REAL-TIME TRANSMISSIONS OF JPEG 2000 IMAGES OVER THE INTERNET

BY

HANG YU

B.E., Zhejiang University, 2001

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Electrical Engineering
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2004

Urbana, Illinois

ABSTRACT

This thesis identifies the trade-offs encountered in transmitting images in real-time over the Internet and proposes two approaches to solve them. Image delivery over the Internet can use either the TCP or the UDP protocol. Delivery by TCP causes no image quality degradations, but has long delays when there are packet losses. On the other hand, although UDP has shorter delays, its delivery is not preferred because it does not recover lost packets during transmission, thus causing serious quality degradations.

In this thesis, we address the quality-delay trade-offs by using multiple description coding (MDC) and unequal error protection (UEP), together with the UDP protocol, to deliver JPEG 2000 images over the Internet. We first analyze Internet traffic data and determine a proper interleaving factor by which UDP packets are interleaved in order for losses to be below a certain threshold. We then propose an MDC scheme with a new reconstruction strategy carried out in the frequency domain, thereby avoiding sample-domain segmentation, which is a major cause for performance degradations in previous sample-domain MDC schemes. An optimized reconstruction-based transform for the new scheme is also derived and incorporated into our proposed MDC scheme. We examine the performance of our MDC algorithm under controlled-loss scenarios and real Internet trace-driven simulations, both of which perform quite well. We also study the quality-delay behavior of our proposed scheme under UDP and compare it with that of the traditional TCP delivery. The results show that this new MDC scheme can be an attractive alternative to the traditional approach.

Last, we study UEP schemes that pack code-blocks and FEC codes together in both serial and parallel styles. Based on this study, we propose a hybrid packing algorithm that applies serial or parallel packing for different parts of a code stream. For UEP schemes, we test our proposed hybrid packing algorithm under several measures, namely, PSNR, undecodable probability, and sensitivity. The results show that our proposed hybrid packing scheme is favorable as compared to that of TCP.

To my dear wife and parents

ACKNOWLEDGMENTS

First I would like to thank my thesis adviser, Professor Benjamin W. Wah. It is with his insightful suggestions and consistent encouragement that I was able to start, continue with this research, and finally complete this thesis. I have always benefited from his professionalism and unwaveringly high standards.

Special thanks go to Xiao Su and Dong Lin for their kind help to me when I first started this study. I am very grateful to their valuable opinion in our discussions and correspondences.

I also want to thank all the members of my research group, especially Minglun Qian, Yixin Chen, and Dong Xin, for their valuable opinion and all the useful group meetings.

I am fully indebted to my dear wife, Lei Wang, for her everlasting love and support during my study. I would also like to thank my parents for their encouragement and motivation. They are always the source where my strength comes from.

Finally, I would like to acknowledge the financial support of the Motorola Communication Center, University of Illinois at Urbana-Champaign.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

Figure **Page**

1 INTRODUCTION

1.1 Motivations

As the World Wide Web (WWW) experiences an incredible growth today, real-time multimedia applications are becoming more important to users. Among all these applications, real-time image delivery plays a fundamental role.

Currently, there are two major transport protocols that can be used for transmitting image data over the Internet: the Transmission Control Protocol (TCP) and the User Datagram Protocol (UDP). TCP is widely used in current Web servers to insure the quality of received images. However, the speed of transmitting images by TCP is slow because TCP's retransmission and back-off mechanisms usually cause long delays when packets are lost. On the other hand, UDP can transmit images fast but does not guarantee their quality because it does not have mechanisms to retransmit packets when they are lost. Thus, neither TCP nor UDP alone is suitable for real-time image transmission applications. This motivates us to find a solution in order to transmit an image with short delay but good image quality.

1.2 Problem Statement

In this thesis, we study the quality-delay trade-offs of transmitting images in real time over the Internet, without special hardware support. The most important characteristic of this problem is its real-time aspect. Although image transmission was traditionally thought to be a non real-time application, people today realize that receiving an image in a short time with a little quality degradation is much more desirable than receiving an image with perfect quality after a long delay. This is true because the human visual system can tolerate some image quality degradations. Another property of this problem is that real-time image transmissions do not have jitters to correct in its delivery, which is important in real-time video and audio transmissions. The order in which different parts of an image are received usually does not affect image quality.

Several new protocols have been devised to facilitate real time applications. These protocols can

be classified into application-layer and network-layer protocols. Two major application-layer real-time protocols are the Real-time Transport Protocol (RTP) and the Real-time Transport Control Protocol (RTCP). These two protocols are built on top of UDP and provide feedback for real-time applications, but do not guarantee delivery because the underlying physical link is still error-prone. Another network-layer protocol, which includes the Internet Protocol version 6 (IPv6), has flow-labeling ability and a traffic class field. These can be used to determine the type of data a packet carries. However, IPv6 does not guarantee delivery because it does have mechanisms to recover from errors in unreliable physical links. Since none of the existing Internet protocols provides reliable data delivery, we choose the UDP protocol in this thesis because UDP provides short delays in packet delivery.

The most important measures of image transmission over the Internet are quality and delay. The quality of an image is usually reflected by subjective or objective measures. Although subjective measures are closer to what a human perceives of an image, ithey are usually difficult to quantify and may vary among different people. Objective measures, on the other hand, are usually easy to evaluate and only depend on the image itself. One of the most commonly used objective measures is the peak-signal-to-noise ratio (PSNR), which we use in the thesis. Besides image quality, another important measure of image transmission over the Internet is its delay, or the end-to-end delay. The end-to-end delay of image transmission consists of the algorithmic delay and the network delay. The algorithmic delay is the time a coder takes to encode and decode an image, which is usually determined by the image size and the image coders. The network delay is the time between a packet is transmitted by the sender and is received correctly by the receiver. It is usually the dominant part of the total delay for an image sent over the Internet.

Different Internet protocols achieve different quality and delay trade-offs. For instance, image delivery by TCP has perfect image quality, but with long delay. On the other hand, image delivery by UDP has a short delay, but the decoded image may have a poor quality. Today's real-time applications favor a good image quality but with a short delay. Neither TCP nor UDP alone can provide such an option. In this thesis, we study the quality-delay trade-offs of several new delivery strategies.

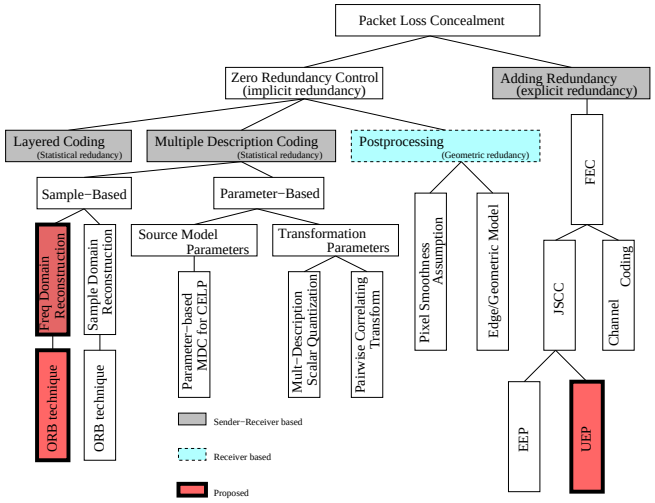


Figure 1.1 A classification of packet loss concealment strategies. The boxes with thick borders are what we study in this thesis.

1.3 Previous Work

In the literature, packet loss concealment strategies can be classified into two categories: explicit redundancy and implicit redundancy, as shown in Figure 1.1.

