



a) comparison with pure UDP delivery b) comparison with pure TCP delivery

Figure 6.26: The quality-delay trade-offs when delivering *goldhill* to China with $x = 50$ in the hybrid TCP/UDP delivery.

Figures 6.27 to 6.30, 6.31 to 6.34, and 6.35 to 6.38 show the trade-offs between quality-delay and bandwidth-delay using the above approaches for transmissions to China, UK, and California at midnight, 6am, 12 noon, and 6pm its local time, respectively. First, we can see that the hybrid scheme is a general approach that generates a range of trade-off

available. Since the first few packets delivered by TCP incur insignificant delays, we can transmit them by TCP and deliver the MDC residuals by UDP. In general, we can characterize this approach as follows:

$$\{x : \text{SDC } by \text{ TCP}\} + \{\alpha(100 - x) : \text{MDC } by \text{ UDP}\} \qquad 0 \le x \le 100 \text{ and } \alpha \ge 1.$$

In this approach, the first $x\%$ of the bitstream is coded by SDC and delivered by TCP, and the rest of the bitstream is coded either redundantly ($\alpha > 1$) or non-redundantly ($\alpha = 1$) by MDC and delivered by UDP. Let us analyze why this combined approach would give better trade-offs. In the previous section, we have identified three factors that cause quality degradations of UDP delivery of MDC coded images, namely, MDC coding, fixed segmentation of images, and packet losses and reconstruction. The combined approach can, to some extent, reduce all the three kinds of losses, depending on the value of x chosen. Since the first $x\%$ of the bit stream is coded in SDC and transmitted by TCP, it suffers from neither of the above three kinds of losses. The larger x is, the less degradation in quality one has to encounter. In fact, if both x and α are equal to one, this approach is reduced to the pure TCP delivery of SDC images, leading to the highest quality but the longest delay. On the other hand, the larger x is, the longer delay users have to wait. Again, this approach involves a trade-off between delay and quality. In addition, there is a new trade-off between bandwidth and delay that is introduced by redundancy factor α. The larger α is, the more tranmission bandwidth is required, the better image quality is achieved, and the longer end-to-end response time is expected. Such trade-offs are acceptable when users have sufficient bandwidth available and the transmission of additional packets does not incur long delays.

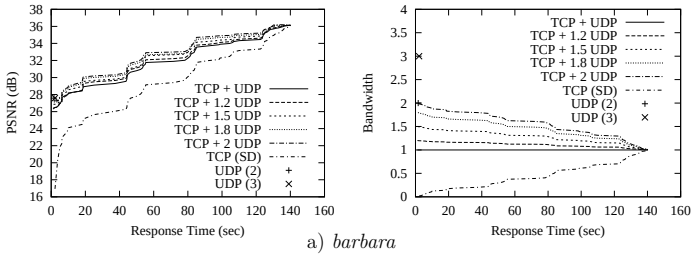
points including both TCP and UDP deliveries. To illustrate this concept, Figure 6.26 is annotated to show the trade-offs involved for delivering image *goldhill* to China. In the figure, the point labeled $x = 0$ corresponds to the trade-off of pure UDP delivery, and that labeled $x = 100$ shows the trade-off of pure TCP delivery. The points between $x = 0$ and $x = 100$ correspond to a range of hybrid alternatives to pure TCP and UDP deliveries. For example, the point with $x = 50$ shows the trade-off when delivering one half of a coded image by TCP and the other half by UDP. This hybrid approach ($x = 50$), when compared to pure UDP delivery, gives an improved PSNR of 3.57 dB and increased delay of 46.5 seconds, and when compared to pure TCP delivery, decreases transmission delay by 39.8 seconds and reduces PSNR by 1.96 dB. Second, we can see that the quality of the hybrid approach improves with increasing redudancies, characterized by α and plotted in the graphs on the right. Last, the redundant UDP delivery of MDC images does not appear to be attractive choices because of high redundancy, mediocre quality and long delays.

An interesting observation from the graph indicates that for transmissions to UK and California (plotted in Figures 6.31 to 6.38), the beginning part of some delay-quality lines of the hybrid approach is not monotonic as those in the experiments to China. To understand this phenomenon, we know that these beginning points correspond to delay-quality results of transmitting the majority of a bit stream by UDP and very little by TCP packets. Since the first few TCP packets normally incur less delay than UDP packets for this transmission (but not true for transmissions to China), the total delay is dominated by that of UDP packets. Therefore, the initial delay tends to reduce and the quality generally improves with decreasing UDP packets to transmit and more image data coded by SDC. This trend stops when it reaches the point when the TCP

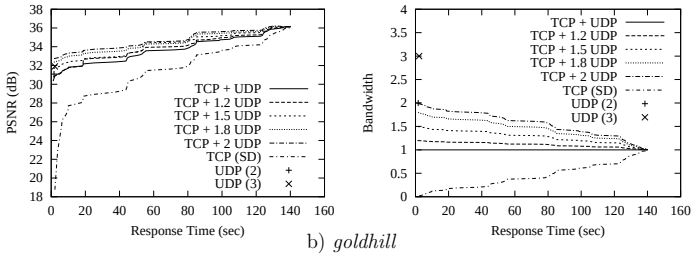
delay begins to dominate the total response time. From that point on, the delay-quality lines monotonically increase until they converge to the point corresponding to the TCP delivery of SDC data. In fact, these initial points up to the turning points are at least not as desirable as the turning points in terms of the quality-delay trade-offs, so they should be avoided when choosing coding and transmission schemes.

As an illustration, Figure 6.39 depicts images *barbara*, *goldhill*, *lena*, and *peppers* when they are transferred using the hybrid approach with $x = 50$ and $\alpha = 1$. We can see that they are both subjectively and objectively closer to the SDC images (shown in Figure 6.24) delivered using TCP.

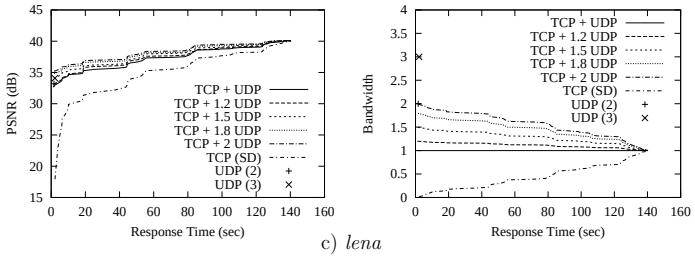
Based on the above experiments, we conclude that the hybrid approach provides a range of choices that give higher quality than pure UDP delivery and shorter delays than pure TCP delivery. Users can make suitable choices according to their QoS requirements and available resources.