The strategy of using explicit redundancy usually involves using forward error correcting (FEC) codes, by which explicit redundancy is added to a code stream. We can further classify this strategy by how the amount of external redundancy is determined: channel coding and joint source channel coding (JSCC). In channel coding, the amount of FEC codes is determined only by the property of the channel. With this, it is easy to control the amount of redundancy to add according to the channel situation and independent from source coding. In JSCC, the amount of FEC codes is determined by the channel and the source together. It takes the property of the source into consideration so that the amount of FEC codes added can be directly related to the distortions of the source when data is decoded. In [1, 2], the authors used Reed-Solomon (RS) codes and designed an unequal error protection (UEP) scheme to protect images transmitted in a packet network. However, this scheme assumes a known channel model that is difficult to derive for the Internet.

The strategy of using implicit redundancy utilizes implicit redundancy in the original source to do recovery when loss happens. There are several zero redundancy control strategies, which are classified by the type of redundancy they utilize. The first kind of strategies uses the inherent redundancy in an image. In [3], the authors assumed a convex image boundary and used a computationally expensive scheme called projection onto convex set (POCS) to reconstruct lost pixels. Reconstruction using edge models are proposed in [4, 5, 6, 7] by assuming that edges in an image are usually smooth. This assumption is used as *a priori* information to do reconstruction; however, the computation cost is high.

There are two other coding schemes that utilize statistical redundancies of image pixels. One is layered coding (LC), and the other is multiple description coding (MDC). In LC, a source image is cut into several layers with decreasing importance. The first and the most important layer is called the *base layer* that can be used to reconstruct a rough version of an image. The other layers, called *enhancement layers*, can be used to refine the image reconstructed using the base layer. LC must have underlying network support that drops packets containing enhancement layers first when congestion happens. Since this kind of quality-of-service (QoS) support does not exist in the Internet, LC cannot be used in concealing IP packet losses.

Unlike LC, MDC [8] decomposes a source image into equally important parts before trans-

mitting them. Each of these parts is called a *subdescription*. When some of the subdescriptions are received, the image can be reconstructed with acceptable quality; when all the subdescriptions are received, they can produce better quality. In [9] the author used two scalar quantizers, called *multiple description scalar quantizer* (MDSQ), to quantize the frequency coefficients of an image. Further work on MD quantizers can be found in [10], in which the author designed the quantizers under the source entropy constraint. The problem of MDSQ is that the design of a good quantization index assignment for the two scalar quantizers is quite difficult. The idea of pairwise correlating transform (PCT), proposed in [11] and generalized in [12], is to apply a correlating transforms on the frequency coefficients and reconstruct lost frequency coefficients using this correlation. The original idea of PCT was used in Discrete Cosine Transform (DCT) and is further extended to wavelet transforms in [13]. Similar to this approach, the author of [14] applied data fusion technique in the frequency domain to correlate the coefficients, and the author of [15] partitioned the wavelet domain before correlating the wavelet coefficients to pursue extra gains. As this correlation may not be high enough in some cases, the quality of a reconstructed image may not be good when only some of its subdescriptions are received. In [16] a parameter-based MDC for Code-Excited-Linear-Predictive (CELP) coding was proposed. This parameter-based MDC utilizes redundancy in the parameters of the vocal-tract model of human speech, but cannot be used for images because images cannot be well modelled by a few parameters. Another MDC strategy is the sample-based MDC, in which image pixels are interleaved into different subdescriptions, encoded, and transmitted. When one subdescription is lost, its lost pixels can be reconstructed by interpolating pixels from other subdescription(s) received. This approach works well for images, since image samples usually have a high correlation.

One of the problems in sample-based MDC is the segmentation problem. Segmentation is needed when we transmit large images using UDP because a UDP packet, including its header, transmitted over the Internet needs to be smaller than 556 octets in order not to be fragmented. Each UDP packet has to be independently decodable from other packets, since no packet is guaranteed to be received by the receiver. Segmentation can be done either before or after image pixels are interleaved because segmentation and interleaving are both linear procedures, and their order can be exchanged without affecting the output. Table 1.1 illustrates the effect of segmentation that prevents high sample-domain redundancy within one subdescription to be exploited in another subdescription. We can see from this table that, with segmentation, the quality of decoded images

Table 1.1 The PSNR of a 512×512 8-bit gray-scale *lena* image compressed by JPEG2000 at 5 different bit rates using 3 different segment sizes. Here, we assume that no segment is lost. When the segment size is set at 64×64 and compressed at 0.25 bpp and 0.125 bpp, respectively, there is not enough bit budget even for the JPEG2000 headers.

Segment Size	Bits Per Pixel (bpp)				
	2	1	0.5	0.25	0.125
No segmentation	43.91	40.07	37.16	34.08	30.97
256×256 segments	43.51	39.34	36.02	32.76	29.41
128×128 segments	42.55	37.93	33.92	29.42	24.14
64×64 segments	40.01	33.85	27.01	–	–

degrades a lot even when there is no loss.

In MDC of images, image samples can be divided into multiple descriptions before each description is subband coded. In this approach, an image is first segmented before interleaving subimage samples. Each interleaved subimage is then coded by subband transform and packetized. At the receiver end, received packets are decoded before reconstructing lost pixels by average interpolations. Further in [17], an Optimized Reconstruction-Based Subband Transform (ORB-ST) was proposed. Instead of minimizing distortions by assuming no loss, as in standard encoders, ORB-ST is designed to minimize distortions by assuming one of the descriptions is lost.

ORB-ST is one of a collection of schemes in the category of the sample-domain MDC techniques. All sample-based MDC schemes can be characterized by the following four attributes: the availability of channel statistics, the reconstruction method, the reconstruction domain, and whether ORB-ST is applied. These four attributes should be decided according to the application requirements. The first attribute is channel loss statistics, which is difficult to get in practice, although the long-term average packet loss rate within a certain period of time may be available via receiver feedbacks. The second attribute is the reconstruction method used. In addition to interpolations, reconstruction methods include padding by zeroes and duplication. The best reconstruction method is generally application dependent. The third attribute is whether to do reconstruction in the frequency or the sample domain. The last attribute is whether to apply ORB-ST or not and it should be considered based on the complexity and the effectiveness of the ORB-ST technique.

1.4 Approach

In this thesis, we study the problem of coding and reconstructing images in order to protect them from losses during transmissions over the Internet. We propose a novel MDC algorithm to protect and reconstruct information lost during transmission. The MDC algorithm is frequency-domain based in order to eliminate the annoying segmentation problem discussed in the last section.

In addition to MDC, we also study the UEP scheme. Since FEC codes have long been used to protect an image evenly, not much research has been done on unequal error protection schemes. In this thesis we discuss several packing algorithms of UEP and propose a new hybrid packing algorithm. This UEP algorithm can be applied to real-time image delivery over the Internet.

1.5 Contributions of This Thesis

The contributions of this thesis are on a novel frequency-based reconstruction scheme for MDC coded JPEG 2000 images and a novel hybrid packing UEP algorithm.

The proposed frequency-based reconstruction scheme does not require images to be segmented in the coding process, and performs reconstruction in the frequency domain instead of the sample domain. This approach overcomes the segmentation problem and can perform better than the sample-domain reconstruction approach.

Our proposed hybrid packing UEP algorithm packs code blocks and FEC codes using a number of strategies. In addition to adding a different amount of FEC codes to different parts of a code stream, this approach further applies different packing styles to different parts of the code stream. It can improve the average decoded quality of an image and decrease its undecodable probability.

1.6 Organization of This Thesis

Chapter 2 introduces the property of the JPEG 2000 coder. Chapter 3 analyzes the Internet traffic. Chapter 4 describes the frequency-based reconstruction algorithm and our experimental results. Chapter 5 present our UEP algorithms. Finally, Chapter 6 provides our conclusions and directions for future research.

2 AN INTRODUCTION OF JPEG 2000 CODERS

2.1 From JPEG to JPEG 2000

Over the past few decades, Joint Photographic Experts Group (JPEG) [18], developed by the International Standards Organization and the International Electrotechnical Commission (ISO/IEC), has found wide acceptance in diverse application areas, including the Internet, digital cameras, and printing and scanning devices. However, as low bit-rate image coding is increasingly demanded by modern applications, JPEG suffers from its poor performance because its block-based Discrete Cosine Transform (DCT) has serious blocking effects under low bit-rate coding situations.

The JPEG 2000 standard [19] is intended to succeed JPEG in many application areas. It is motivated primarily by the need for compressed-image representations that offer new features increasingly demanded by modern applications, while also offering superior compression performance, especially at low bit-rates. The JPEG 2000 standard uses a wavelet transform, which provides more powerful compression and better image quality at low bit-rate. The structure of JPEG 2000 image's subband decomposition makes possible the progressive transmission of an image by either resolution or region.

In this chapter, we overview the JPEG 2000 system architecture depicted in Figure 2.1. (Details can be found in [19].) The JPEG 2000 encoding system consists of four main stages: preprocessing, wavelet transform, quantization, and entropy coding. The decoding system also consists of four main stages, which are basically the reverse of the encoding system.

2.2 JPEG 2000 Encoding Process

Before we describe the JPEG 2000 encoding process, we introduce some notations. First, a 2D image is represented by $f(x, y)$ in this chapter, where x and y are the row and column indices, and $f(x, y)$ is a scalar for a gray scale image. For color images, $f(x, y) = [f_R(x, y), f_G(x, y), f_B(x, y)]$ is a vector. Its three components represent the intensity of red, green, and blue (RGB) colors, respectively. In this thesis, an image is a gray-scale image by default, if its type is not specified.

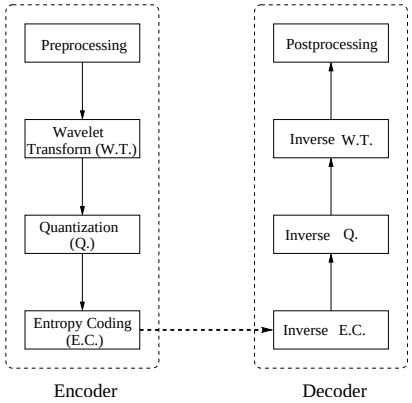
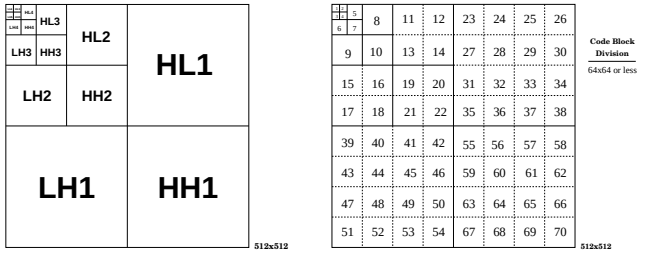


Figure 2.1 JPEG 2000 encoding and decoding stages.

The range of the value of $f(x, y)$ depends on the depth of each sample. Usually we assume a pixel has eight bits; hence, $f(x, y)$ is between $[0, 255]$.

2.2.1 Preprocessing

In a JPEG 2000 encoder, two basic operations are performed during preprocessing: level shift and color-component transform. The purpose of the level-shift procedure is to remove the DC component of an image. For an 8-bit image, level shift can be represented by $f(x, y) = f(x, y) - 128$. A color-component transform is applied after the level-shift operation if the image is a color image. This transform is designed to remove the redundancy between different color components by transforming the three RGB color components into one luminance and two color difference components. Usually, the two color-difference components account for less than 20% of the total bit budget. Two types of color component transforms are available in JPEG 2000. One is the Irreversible Color Transform (ICT), which uses a floating point transform and is irreversible. Another one is the Reversible Color Transform (RCT), which is specially designed for lossless compression. RCT is a strict integer-to-integer mapping and can be reversed.



(a) five level decomposition (b) Code-block organization

Figure 2.2 An example of wavelet decomposition and code-block organization.

2.2.2 Wavelet transform

The core component of the JPEG 2000 image encoding system is the wavelet transform (WT). The default wavelet used is the Cohen-Daubechies-Feauveau (CDF) 9/7 wavelet [20] , which is a biorthogonal wavelet. In the mode of lossless compression, a reversible wavelet transform is used. This special reversible wavelet transform is an integer-to-integer mapping, which can be strictly reversed without any loss of precision. In the future, more wavelet transforms will be included in the standard.

A wavelet transform is used to obtain a typical subband decomposition of an image. A subband decomposition consists of filtering and up/down-sampling, which are both linear processes. The 2D decomposition is done by applying the wavelet transform in one dimension followed by the other. Figure 2.2a shows the structure of a subband decomposed image, where the original image size is 512×512 and the decomposition level is set to 5 by default in JPEG 2000. Since the wavelet used is biorthogonal, the subband energy in each subband adds up approximately to the total energy.

2.2.3 Quantization

After the wavelet transform, we obtain the subband coefficients or wavelet coefficients of the original image. Before these subband coefficients can be further quantized, each single subband is divided into code-blocks, which are coded independently. This code-block organization allows errors to propagate only within a code-block.

Figure 2.2b shows the code-block organization of the subband structure in Figure 2.2a. Code

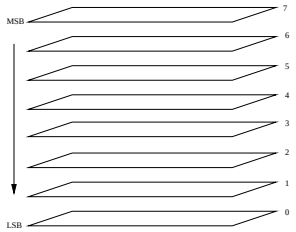


Figure 2.3 Bit plane decomposition of a code-block.

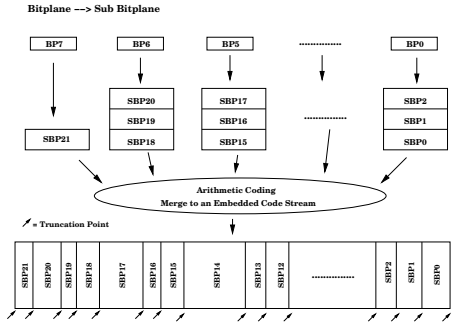


Figure 2.4 Sub-bit-plane grouping.

blocks are generally in a square shape, and their sizes are user specified. The size of a code-block cannot exceed 64×64 in order to prevent long error propagations.

After the coefficients are grouped into code-blocks, they are quantized by the JPEG 2000 encoder. First, a scalar dead-zone quantization method is used. This scalar quantization uses small quantization steps so that coefficients will not lose too much precision. This is unlike the quantization method of JPEG, which controls the compression ratio by adjusting the quantization steps for different frequency bands. In JPEG 2000, controlling the output bit-rate is largely manipulated by the rate-distortion control mechanism, which is described later.

After the rough scalar quantization, the coefficients in each code-block are first cut into bit-planes as shown in Figure 2.3. The bits in each bit-plane are further grouped into three sub-bit-

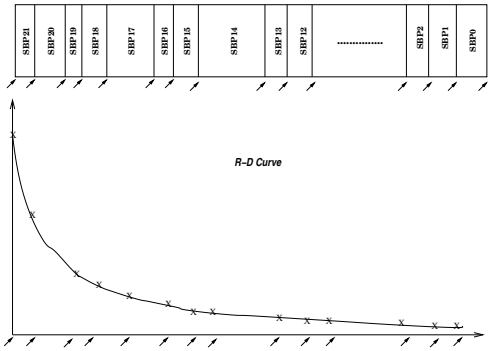


Figure 2.5 Approximating the local rate-distortion curve.

planes according to their significance. The sub-bit-planes are then used for further rate-distortion (R-D) optimization.

Figure 2.4 shows an example of this process. In this figure, **BP0** to **BP7** are the eight bit-planes of a code-block, with **BP7** being the most significant bit-plane. Each bit in a bit-plane has different significance, which is determined by its value and the pattern of its eight adjacent bits. According to its significance, the bits can be classified into three groups, with each group in a sub-bit-plane. In the figure, the sub-bit-planes are denoted by **SBP0** to **SBP21**. As **BP7** is the most significant bit-plane, it is not further divided and has only one sub-bit-plane **SBP21**.