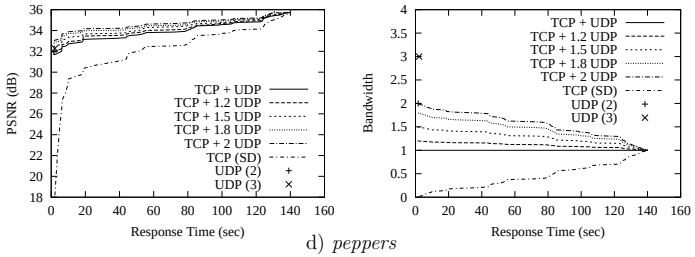
a) *barbara*



b) *goldhill*

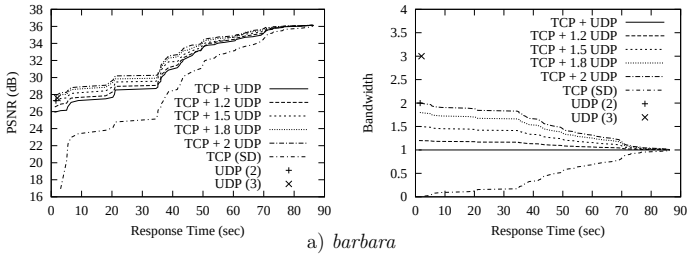


c) *lena*

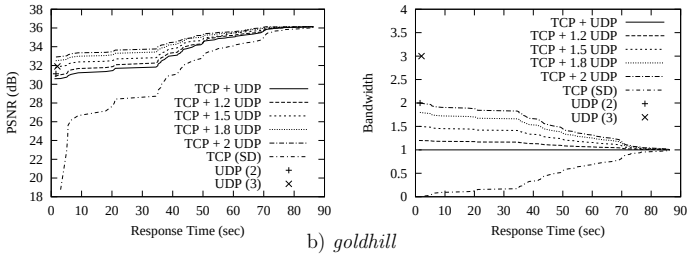


d) *peppers*

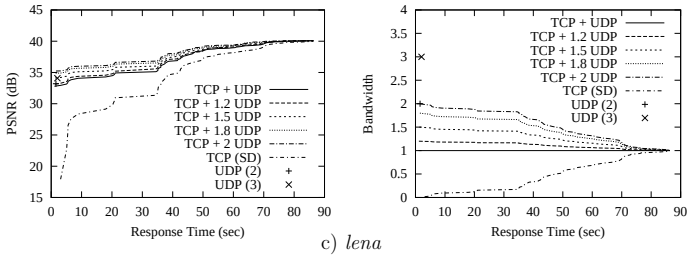
Figure 6.28: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data for the Urbana-China connection at 6am Beijing Standard Time.



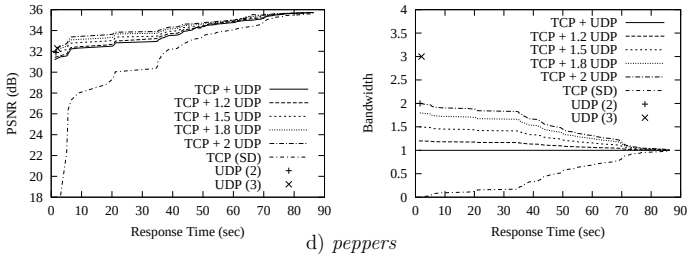
a) *barbara*



b) *goldhill*



c) *lena*



d) *peppers*

Figure 6.27: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data for the Urbana-China connection at 0 midnight Beijing Standard Time.

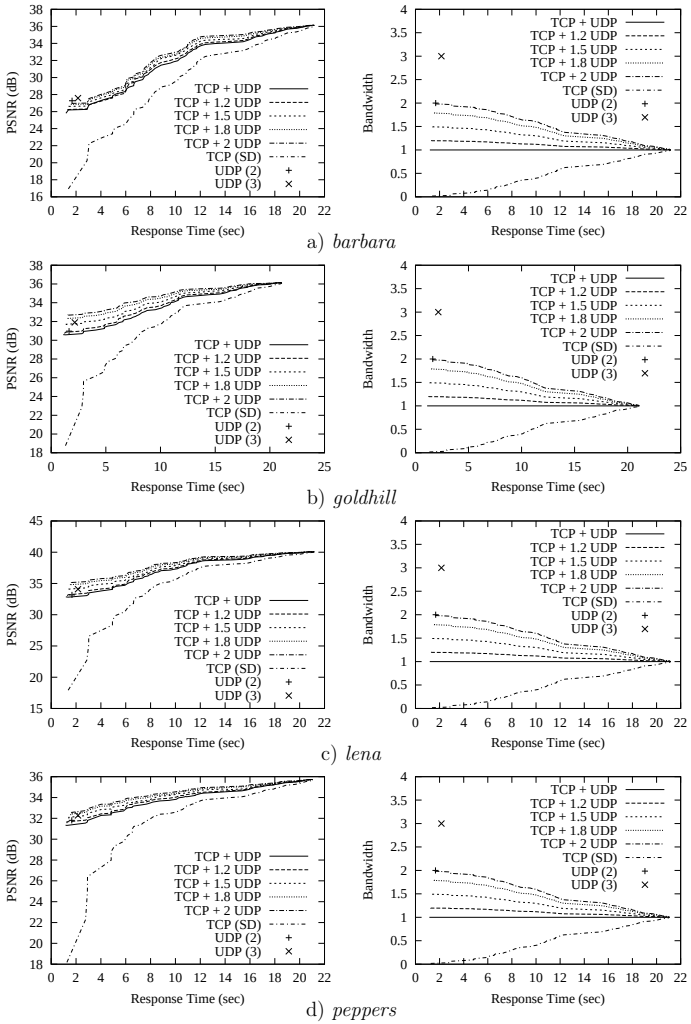


Figure 6.30: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data for the Urbana-China connection at 6pm Beijing Standard Time.

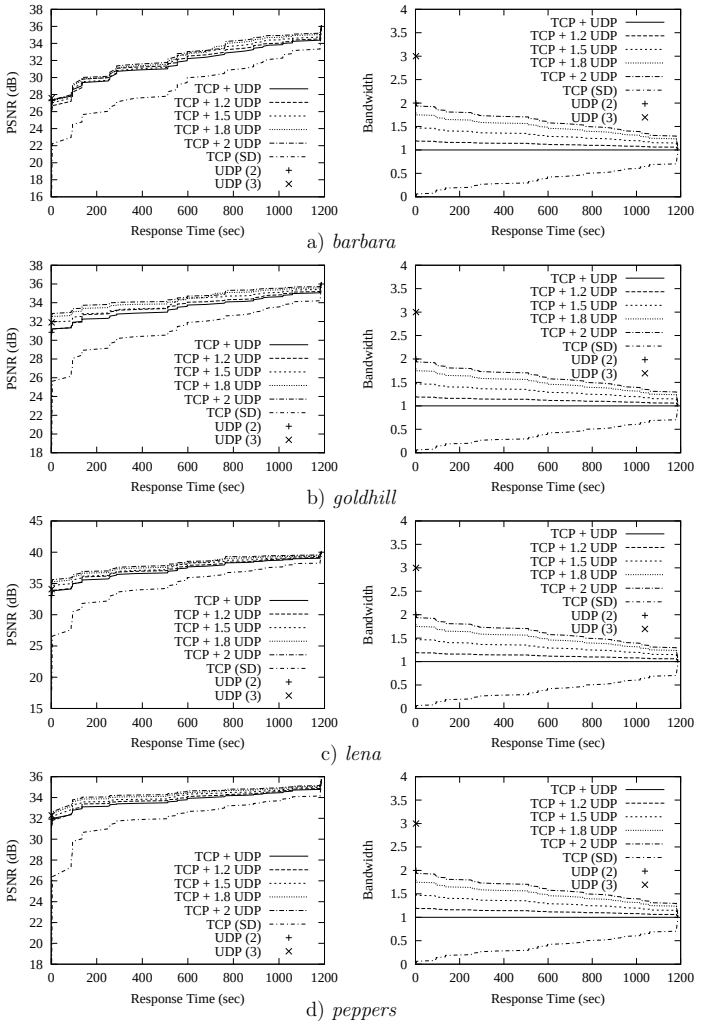


Figure 6.29: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data for the Urbana-China connection at 12 noon Beijing Standard Time.