Next, the rate distortion (R-D) control mechanism is applied. As obtaining a global R-D curve for the whole image is practically impossible, JPEG 2000 uses local R-D curves of each code-block to do rate distortion control. These local R-D curves are approximated by calculating the size and contribution of each sub-bit-plane in a code-block, as shown in Figure 2.5. The contribution of a sub-bit-plane is quantified by how much distortion it can reduce, and its size is the number of bits occupied by this sub-bit-plane. In Figure 2.5, starting from **SBP21**, one more sub-bit-plane is added in each step during which the distortion and the rate are calculated. These two values are kept for each code-block for further global R-D control.

The problem of the global R-D control is to find for every code-block the set of sub-bit-planes that maximizes their contributions, under the constraints of the given total bit budget R_0. JPEG

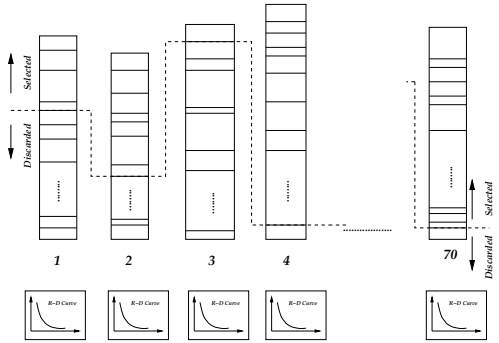


Figure 2.6 Heuristic to get the optimal R-D trade-off point.

2000 uses a near optimal heuristic to solve this problem. Assuming d_i and r_i are the contribution and the size of a sub-bit-plane, respectively, and the weight of a sub-bit-plane is defined as $w_i = d_i/r_i$, the algorithm uses binary search to find w^* such that all sub-bit-planes having w_i larger than w^* have a total size smaller than R_0.

An example is shown in Figure 2.6. In this example, we assume 70 code-blocks in an image, and all the sub-bit-planes below the dashed line are discarded, whereas all those above the dashed line are chosen. The dashed line is adjusted in each step by the binary search.

Figure 2.7 gives the detailed algorithm. Here N is the total number of sub-bit-planes of the image, and e_0 is the stopping criteria and controls the algorithm's precision and complexity. In the algorithm, the upper and lower bounds to the threshold are first established (**Line 10-11**), and the current threshold is set at the middle (**Line 12**). A code block is included if its weight exceeds this threshold (**Line 20**). The threshold is lowered if the total bit rate exceeds the bit budget (**Line 24-25**) or raised if it does not(**Line 27-28**). This loop continues until the bit rate is within a certain range to the given bit budget (**Line 15**).

2.2.4 Entropy coding

After all the sub-bit-planes are selected, those within one code-block need to be further entropy coded. The JPEG 2000 encoder uses an arithmetic coder, which can produce embedded arithmetic


```
1   INPUT:
2   R_0,                                  /* Total bit budget */
3   w_i,                                  /* Weight of each code-block */
4   s_i,                                  /* Size of each code-block (in bits) */
5   e_0,                                  /* stopping criteria */
6   N                                     /* Number of code-blocks */
7   OUTPUT:
8   w*,                                   /* Threshold. Code blocks having a
                                             weight larger than this are selected. */
9   BEGIN
10      T_max = max_i(w_i);               /* Initialize the higher threshold */
11      T_min = min_i(w_i);               /* Initialize the lower threshold */
12      T_1 = (T_max + T_min)/2;          /* Initialize the current threshold */
13      i = 0;                            /* Current iteration index */
14      R = 0;                            /* Current bit rate */
15      WHILE R_0 − R > e_0               /* Iteration continues if the current
                                             bit rate is not close enough to
                                             to the given bit budget */
16          i = i + 1;
17          R = 0;                        /* Clear current bit rate */
18          FOR j = 1 TO N
19              IF w_j ≥ T_i THEN
20                  R = R + s_j;          /* If the weight of a code-block is larger
                                             than the threshold, add its size
                                             to the current bit rate */
21              ENDIF
22          ENDFOR
23          IF R > R_0 THEN               /* If the bit budget is exceeded,
                                             lower the current threshold */
24              T_{i+1} = (T_i + T_min)/2;
25              T_max = T_i;
26          ELSE                          /* If the bit budget is not exceeded,
                                             raise the current threshold */
27              T_{i+1} = (T_i + T_max)/2;
28              T_min = T_i;
29          ENDIF
30      END WHILE
31      w* = T_i;
32  END
```

Figure 2.7 R-D Control algorithm.

codes. The advantage of using arithmetic codes rather than Huffman codes is that the coder does not need to transmit fixed codebooks, thereby saving bandwidth in low bit-rate encoding.

2.3 JPEG 2000 Decoding Process

The decoding processes of JPEG 2000 is mainly the reverse procedures of the previous encoding stages. The first step is inverse entropy coding, followed by the decoding of the wavelet coefficients. The second step is the inverse wavelet transform. After this step, the original DC-removed image is decoded, and the inverse color-component transform is applied if the image is a color image. Finally the removed DC component is added back with clipping if necessary.

2.4 Summary

In this chapter, we have overviewed the JPEG 2000 system. We show that the wavelet transform with subband decomposition is a linear transform, and all the processes, such as filtering and up/down-sampling in wavelet transform, can be represented by matrix multiplications. This linear property allows us to formulate and solve the ORB-ST problem because the encoding and decoding process together can be considered as a linear process, assuming that quantization and de-quantization are perfect and lossless. We also show that JPEG 2000 has provided independent decodable units, called code-blocks, that can help prevent error propagation. This property enables us to put code-blocks in different UDP packets because the loss of a single UDP packet will not affect the decoding of code-blocks in other UDP packets.

3 INTERNET TRAFFIC STUDY

In this chapter, we study the basic traffic behavior of the Internet, namely, the end-to-end delay of both TCP and UDP and the loss characteristics of UDP. Also we collect Internet traces that are needed in our trace-driven simulations.

3.1 Experimental Setup

The objective of our experiments is to collect and analyze data from real Internet connections. We set up a few Internet connections and collect statistics on their traffic statistics, both for TCP and UDP.

Four remote computers, located either within or outside the US and represent very low, low, medium and high loss connections, are chosen. From a computer located at the University of Illinois at Urbana-Champaign (UIUC) (`trace18.crhc.uiuc.edu`), we connect to the four remote hosts with their echo ports opened: New Jersey (`www.princeton.edu` with less than 1% loss on average), Taiwan (`pager.mit.com.tw` with around 5% loss on average), Thailand (`www.kmitnb.ac.th` with around 15% loss on average), and Argentina (`www.legisrn.gov.ar` with around 35% loss on average).

We have implemented the test bed described in [17] to carry out our experiments. In this test bed, we send probe packets from the computer located at UIUC to the echo port of each remote computer. For each probe packet, we add a sequence number and a time stamp. Based on the probe packets bounced back by the remote computers to our local computer at UIUC, we use their time stamps to calculate the delay and the sequence numbers to identify packet losses.

The above packet-echoing approach works fine for UDP but is not accurate for TCP. In our experiment, the process of bouncing packets to a remote computer and back is meant to emulate either a single virtual TCP connection or a UDP path that is twice as long as the path from the local computer to the remote computer. This approach is accurate for UDP because a UDP echo port echoes incoming packets immediately. However, in TCP, a TCP packet actually travels

through two virtual connections, each having their own window-based flow control mechanisms. As a result, its timing result is not accurate.

To make a fair comparison between TCP and UDP, we need each TCP probe packet to travel a single virtual path with only one window-based flow control. To achieve this, we follow the method described in [17] to encapsulate a TCP segment into a UDP packet. We intercept every TCP segment and encapsulate it into a UDP packet, before the kernel encapsulates this UDP packet in an IP packet. The encapsulated packet is then handed to the IP layer as usual and sent to the remote UDP echo port that bounces it back immediately. When the kernel has received the bounced-back UDP packet, it removes its UDP header and hands it to the TCP management module. This modification was implemented in Redhat Linux 8 with kernel version 2.4.19. After this modification, both TCP and UDP packets only travel a single virtual path to a remote computer and back.