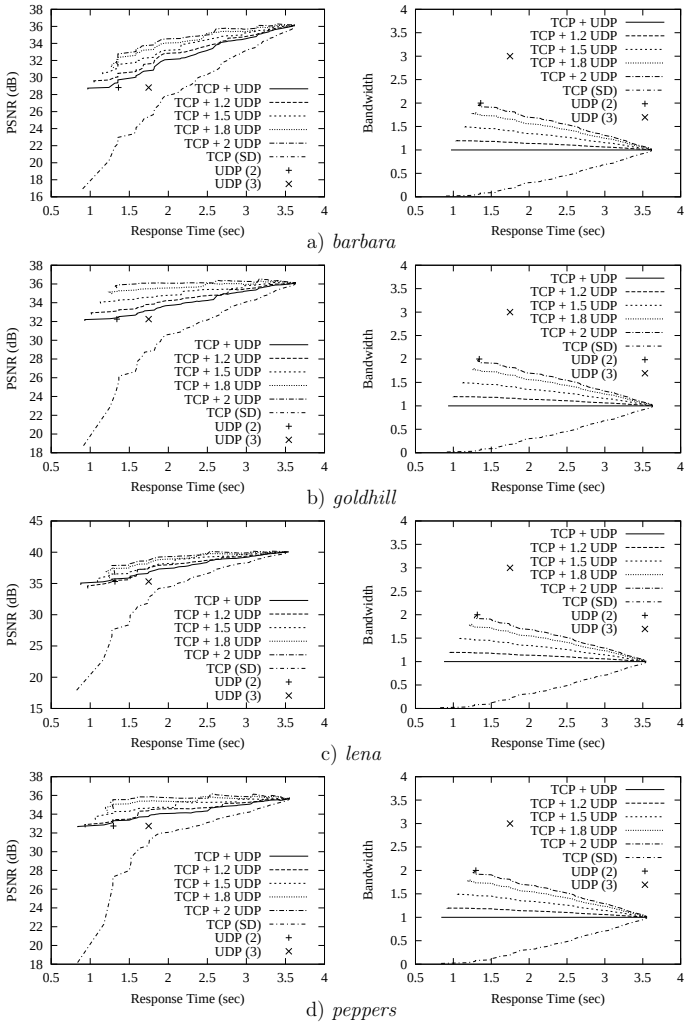


Figure 6.32: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data data for the Urbana-UK connection at 6am GMT.

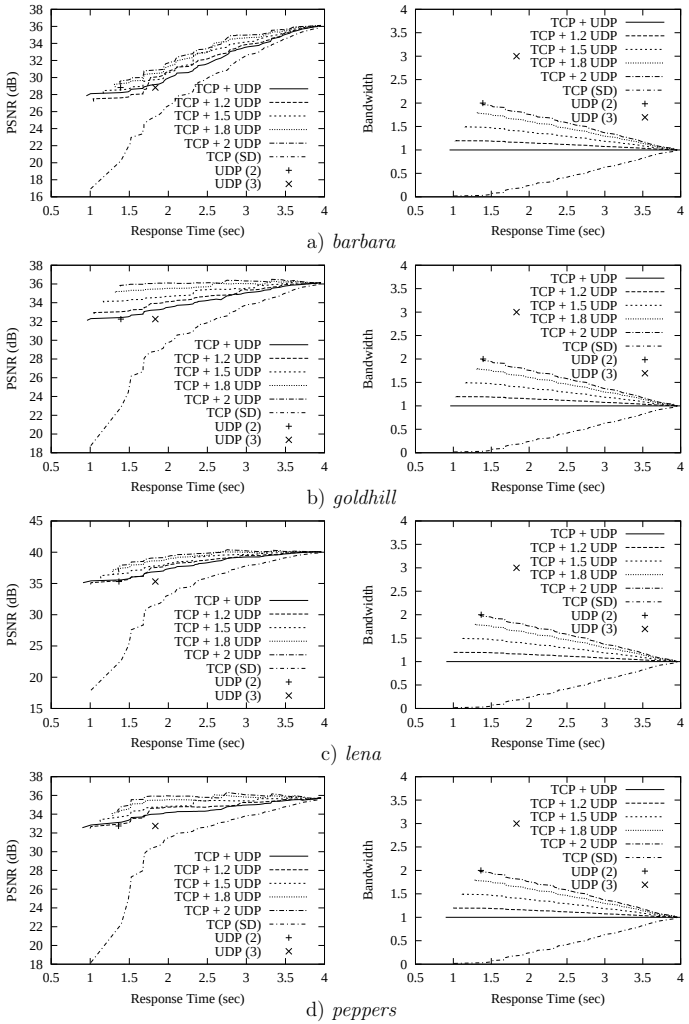
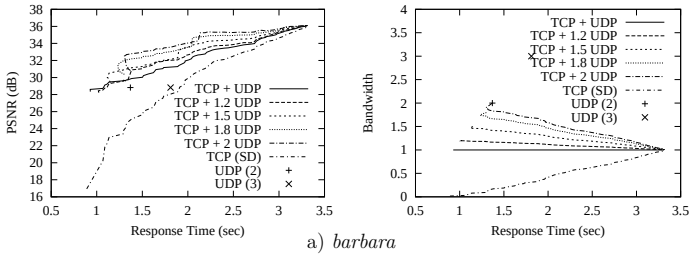
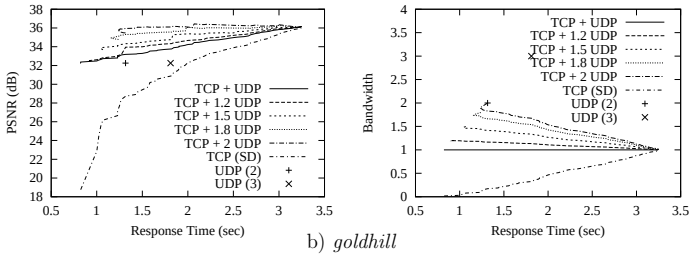


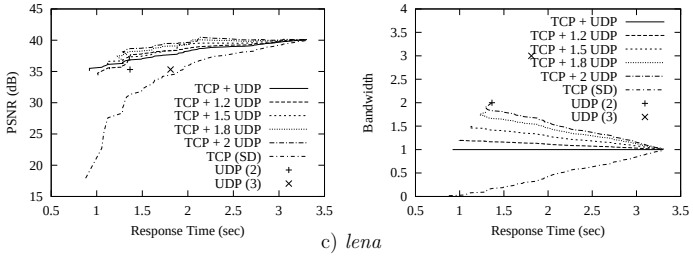
Figure 6.31: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data for the Urbana-UK connection at 0 midnight GMT.



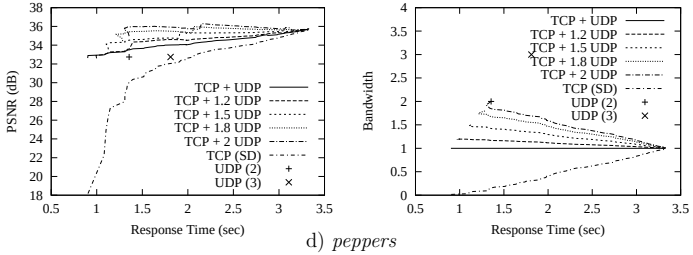
a) *barbara*



b) *goldhill*

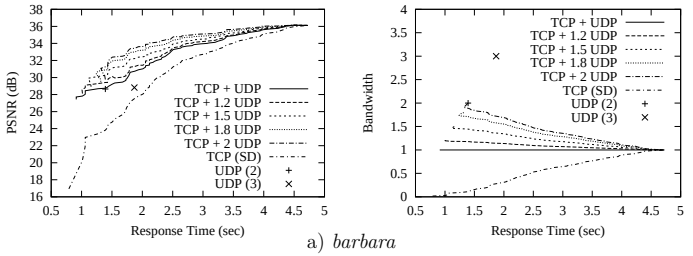


c) *lena*

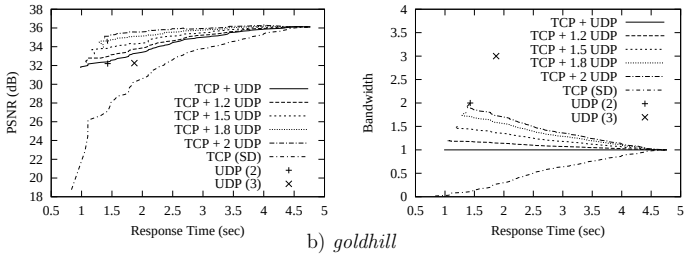


d) *peppers*

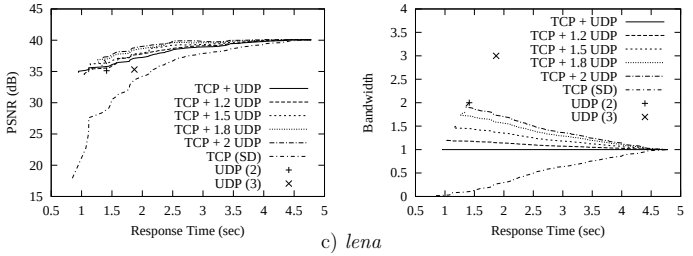
Figure 6.34: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data for the Urbana-UK connection at 6pm GMT.



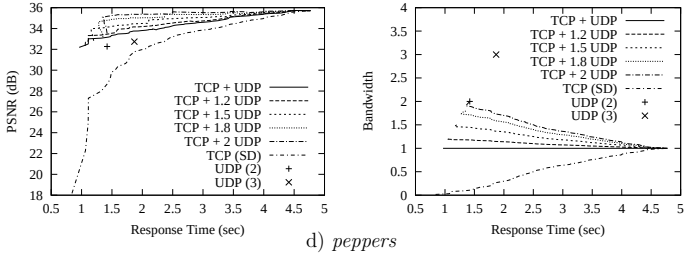
a) *barbara*



b) *goldhill*



c) *lena*



d) *peppers*

Figure 6.33: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data for the Urbana-UK connection at 12 noon GMT.

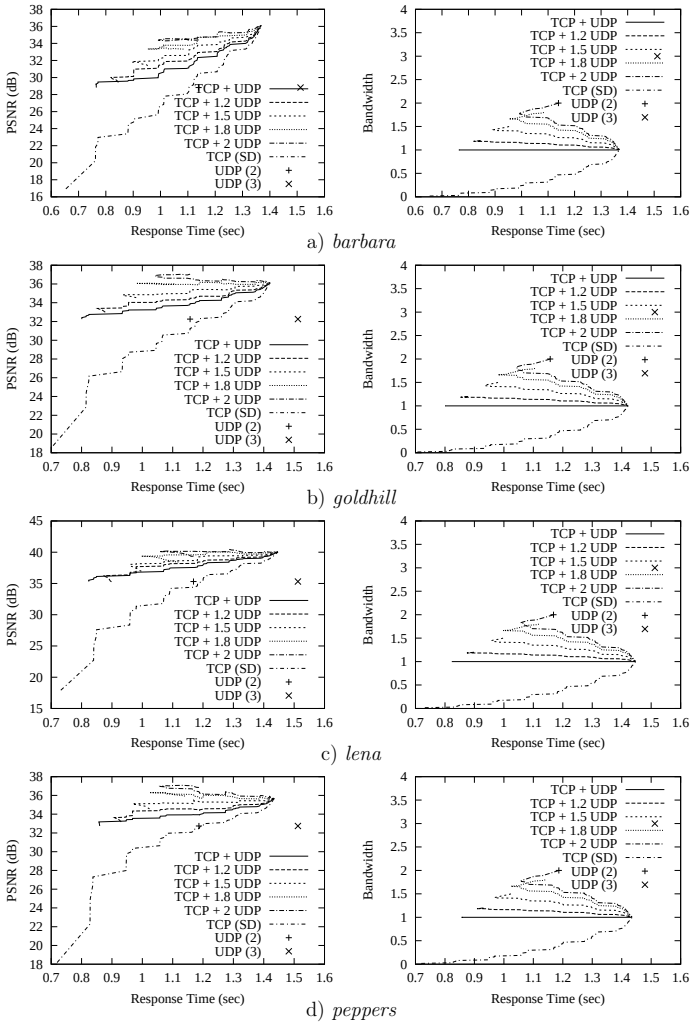


Figure 6.36: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data for the Urbana-California connection at 6am PDT.

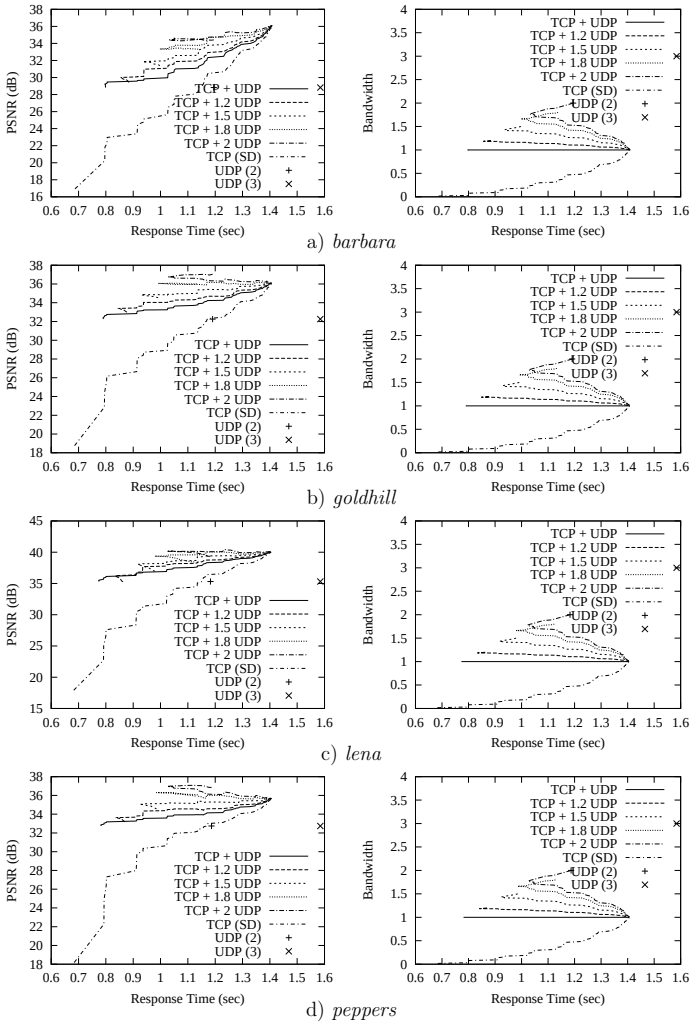


Figure 6.35: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data for the Urbana-California connection at 0 midnight PDT.