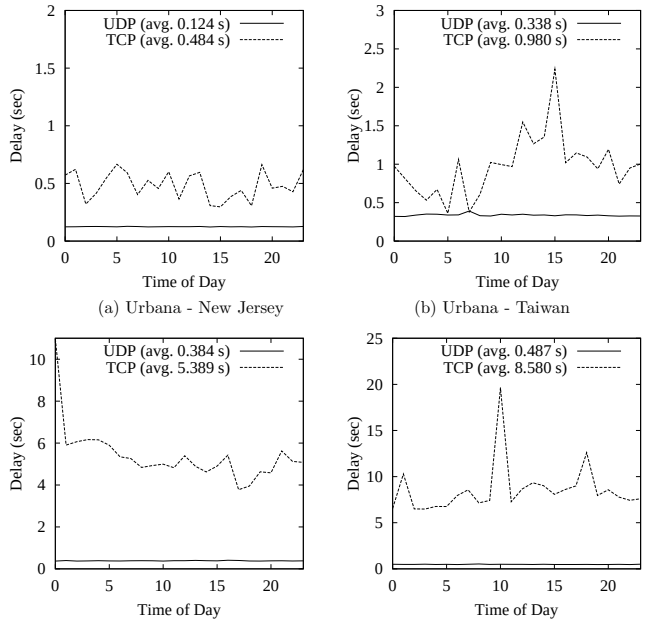
3.2 Comparison of End-to-End Delays

Figure 3.1 shows our experimental results. This experiment was done during the 24-hour period on April 23, 2003. Two thousand TCP/UDP probe packets were sent over each of the four connections, and their delay times are shown in Figure 3.1. In the experiments, we sent packets in 20-ms intervals. Although image transmissions do not have natural time-interval constraints, we have chosen a rate of 20 ms because sending data in shorter intervals will flood the receiver's buffer and incur significant losses.

From the figure, we observe that UDP usually has small delays with less variations, whereas TCP has long delays and large variations. For instance, UDP packets going from Urbana to New Jersey only have an average delay of 124 ms, but TCP packets have an average delay of 484 ms, which is about 4 times longer than that of UDP. Moreover, the delays of UDP packets at each hour do not vary much, whereas the delays of TCP packets vary between 300 ms and 600 ms. TCP's long delays and large variations are attributed to TCP'a coarse grained timeout and congestion avoidance algorithms. All other three cases show similar results.

3.3 Loss Behavior of UDP Transmissions

Although UDP is faster than TCP in transmitting packets, it may suffer packet losses because UDP performs no retransmissions when packets are lost. In order to understand the behavior of UDP packet losses, we collect traces of UDP packet transmissions and study them with respect to



(a) Urbana - New Jersey (b) Urbana - Taiwan

(c) Urbana - Thailand (d) Urbana - Argentina

Figure 3.1 Average round-trip delay of sending 2000 UDP packets and 2000 encapsulated TCP packets to the UDP echo ports of four remote computers. The experiments were done at the beginning of every hour on April 23, 2003.

three properties: packet loss rate, variation of packet loss rate, and burst length of packet loss.

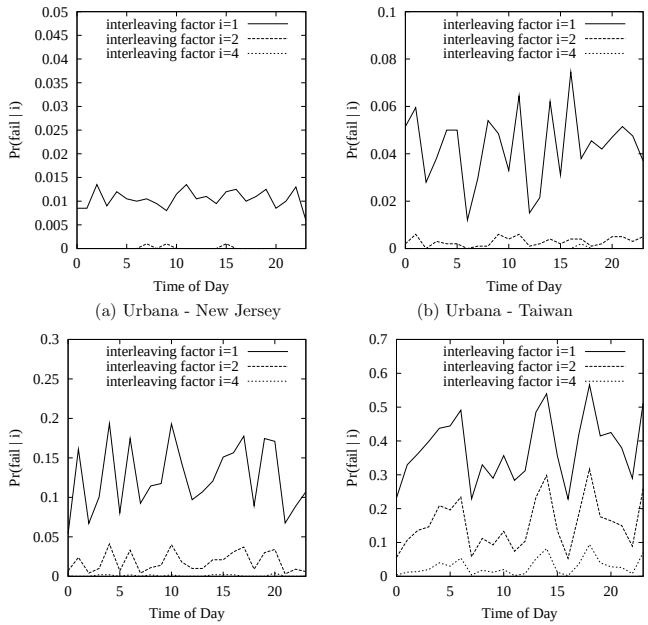
Figure 3.2 shows our experimental results. All four cases show that the packet loss rate can vary from very low to very high. Even for one connection, the loss rate can vary at different times of the day. We also depict the cumulative probability of bursty losses in Figure 3.3, which show that bursty losses exist in each connection.

To conceal packet losses in UDP, we interleave related information into unrelated packets. (Note that interleaving is a form of MDC coding). In general, an interleaving factor i allows reconstructions by interpolation of a burst length less than i packets if losses are from the same interleaved set, or a burst length in the range $[i, (2i-2)]$ packets if losses are from different interleaved sets. Since a receiver will incur more delay time and coding efficiency with a large i, i must be chosen in such a way that the probability of packets that are not recoverable using this i is kept small enough, which is set to 5% in this experiment.

Figure 3.2 shows the result of loss probability conditioned on interleaving factor i, $Pr(fail|i)$, which is defined as $Pr(fail|i) = (i \times m_i^i)/n_p$ in [17]. Here, we assume sending n_p packet in all, and the loss of j consecutive packets over an interleaved set happen m_j^i times. From the figure, we see that, for very low-loss connections, using UDP with no interleaving would be adequate. For low-loss connections, an interleaving factor of 2 is enough to conceal most of the losses. For medium-loss connections, $i = 2$ is also enough to lower the loss rate to under 5%. Finally, for high-loss connections, we need $i = 4$ in order for for the average loss rate to be smaller than 5%. From our experiments, we conclude that an interleaving factor of 2 to 4 is enough for most Internet connections.

3.4 Summary

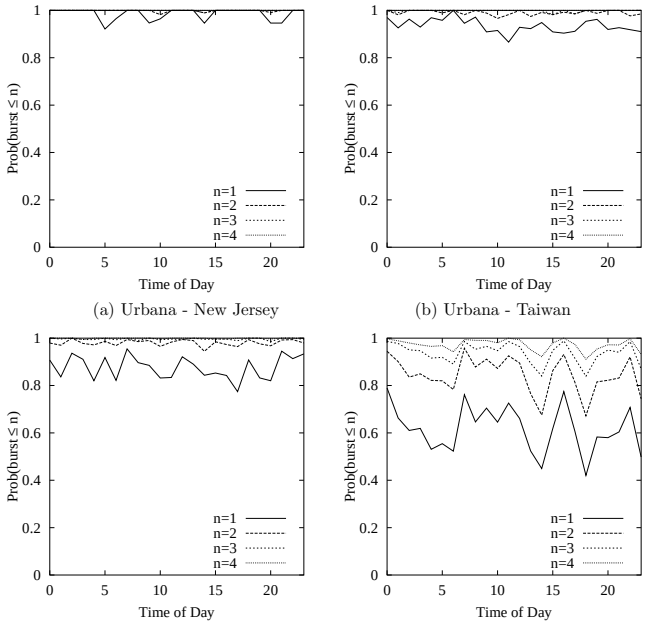
In this chapter, we have focused on the study of Internet traffic. We collect traffic data from real Internet connections for both TCP and UDP. We analyze the delay of TCP and UDP and the loss statistics of UDP. We find out that an interleaving factor of 2 is enough to conceal most of the losses for low and mediumi-loss connections, and an interleaving factor of 4 is needed for high-loss connections. This conclusion can help in our design of MDC schemes in the next chapter. We will design two-way MDC scheme in low and medium loss scenarios and four-way MDC scheme in high-loss scenarios.



(a) Urbana - New Jersey (b) Urbana - Taiwan

(c) Urbana - Thailand (d) Urbana - Argentina

Figure 3.2 $Pr(fail|i)$, probability of bursty losses that cannot be recovered, conditioned on interleaving factor i, at the beginning of every hour of April 23, 2003 to the echo ports of four remote computers.



(a) Urbana - New Jersey (b) Urbana - Taiwan

(c) Urbana - Thailand (d) Urbana - Argentina

Figure 3.3 $Pr(burst \leq n)$, cumulative probability of bursty losses at the beginning of every hour of April 23, 2003 to the echo ports of four remote computers.

4 PROPOSED FREQUENCY-BASED MDC ALGORITHM

In this chapter, we present our proposed frequency-domain reconstruction-based MDC scheme. First, we review the segmentation problem of previous MDC schemes and show the system architecture of our proposed scheme. In Section 4.2, we study the correlation of two subdescriptions in each subband and compare different reconstruction methods in a single subband. In Section 4.3, we derive a modified ORB-ST scheme for our proposed frequency-based MDC. Next, we give a detailed derivation of a subband reconstruction method in Section 4.4. Finally, we discuss some alternatives of this scheme and show our experimental results in Section 4.5.

4.1 Overview

In this section, we first review the segmentation problem in previous MDC schemes and present the problem statement. Next, we explain the system architecture of our proposed scheme.

4.1.1 Problem Statement

As illustrated in Chapter 1, one of the problems in sample-based MDC schemes is the segmentation problem. In general MDC+UDP schemes, a large image has to be segmented into small subimages before subdescriptions are generated. This segmentation operation will result in performance degradations for the decoded image because it prevents redundancies between subimages to be removed. Our goal in chapter is to solve this problem by developing an MDC scheme that can get rid of the segmentation operation and can achieve better decoded image quality.

We propose a new MDC scheme to solve the segmentation problem and derive an ORB-ST transform and a subband reconstruction technique to improve the performance of our scheme in this chapter. Our approach to solve the segmentation problem is to do reconstruction in the frequency domain in order to avoid the segmentation operation. We have made the following assumptions in this chapter:

Assumption 4.1: A compressed JPEG 2000 code stream contains independently decodable units

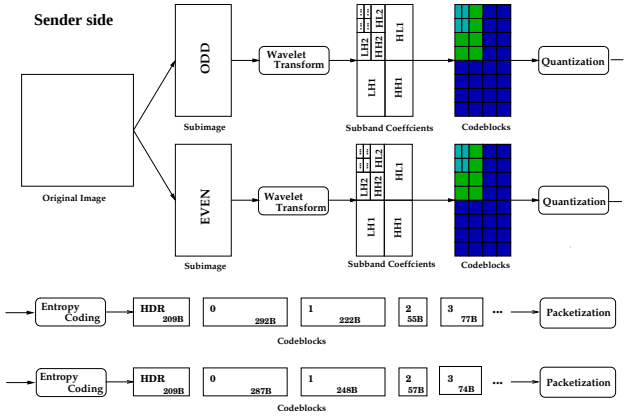


Figure 4.1 The sender side of our frequency-based reconstruction system. HDR is the header information of the image.

called *code blocks*, which we have discussed in Chapter 2.

Assumption 4.2: We assume that a receiver can provide feedbacks to the sender and that the sender can get the feedbacks after some delay, which are determined by the connection. The feedback information can be used by the sender to improve its reconstruction quality by adjusting the transforms at the sender. The type of the feedback information will be given in Section 4.3 when we discuss the use of feedbacks.

Assumption 4.3: We assume that the reconstruction can be done in either the sample domain or the frequency domain.

4.1.2 System Architecture

We describe and explain the overall system architecture, shown in Figures 4.1 and 4.2, in this subsection. These two figures illustrate the sender and the receiver sides of a two-way MDC coding system, respectively.

At the sender side, we first interleave the pixels of the original image and obtain an odd and an even subdescriptions. These two subdescriptions are further decomposed into subband representations by wavelet transforms, and the subband coefficients are grouped into code blocks. After

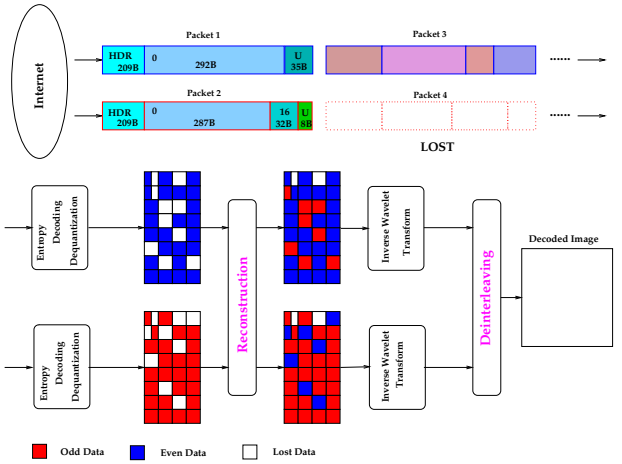


Figure 4.2 The receiver side of our frequency-based reconstruction system. The code blocks in each UDP packet are shown with its size (in bytes). U stands for unused data.

the code blocks are quantized and entropy coded, they are put into UDP packets in such a way that the corresponding code blocks from different subdescriptions will be in the same interleaved set. As each code block contains different subband coefficients, they have different contributions to the image quality, with the code block containing the lowest subband coefficients being the most important one. The header of JPEG 2000 will be combined with this code block because losing either of them will render a subdescription undecodable.

At the receiver side, the received code blocks are first decoded by reverse entropy coding and de-quantization. The decoded code blocks are put back into the subband domain for each subdescription, respectively. When a code block is lost, we reconstruct it from the code block at the same position in the other subdescription by one of the following methods: duplication, zero-padding, and interpolation. After all the lost code blocks are reconstructed, each subdescription is inverse wavelet transformed into sample-domain representations. Finally we de-interleave these two subdescriptions in order to get the reconstructed image.

In order for our architecture to work, we have to examine whether there are any correla-

tions between the two subdescriptions and whether reconstruction in the frequency domain lead to good quality. We first examine the correlations and the three reconstruction methods in subbands in Section 4.2. After verifying that frequency-based reconstruction methods are possible, we further derive an ORB-ST for frequency-based reconstruction methods in Section 4.3. Last, we propose a subband decomposed reconstruction method to better improve the reconstruction quality in Section 4.4. Note that the optimization of reconstruction quality can only be achieved when we optimize the reconstruction method and the subband transform together, which is computationally prohibitive. In our design, we first fix the reconstruction method and derive the ORB-ST, after which we further modify the reconstruction method to achieve better quality.

4.2 Correlation Analysis and Reconstruction in Subbands

In this section, we do correlation analysis and show that reconstruction can be done in each frequency band. We first define the correlation used in this chapter.

Definition 4.1: Let X and Y be two $M \times N$ 2-D signals. The terms $x(m,n)$ or $y(m,n)$ is the value of X or Y at position (m,n), respectively. The *correlation*, or the *correlation coefficient*, of X and Y, $Corr(X,Y)$ is defined as:

$$\begin{aligned} Corr(X,Y) &= \frac{E[(X-\bar{x})(Y-\bar{y})]}{(E[(X-\bar{x})^2])^{1/2}\ (E[(Y-\bar{y})^2])^{1/2}} \\ &= \frac{\sum_{m=1}^{M}\sum_{n=1}^{N}((x(m,n)-\bar{x})(y(m,n)-\bar{y}))}{(\sum_{m=1}^{M}\sum_{n=1}^{N}((x(m,n)-\bar{x})^2)^{1/2}\ (\sum_{m=1}^{M}\sum_{n=1}^{N}((y(m,n)-\bar{y})^2)^{1/2}}, \end{aligned}$$

where $\bar{x} = \frac{1}{M\times N}\sum_{m=1}^{M}\sum_{n=1}^{N}x(m,n)$, $\bar{y} = \frac{1}{M\times N}\sum_{m=1}^{M}\sum_{n=1}^{N}y(m,n)$

Reconstruction is possible only when there is enough correlation between the lost information and the received information. Since high correlations exist in the sample domain between subdescriptions, MDC has been traditionally carried out in the sample domain by interleaving image samples.