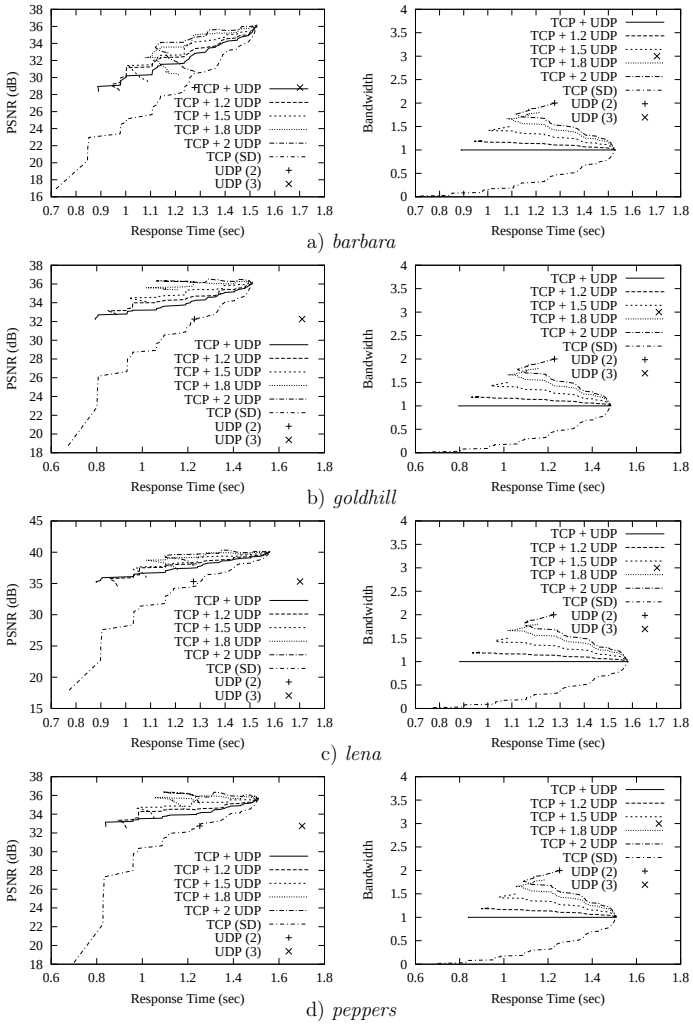


Figure 6.38: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data for the Urbana-California connection at 6pm PDT.

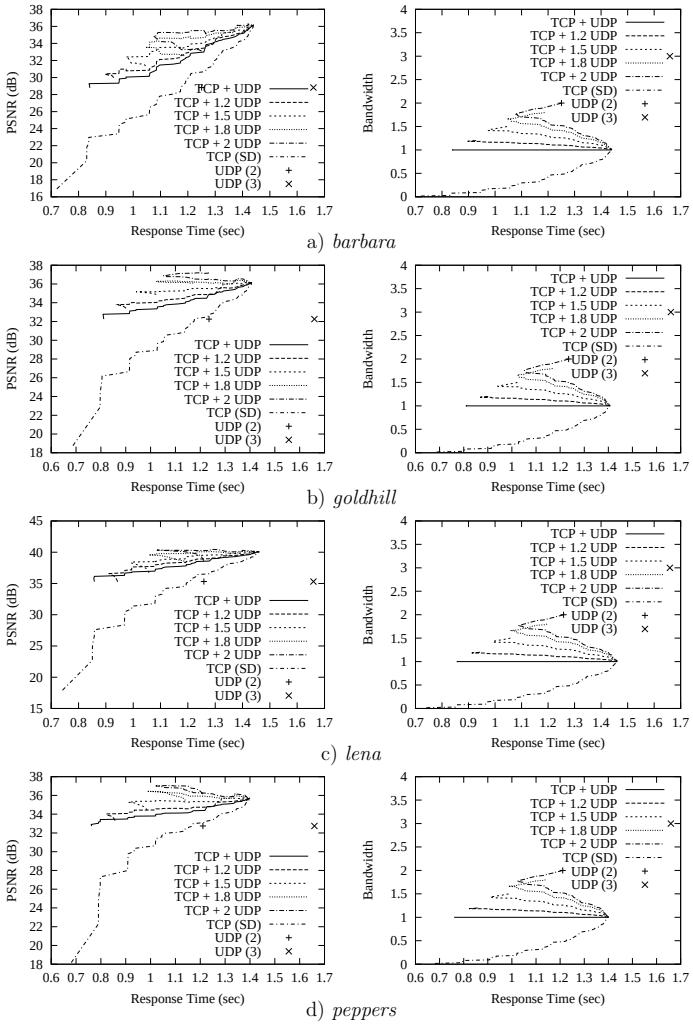


Figure 6.37: Quality-delay and bandwidth-delay trade-offs of hybrid TCP/UDP delivery of image data for the Urbana-California connection at 12 noon PDT.

a) *barbara* transferred to UK (32.67 dB)

b) *goldhill* transferred to China (34.16 dB)

c) *lena* transferred to China (38.34 dB)

d) *peppers* transferred to California (34.38 dB)

Figure 6.39: Images *barbara*, *goldhill*, *lena* and *peppers* when transferred using the hybrid TCP/UDP approach with $x = 50$ and $\alpha = 1$.

6.6 Summary

In this chapter, we have studied coding and transmission schemes of subband-coded images for delivery on the Internet. Our traffic experiments (described in Chapter 3) reveal that TCP delivery of coded images is one to two orders slower than UDP delivery when networks are congested, which implies that TCP delivery is not a good choice in applications where response time is more important than absolute quality. To address the need of delivering coded images in shorter response time, faster UDP is more preferrable, but packet losses in UDP may lead to poor decoding quality if the image is single-description coded (SDC) and the losses cannot be concealed. To reduce the effects of packet losses, we propose to use multi-description coding (MDC) by designing optimized reconstruction-based subband transform (ORB-ST).

Next, we have carefully evaluated delay-quality trade-offs of two coding and delivery approaches: TCP delivery of SDC images and UDP delivery of MDC images, and have found that UDP delivery is an attractive alternative to conventional TCP delivery when the end-to-end response time that a user can tolerate is small. We have also identified three factors that lead to the quality degradations in UDP delivery when compared to TCP delivery given that transmission times are sufficiently long.

Finally, we have proposed a hybrid TCP/UDP delivery to improve the quality of pure UDP delivery and to reduce the delays of pure TCP delivery. In this hybrid approach, an image is first coded by SDC using part of the bandwidth. This part is delivered by TCP since it is very susceptible to packet losses. Next, the residual image is coded by our proposed MDC using the remaining bandwith. This part is delivered by UDP since it is resilient to packet losses. This hybrid approach is, in fact, a general scheme that

can provide users a range of attractive coding and transmission alternatives to pure TCP and UDP deliveries.

Chapter 7

Conclusions and Future Work

7.1 Summary of Accomplished Research

With the advent of the World Wide Web (WWW), coding and transmissions of real-time image and video data on the Internet have attracted more and more attention. In this thesis, we have designed and evaluated end-to-end robust coding, error concealment and delivery schemes for image and video transmissions on the Internet. The major contributions of this thesis are summarized as follows.

- First, we have proposed a practical approach by using remote echo ports in order to compare Internet traffic to various sites around the world. We have also modified the Linux kernel in order to allow fair comparisons of TCP and UDP transmissions. Using the modified kernel, we have conducted traffic experiments, simulating both image and video transmissions. Our experiments have led to two conclusions: a) We need to use UDP, instead of TCP, as the transport service for the timely delivery of images and videos. 2) UDP transmissions frequently encounter packet losses in small bursts. These measurements not only have provided guidelines but have also prepared traffic traces for the design and evaluation of error concealment schemes.

- Second, we have addressed two issues involoved in coding and transmission of H.263 videos, namely, information loss and bandwidth limitations. For the three kinds of information loss that we have identified, we have proposed ORB-DCT to cope with bitstream loss caused by dropped packets, ANN-based reconstruction to compensate for compression loss due to multiple description coding, and packetization and reconstruction-feedback to reduce propogation loss. To address bandwidth limitations in transmission, we have studied rate control approaches in both the GOB and the block levels. Our study of rate control problems differs from previous approaches in that our approach is reconstruction-based and includes the error of the reconstruction process in the final quality measure, while previous schemes do not. Tests on the Internet have verified that our proposed schemes work well in real-world transmissions.

- Last, we have carefully studied the quality-delay trade-offs involved in transmitting subband-coded images in the Internet. Delivery by TCP gives superior decoding quality but with very long turnaround time when the network is unreliable, whereas delivery by UDP has negligible delays but with degraded quality when packets are lost. Although images are delivered primarily by TCP today, we have found that the use of UDP to deliver multi-description reconstruction-based subband-coded images is more attractive than the TCP delivery of single-description coded images. After carefully analyzing these two delivery schemes, we have further proposed a hybrid TCP/UDP approach that delivers part of an image coded in SDC by TCP and the remaining part coded in reconstruction-based MDC by UDP. Experimental results show that the hybrid approach can lead to various delay-quality trade-offs between the two extremes of pure UDP and TCP delivery.

7.2 Future Work

In the future, research can be carried out along a number of possible directions.

- *Designing schemes to improve subjective quality.* In this thesis, we have used the objective measure of PSNR to assess the quality of images and videos. As is well known, PSNR values are not well correlated to subjective results. In the future, we plan to study schemes to improve subjective image and video qualities. To this end, we need to first identify a quantitative quality measure that better reflects subjective perceptions than simple PSNR. Undoubtedly, we must exploit the properties of the human visual system, such as luminance and frequency masking effects, to derive this criteria. At the same time, we need to keep it simple in order to facilitate real-time applications. Next, we plan to use the new measure as the objective for optimization and follow similar approaches as discussed in this thesis to develop the corresponding error concealment schemes.

- *Extensions to multicast appliations.* With the explosive demand for multimedia transmissions on the Internet, an increasing number of applications will turn to multicasts to efficiently reduce bandwidth consumption. Unlike unicast transmissions, multicast delivery needs to accomodate heterogenous receivers, varying networking capacities, and a variety of loss characteristics. For efficient and reliable multicast delivery of multimedia in lossy networks, it is necessary to extend our MDC-based approach to support a range of loss situations: for low loss connections, MDC needs to be designed with an emphasis on quality when all descriptions are received; conversely for high loss connections, MDC needs to be designed with an emphasis on quality when only part of the desciptions are received. In addition,

our proposed MDC scheme may need to be combined with layered coding, in order to support transmissions in heterogeneous networks with different capacities.

- *Modifying operating system kernels to support hybrid TCP/UDP delivery of images.* To improve delay-quality trade-offs in image transmissions, we propose to use a hybrid delivery scheme using both TCP and UDP; *i.e.,* part of the bit stream is coded in SDC and delivered by TCP, and the rest is coded in MDC and delivered by UDP. At some point in time, the receiver decodes both TCP and UDP packets that have been received and displays the decoded image. To maximize the quality of decoded images, the decoder needs to make use of all the packets that have been received from the network. This is the case with UDP packets but not with TCP ones, since TCP never delivers out-of-order packets to applications, resulting in packets held for an indefinite amount of time in TCP buffers. Therefore, for the hybrid delivery approach to work better, we need to modify the TCP protocol stack in both the sender and receiver sides. At the receiver, its receiving window must be advanced after a given time threshold, and whatever data in the TCP buffers will be delivered to applications together with their sequence numbers. At the same time, the TCP window in the sender side needs to be advanced accordingly. In addition, in order for applications to make the maximum use of the fragmented data, the sender must also packetize data based on "hints" from applications. With the above modifications in a kernel, the hybrid TCP/UDP delivery approach can be extended easily to the transmission of image sequences.

Bibliography

[1] ITU-T recommendation H.323: Packet-based multimedia communications systems.

[2] RFC 1889: RTP − A transport protocol for real-time applications.

[3] RFC 1890: RTP profile for audio and video conferences with minimal control.

[4] RFC 2205: Resource ReSerVation protocol −− version 1 functional specification, September 1997.

[5] RFC 2326: Real time streaming protocol (RTSP), April 1998.

[6] S. Aign and K. Fazel. Temporal and spatial error concealment techniques for hierarchical MPEG-2 video codec. In *Proc. Globecom'95*, pages 1778-1783, 1995.

[7] A. Albanese, J. Bloemer, and J. Edmonds. Priority encoding transmission. In *Proc. Foundations of Computer Sciences*, pages 604-612, Santa Fe, NM, 1994.

[8] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies. Image coding using wavelet transform. *IEEE Trans. on Image Processing*, 1:205-220, April 1992.

[9] R. Aravind, M. R. Civanlar, and A. R. Reibman. Packet loss resilience of MPEG-2 scalable video coding algorithms. *IEEE Trans. on Circuits and Systems for Video Technology*, 6(5):426-435, October 1996.

[10] J. C. Batllo and V. A. Vaishampayan. Asymptotic performance of multiple description transform codes. *IEEE Trans. on Information Theory*, 43:703-707, March 1997.

[11] M. Brown, P. C. An, C. J. Harris, and H. Wang. How biased is your multi-layer perceptron? In *World Congress on Neural Networks*, pages 507-511, 1993.

[12] S. Cen, C. Pu, R. Staehli, C. Cowan, and J. Walpole. A distributed real-time MPEG video audio player. In *Proc. 5th Int'l Workshop Network and Operating System Support for Digital Audio and Video*, pages 151-162, April 1995.

[13] Z. G. Chen. *Coding and transmission of digital video on the Internet*. PhD thesis, University of Illinois at Urbana-Champaign, 1997.

[14] Z. G. Chen, S. M. Tan, R. H. Campbell, and Y. C. Li. Real time video and audio in the world wide web. In *Proc. 4th World Wide Web Conf.*, 1995.

[15] G. Cote, B. Erol, M. Gallant, and F. Kossentini. H.263+: video coding at low bit rates. *IEEE Trans. on Circuits and Systems for Video Technology*, 8(7), November 1998.

[16] J. M. Danskin, G. M. Davis, and X. Song. Fast lossy Internet image transmission. In *ACM Multimedia Conf.*, San Francisco, November 1995.

[17] P. J. Davis. *Circulant Matrices*. John Wiley Sons, NY, 1979.

[18] W. Ding and B. Liu. Rate control of MPEG video coding and recording by rate-quantization modeling. *IEEE Trans. on Circuits and Systems for Video Technology*, 6:12-20, February 1996.

[19] P. Eisert, T. Wiegand, and B. Girod. Model-aided coding: a new approach to incorporate facial information into motion-compensated video coding. *IEEE Trans. on Circuits and Systems for Video Technology*, 10:344-358, April 2000.