We now examine the correlations of frequency coefficients in the same subband between two subdescriptions. We illustrate this process in Figure 4.3, in which an example of calculating the correlation coefficient of subband $HH2$ is shown. After removing the DC component of an image X, pixels of the image are interleaved to generate two subdescriptions X_w^e and X_w^o, and are further transformed into the frequency domain. As correlation is usually measured in the sample

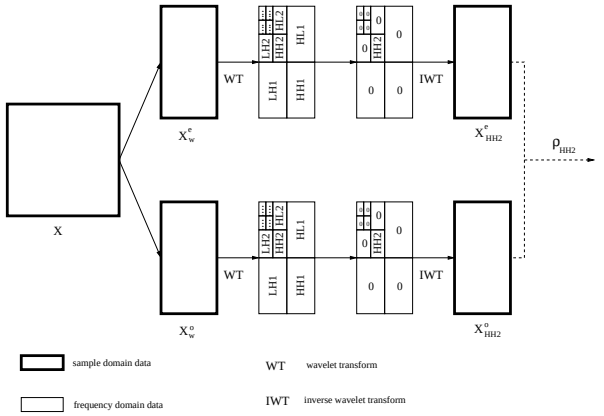


Figure 4.3 An example illustrating how the correlation coefficient in subband $HH2$, $\rho_{HH2} = Corr(X_{HH2}^e, X_{HH2}^o)$, is calculated.

domain, we transform frequency domain data in subbands $HH2$ of the two subdescriptions into the corresponding sample-domain representation X_{HH2}^e and X_{HH2}^o , setting the coefficients in other subbands to be zero. The correlation coefficient , $\rho_{HH2} = Corr(X_{HH2}^e, X_{HH2}^o)$ is then calculated between the sample domain data in the two subdescriptions.

Table 4.1 shows the correlation coefficient of each subband calculated in the sample domain for both a natural and a texture image. The results show that the correlations actually depend on the subbands considered. The correlations are usually lower in higher subbands, but higher in lower subbands. However, in higher subbands, although the correlations are low, the total energy is smaller as compared to that of lower subbands. As a result, errors in higher subbands will not affect the decoded image quality as much as those of lower subbands, so reconstruction in each frequency subband is feasible.

We also show the reconstruction results of three reconstruction methods in this table: padding by zeros, duplication, and interpolation. Padding by zeroes means that we zero the lost coefficients; duplication means that we copy the coefficients from the subdescription received to the subdescription lost; and interpolation means that we interpolate the inverse transformed samples instead of interpolating the subband coefficients. In interpolation, we assume a flat extension at the image

Table 4.1 Correlation coefficients between the inverse transformed images based on the coefficients in two corresponding subbands and zeros in other subbands of *lena* and *teeth*, and the error of three reconstruction methods applied in each subband. The error is represented in terms of the average noise energy per pixel (total noise energy divided by the number of image pixels).

Subband	Image *lena* in Group 1				Image *teeth* in Group 2			
	ρ	Avg. Distortion Per Pixel			ρ	Avg. Distortion Per Pixel		
		Dupl.	Padding	Intpl.		Dupl.	Padding	Intpl.
Unfiltered	0.972	64.60	1151.86	22.20	0.993	46.79	3609.21	18.36
LH1	0.369	31.44	25.07	20.54	0.367	19.75	15.43	13.28
HL1	0.370	2.12	1.68	1.96	0.528	3.16	3.39	2.97
HH1	0.119	3.44	1.96	2.08	0.174	4.30	2.61	2.67
LH2	0.820	12.76	42.83	14.64	0.827	8.77	25.52	9.37
HL2	0.851	1.06	2.97	1.11	0.901	1.57	7.97	1.59
HH2	0.674	2.43	3.73	2.41	0.728	2.37	4.37	2.36
LH3	0.954	6.50	72.78	8.43	0.956	3.55	40.92	4.67
HL3	0.954	0.41	4.50	0.46	0.984	0.61	19.34	0.65
HH3	0.914	1.07	6.28	1.38	0.935	1.32	10.06	1.65
LH4	0.992	2.41	160.93	1.65	0.989	1.29	61.83	1.77
HL4	0.981	0.21	5.79	0.23	0.996	0.27	38.59	0.31
HH4	0.977	0.47	10.80	0.63	0.984	0.59	18.79	0.81
LL4	0.999	2.07	811.08	4.01	0.999	0.87	3356.90	0.71

boundary.

Table 4.1 shows the result of applying each of the three methods in each subband, assuming the odd subdescription is lost. We see that the results for different methods in different subbands are very different, and that each subband has its own best strategy. For instance, padding by zeroes is certainly not good in lower subbands, as its performance is obviously poor, but it can be the best method in some higher subbands.

Although the results of interpolation are good in some subbands, it is not feasible in practice due to its high complexity. The decoding time for one inverse wavelet transform is of the order of 100 ms to 200 ms. In addition to the expensive computational cost, the interpolation operation cannot be decomposed into subbands, as we notice in the table that the total reconstruction errors in each subband do not sum up to the error in the unfiltered case for interpolation. However, both padding by zeroes and duplication can operate on the decomposed subband and are computationally inexpensive to carry out. As a result, we examine their properties in the next section.

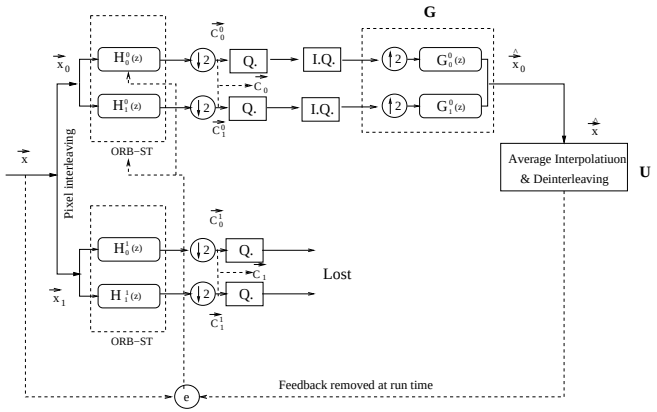


Figure 4.4 Architecture of previous ORB-ST approach.

4.3 Modified ORB-ST for frequency domain reconstruction

In this section, we first review the previous ORB-ST techniques derived for sample-domain reconstructions. Next, we derive a modified ORB-ST for two-way MDC with reconstructions by duplication in the frequency domain. Finally, we generalize this derivation to reconstructions by interpolations in the frequency domain and four-way MDC schemes.

4.3.1 Previous ORB technique

In order to improve the reconstruction quality of our proposed MDC scheme, we apply the ORB-ST technique discussed in the first chapter to our frequency reconstruction method and derive a new ORB-ST transform in this section.

Using a given reconstruction method, the basic principle of ORB-ST is to modify the transform in the encoder in order to minimize the distortions in the decoder by assuming that channel loss will occur during transmissions. Figure 4.4 shows the basic building blocks of the ORB-ST technique presented in [17]. In this figure, the procedures of interleaving, subband transform, down-sampling, quantization, and their reverse procedures are included. The quantization and de-quantization are assumed to be perfect and lossless because including the nonlinear quantization effect can

Table 4.2 A list of symbols used in Section 4.3.