[20] T. J. Ferguson and J. H. Ranowitz. Self-synchronizing huffman codes. *IEEE Trans. on Information Theory*, IT-30:687-693, July 1984.

[21] M. Ghanbari. Two-layer coding of video signals for VBR networks. *IEEE journal on Selected Areas in Communications*, 7:801-806, June 1989.

[22] M. Ghanbari. Postprocessing of late cells for packet video. *IEEE Trans. on Circuits and Systems for Video Technology*, 6:669-678, December 1996.

[23] M. Ghanbari and V. Seferidis. Cell-loss concealment in ATM video codecs. *IEEE Trans. on Circuits and Systems for Video Technology*, 3:238-247, June 1993.

[24] R. M. Gray, K. Perlmutter, and R. A. Olshen. Quantization, classification, and density estimation for Kohonen's gaussian mixture. In *Proc. of Data Compression Conference*, pages 63-72, 1998.

[25] S. Haykin. *Neural Networks: A Comprehensive Foundation*. Prentice-Hall, NJ, 2nd edition, 1999.

[26] S. S. Hemami. Robust image transmission using resynchronizing variable length codes and error concealment. *IEEE journal on Selected Areas in Communications*, June 2000.

[27] S. S. Hemami and R. M. Gray. Subband coded image reconstruction for lossy packet networks. *IEEE Trans. on Image Processing*, April 1997.

[28] S. S. Hemami and T. H.-Y. Meng. Transform coded image reconstruction exploiting interblock correlation. *IEEE Trans. on Image Processing*, 4(7):1023-1027, July 1995.

[29] C.-Y. Hsu, A. Ortega, and M. Khansari. Rate control for robust video transmission over burst-error wireless channels. *IEEE Trans. on Circuits and Systems for Video Technology*, 17:756-773, May 1999.

[30] D. A. Huffman. A method for the construction of minimum redundancy codes. In *Proc. of IRE*, volume 40, pages 1098-1101, 1962.

[31] ITU. ITU-T recommendation P.910 - subjective video quality assessment methods for multimedia applications, 1999.

[32] V. Jacobson. Congestion avoidance and control. In *Proc. SIGCOMM*, pages 314-319, August 1998.

[33] N. Jayant, J. Johnston, and R. Safranek. Signal compression based on models of human perception. *Proc. of The IEEE*, 81:1385-1422, October 1993.

[34] N. Jayant and P. Noll. *Digital Coding of Waveforms: Principles and Applications to Speech and Video*. Englewood Cliffs, Prentice-Hall, 1984.

[35] T. Kawahara and S. Adachi. Video transmission technology with effective error protection and tough synchronization for wireless channels. In *Proc. IEEE Int'l Conf. on Image Processing*, pages 101-104, November 1996.

[36] L. H. Kieu and K. N. Ngan. Cell-loss concealment techniques for layered video codecs in an ATM network. *IEEE Trans. on Image Processing*, 3:666-677, September 1994.

[37] W. Kwok and H. Sun. Multi-directional interpolation for spatial error concealment. *IEEE Trans. on Consumer Electronics*, 39(3):455-460, August 1993.

[38] W. M. Lam and A. R. Reibman. Self-synchronizing variable length codes for image transmission. In *Proc. ICASSP'92*, pages 477-480, March 1992.

[39] M. S. Lazar and L. T. Bruton. Fractal block coding of digital video. *IEEE Trans. on Circuits and Systems for Video Technology*, 4(3):297-308, June 1994.

[40] Y. LeCun, I. Kanter, and S. A. Solla. Eigenvalues of covariance matrices: application to neural network learning. *Physical Review Letters*, 66(18):2396-2399, 1991.

[41] S. M. Lei and M. T. Sun. An entropy coding system for digital HDTV applications. *IEEE Trans. on Circuits and Systems for Video Technology*, 1:147-154, March 1991.

[42] D. A. Lelewer and D. S. Hirshberg. Data compression. *ACM Computing Surveys*, 19(3):261-295, 1987.

[43] L.-J. Lin and A. Ortega. Bit-rate control using piecewise approximated rate-distortion characteristics. *IEEE Trans. on Circuits and Systems for Video Technology*, 8:446-459, August 1998.

[44] S. M. LoPresto, K. Ramchandran, and M. T. Orchard. Image coding based on mixture modeling of wavelet coefficients and a fast estimation-quantization framework. In *Proc. of Data Compression Conference*, pages 221-230, March 1997.

[45] J. W. Modestino and D. G. Daut. Combined source-channel coding of images. *IEEE Trans. on Communications*, COM-7:1644-1659, November 1979.

[46] J. W. Modestino, D. G. Daut, and A. L. Vickers. Combined source-channel coding of images using the block cosine transform. *IEEE Trans. on Communications*, COM-29:1261-1273, September 1981.

[47] M. T. Orchard, Y. Wang, V. Vaishampayan, and A. R. Reibman. Redundancy rate-distortion analysis of multiple description coding using pairwise correlating transforms. In *Proc. IEEE Int'l Conf. on Image Processing*, pages 608-611, October 1997.

[48] International Standards Organization. Jpeg 2000 image coding system. Final Committee Draft of ISO International Standard 15444 Part 1.

[49] A. Ortego and K. Ramchandran. Rate-distortion methods for image and video compression. *IEEE Signal Processing Magazine*, 15:23-50, November 1998.

[50] V. Parthasarathy, J. W. Modestino, and K. S. Vastola. Design of a transport coding scheme for high quality video over ATM networks. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(2):358-376, April 1997.

[51] D. E. Pearson. Developments in model-based video coding. *Proc. of IEEE*, 86(6):892-906, June 1995.

[52] W. B. Pennebaker, J. L. Mitchell, and et. al. Arithmetic coding articles. *IBM Journal of Res. and Dev.*, 32(6):717-774, November 1988.

[53] E. J. Posnak, S. P. Gallindo, A. P. Stephens, and H. M. Vin. Techniques for resilient transmission of JPEG video streams. In *Proc. of Multimedia Computing and Networking*, pages 243-252, San Jose, February 1995.

[54] L. Rade and B. Westergren. *Mathematics Handbook for Science and Engineering*. Studentlitteratur Birkhauser, 1995.

[55] K. Ramchandran, A. Ortega, and K. M. Uz. Multiresolution broadcast for digital HDTV using joint source channel coding. *IEEE journal on Selected Areas in Communications*, 11:6-23, January 1993.

[56] K. Ramchandran and M. Vetterli. Best wavelet packet bases in a rate-distortion sense. *IEEE Trans. on Image Processing*, 2:160-175, April 1993.

[57] K. R. Rao and P. Yip. *Discrete cosine transform: algorithms, advantages, applications*. Harcourt Brace Jovanovich, Boston, MA, 1990.

[58] D. W. Redmill and N. G. Kingsbury. The EREC: an error resilient technique for coding variable-length blocks of data. *IEEE Trans. on Image Processing*, 5:565-574, April 1996.

[59] I. Rhee. Error control techniques for interactive low-bit rate video transmission over the Internet. In *Proc. SIGCOMM*, 1998.

[60] J. Ribas-Corbera and S. Lei. Rate control in DCT video coding for low-delay communications. *IEEE Trans. on Circuits and Systems for Video Technology*, 9:172-185, February 1999.

[61] J. I. Ronda, M. Eckert, F. Jaureguizar, and N. Garcia. Rate control and bit allocation for MPEG-4. *IEEE Trans. on Circuits and Systems for Video Technology*, 8:1243-1258, December 1999.

[62] A. Said and W. A. Pearlman. A new fast and efficient image codec based on set partitioning in hierarchical trees. *IEEE Trans. on Circuits and Systems for Video Technology*, 6:243-250, June 1996.

[63] S. D. Servetto. *Compression and reliable transmission of digital image and video signals*. PhD thesis, University of Illinois at Urbana-Champaign, 1999.

[64] S. D. Servetto, K. Ramchandran, V. A. Vaishampayan, and K. Nahrstedt. Multiple description wavelet based image coding. *IEEE Trans. on Image Processing*, 9:813-826, May 2000.

[65] J. M. Shapiro. embedded image coding using zerotrees of wavelet coefficients. *IEEE Trans. on Signal Processing*, 41(12):3445-3462, December 1993.

[66] Y. Shoham and A. Gersho. Efficient bit allocation for an arbitrary set of quantizers. *IEEE Trans. on Acoutics, Speech, and Signal Processing*, 36:1445-1453, September 1988.

[67] H. C. Shyu and J. J. Leou. Detection and concealment of transmission errors in MPEG-2 images - a genetic algorithm approach. *IEEE Trans. on Circuits and Systems for Video Technology*, 9(6):937-948, September 1999.

[68] T. Sikora. The MPEG-4 video standard verification model. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(1):19-31, February 1997.

[69] B. C. Smith. *Implementation Techniques for Continuous Media Systems and Applications*. PhD thesis, Univ. of California at Berkeley, 1994.

[70] J. Suh and Y. Ho. Error concealment based on directional interpolation. *IEEE Trans. on Consumer Electronics*, 43(3):295-302, August 1997.

[71] H. Sun and W. Kwok. Concealment of damaged block transform coded images using projections onto convex sets. *IEEE Trans. on Image Processing*, 4(4):470-477, April 1995.

[72] H. Sun, W. Kwok, M. Chien, and C. H. J. Ju. MPEG coding performance improvement by jointly optimizing coding mode decisions and rate control. *IEEE Trans. on Circuits and Systems for Video Technology*, 7:449-458, June 1997.

[73] R. Swann and N. G. Kingsbury. Transcoding of MPEG-II for enhanced resilience to transmission errors. In *Proc. IEEE Int'l Conf. on Image Processing*, pages 813-816, November 1996.

[74] R. Talluri, K. Oehler, T. Bannon, J. D. Courtney, A. Das, and J. Liao. A robust, scalable, object-based video compression technique for very low bit-rate coding. *IEEE Trans. on Circuits and Systems for Video Technology*, 7(1):221-233, February 1997.

[75] W. T. Tan and A. Zakhor. Real-time Internet video using error resilient scalable compression and TCP-friendly transport protocol. *IEEE/ACM Trans. on Networking*, 1:172-186, June 1999.

[76] M. J. Tsai, J. Villasenor, and F. Chen. Stack-run image coding. *IEEE Trans. on Circuits and Systems for Video Technology*, 6:519-521, October 1996.

[77] T. Turletti and C. Huitema. Videoconferencing on the Internet. *IEEE/ACM Trans. on Networking*, 4(3):340-351, June 1996.

[78] C. J. Turner and L. L. Peterson. Image transfer and end-to-end design. In *Proc. SIGCOMM'92*, pages 258-268, 1992.

[79] V. A. Vaishampayan. Design of multiple description scalar quantizers. *IEEE Trans. on Information Theory*, 39(3):821-834, May 1993.

[80] V. A. Vaishampayan. Application of multiple description codes to image and video transmission over lossy networks. In *Proc. 7th Int'l Workshop on Packet Video*, pages 55-60, March 1996.

[81] V. A. Vaishampayan and J. Domaszewicz. Design of entropy constrained multiple description scalar quantizers. *IEEE Trans. on Information Theory*, 40:245-251, January 1994.

[82] V. A. Vaishampayan and N. Farvardin. Optimal block cosine transform image coding for noisy channels. *IEEE Trans. on Communications*, 38:327-336, March 1990.

[83] G. K. Wallace. the JPEG still picture compression standard. *IEEE Trans. on Consumer Electronics*, 38(1), February 1992.

[84] Y. Wang, M. T. Orchard, and A. R. Reibman. Multiple description image coding for noisy channels by pairing transform coefficients. In *Proc. IEEE First Workshop Multimedia Signal Processing*, pages 419-424, June 1997.

[85] Y. Wang and Q. Zhu. Error control and concealment for video communications: a review. *Proc. of the IEEE*, 86(5):974-997, May 1998.

[86] Y. Wang, Q. Zhu, and L. Shaw. Maximally smooth image recovery in transform coding. *IEEE Trans. on Communications*, 41(10):1544-1551, October 1993.

[87] S. Wenger, G. Knorr, J. Ott, and F. Kossentini. Error resilience support in H.263+. *IEEE Trans. on Circuits and Systems for Video Technology*, 8(7):867-877, November 1998.

[88] B. Wohlberg and G. DeJager. A review of the fractal image coding literature. *IEEE Trans. on Image Processing*, 8(12):1716-1729, December 1999.

[89] J. Woods. *Subband image coding*. Kluwer Academic Publishers, 1991.

[90] Z. Xiong, K. Ramchandran, and M. T. Orchard. space-frequency quantization for wavelet image coding. *IEEE Trans. on Image Processing*, 6(5):677-693, 1997.

[91] Z. Xiong, K. Ramchandran, and M. T. Orchard. Wavelet packet image coding using space-frequency quantization. *IEEE Trans. on Image Processing*, 7:892-898, June 1998.

[92] Z. Xiong, K. Ramchandran, M. T. Orchard, and Y. Zhang. A comparative study of DCT- and wavelet-based image coding. *IEEE Trans. on Circuits and Systems for Video Technology*, 9(5):692-695, August 1999.

[93] G. Yu, M. M. Liu, and M. W. Marcellin. POCS-based error concealment for packet video using multiframe overlap information. *IEEE Trans. on Circuits and Systems for Video Technology*, 8(4):422-434, August 1998.

[94] W. Zeng and B. Liu. Geometric-structure-based error concealment with novel applications in block-based low-bit-rate coding. *IEEE Trans. on Circuits and Systems for Video Technology*, 9(4):648-665, June 1999.

[95] Q. Zhu, Y. Wang, and L. Shaw. Coding and cell-loss recovery in DCT-based packet video. *IEEE Trans. on Circuits and Systems for Video Technology*, 3(3):248-258, June 1993.

[96] W. Zhu, Y. Wang, and Q. Zhu. Second-order derivative-based smoothness measure for error concealment in DCT-based codecs. *IEEE Trans. on Circuits and Systems for Video Technology*, 8(6):713-718, October 1998.

Vita

Xiao Su received her B.S. degree in computer science and engineering from Zhejiang University, Hangzhou, P.R.China in 1994, and M.S. degree in computer science from the University of Illinois at Urbana-Champaign in 1997.

Xiao Su came to the University of Illinois in 1996. In working towards her M.S. degree, she studied and evaluated a window-based wireless contention protocol, which was shown to be more scalable and efficient than the IEEE 802.11 standard. For her Ph.D. research, she focused on the design and evaluation of robust coding, error concealment and delivery schemes for reliable and high-quality image and video transmissions on the Internet.

Her current research interests include video coding and transmission, computer networking and wireless communications, with an emphasis on error concealment schemes.

Upon completion of her Ph.D., she will join Inktomi Corporation to continue her work on content delivery for unreliable networks.