Symbols	Definitions
i	Subdescription index 0 for the odd subdescription and 1 for the even subdescription
j	Subband index 0 for the low subband and 1 for the high subband
$\vec{x}$	Column vector of an image $\vec{x} = (x(0), x(1), ...x(n-1))^T$
$\vec{x}_0$	Odd subdescription of $\vec{x}$ $\vec{x}_0 = (x(1), x(3), ...x(n-1))^T$
$\vec{x}_1$	Even subdescription of $\vec{x}$ $\vec{x}_1 = (x(0), x(2), ...x(n-2))^T$
$\vec{c}_i^j$	Output from subband j of subdescription i at the encoder $\vec{c}_i^j = (c_i^j(0), c_i^j(1), ...c_i^j(n/4-1))^T$
$\vec{c}_{iL}$	Upsampled $\vec{c}_i^0$ $\vec{c}_{iL} = (c_i^0(0), 0, ...c_i^0(n/4-1), 0)^T$
$\vec{c}_{iH}$	Upsampled $\vec{c}_i^1$ $\vec{c}_{iH} = (0, c_i^1(0), ...0, c_i^1(n/4-1))^T$
$\vec{c}_i$	Combined vector of $\vec{c}_i^0$ and $\vec{c}_i^1$, also equals $\vec{c}_{iL} + \vec{c}_{iH}$ $\vec{c}_i = (c_i^0(0), c_i^1(0), c_i^0(1), c_i^1(1), ..., c_i^0(n/4-1), c_i^1(n/4-1))^T$
$\mathcal{E}_r$	Distortions between $\vec{x}$ and reconstructed output $\hat{\vec{x}}$
$\mathbf{G}$	Linear transform equivalent to upsampling and inverse wavelet transform
$\mathbf{H}_i$	ORB-ST matrix for subdescription i $\mathbf{H}_i = (\vec{h}_i(0), \vec{h}_i(1), ..., \vec{h}_i(n/2))^T$ $\vec{h}_i(k),\ k = 1...n/2$ is a row vector
$\mathbf{T}$	Combined ORB-ST matrix of $\mathbf{H}_0$ and $\mathbf{H}_1$ $\mathbf{T} = (\vec{h}_0(0), \vec{h}_1(0), \vec{h}_0(1), \vec{h}_1(1), ..., \vec{h}_0(n/2), \vec{h}_1(n/2))^T$
$\hat{\vec{c}}_i^j$	Decoded $\vec{c}_i^j$ at the decoder $\hat{\vec{c}}_i^j = (\hat{c}_i^j(0), \hat{c}_i^j(1), ...\hat{c}_i^j(n/4-1))^T$
$\hat{\vec{c}}_{iL}$	Upsampled $\hat{\vec{c}}_i^0$ $\hat{\vec{c}}_{iL} = (\hat{c}_i^0(0), 0, ...\hat{c}_{i)}^0(n/4-1), 0)^T$
$\hat{\vec{c}}_{iH}$	Upsampled $\hat{\vec{c}}_i^1$ $\hat{\vec{c}}_{iH} = (0, \hat{c}_i^1(0), ...0, \hat{c}_i^1(n/4-1))^T$
$\hat{\vec{c}}_i$	Combined vector from $\hat{\vec{c}}_i^0$ and $\hat{\vec{c}}_i^1$, also equals $\hat{\vec{c}}_{iL} + \hat{\vec{c}}_{iH}$ $\hat{\vec{c}}_i = (\hat{c}_i^0(0), \hat{c}_i^1(0), \hat{c}_i^0(1), \hat{c}_i^1(1), ..., \hat{c}_i^0(n/4-1), \hat{c}_i^1(n/4-1))^T$
$\vec{x}_{iL}, \vec{x}_{iH}$	$\vec{x}_{iL} = \mathbf{G}\vec{c}_{iL},\ \vec{x}_{iH} = \mathbf{G}\vec{c}_{iH}$
$\hat{\vec{x}}_{iL}, \hat{\vec{x}}_{iH}$	$\hat{\vec{x}}_{iL} = \mathbf{G}\hat{\vec{c}}_{iL},\ \hat{\vec{x}}_{iH} = \mathbf{G}\hat{\vec{c}}_{iH}$
$\hat{\vec{x}}_i$	Decoded output of subdescription i
$\hat{\vec{x}}$	Reconstructed output of $\vec{x}$

complicate the design of ORB-ST significantly. We also assume in this figure that subdescription 0 is received, while subdescription 1 is lost and reconstructed from subdescription 0.

We first define some symbols shown in this figure (Table 4.2). Let i be the subdescription index and j be the subband index. Each column of an image, represented by $\vec{x}$, is interleaved into two subdescription $\vec{x}_i,\ i = 0, 1$, which are wavelet transformed into low subband output $\vec{c}_i^0$ and high subband output $\vec{c}_i^1$, respectively. We introduce term $\vec{c}_i,\ i = 0, 1$, to denote a vector of components from $\vec{c}_i^0$ and $\vec{c}_i^1$. Matrix $\mathbf{G}$ is a linear transform that is equivalent to the process of up-sampling followed by filtering by $G_0^0(z)$ and $G_1^0(z)$ [17]:

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

where $g_i(j)$ is the jth coefficient of the filter $G_i^0(z)$. The decoded subdescription 0 is denoted as:

$$\hat{\vec{x}}_0 \ = \ \mathbf{G}\vec{c}_0 \tag{4.1}$$

Matrix $\mathbf{U}$ expresses the interpolation and de-interleaving operations [17]:

$$\mathbf{U} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 0.5 & 0.5 & 0 & 0 & 0 & 0 & \cdots & \\ & \vdots & & \vdots & \vdots & & \vdots & & \end{pmatrix}.$$

The reconstructed output is $\hat{\vec{x}} = \mathbf{U}\hat{\vec{x}}_0$.

The objective of ORB-ST is to find $\vec{c}_0$ in order to minimize $\mathcal{E}_r$, the reconstruction error between reconstructed output $\hat{\vec{x}}$ and input $\vec{x}$. In other words, our goal is to find out c_0 to minimize:

$$\mathcal{E}_r = ||\vec{z} - \vec{x}||^2 = ||\mathbf{UG}\vec{c}_0 - \vec{x}||^2.$$

As $\mathbf{UG}\vec{c}_0 = \vec{x}$ is an over-determined equation, the least square solution to it can be obtained by singular value decomposition (SVD). Let $\mathbf{P} = \mathbf{UG}$. We can use SVD to decompose $\mathbf{P}$ as $\mathbf{P} = \mathbf{SWD}^T$. Here $\mathbf{S}$ is an $n \times \frac{n}{2}$ column-orthogonal matrix; $\mathbf{W}$ is an $\frac{n}{2} \times \frac{n}{2}$ diagonal matrix with

positive or zero diagonal elements w_j; and $\mathbf{D}$ is an $\frac{n}{2} \times \frac{n}{2}$ orthogonal matrix. The solution can be expressed as $\vec{c}_0 = \mathbf{H}_0\vec{x} = \mathbf{D}\mathbf{W}^{-1}\mathbf{S}^T\vec{x}$. In the above formula, $\mathbf{W}^{-1}$ is a diagonal matrix whose elements are w_j^{-1}. If w_j is zero, we just set w_j^{-1} to zero in $\mathbf{W}^{-1}$.

The derivation of ORB-ST matrix $\mathbf{H}_1$ for $\vec{c}_1$, when subdescription 1 is received and subdescription 0 is lost, is similar. In that case, the interpolation matrix $\mathbf{U}$ is replaced by:

$$\mathbf{U} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0.5 & 0.5 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & \cdots \\ & \vdots & & \vdots & & \vdots & & \vdots & \end{pmatrix}.$$

When both subdescriptions are received, an inverse transform can be derived by inverting matrix $\mathbf{T}$ that is made up of de-interleaving row vectors of $\mathbf{H}_0$ and $\mathbf{H}_1$. (See Table 4.2.) In practice, this inverse transform is not perfect, as truncation and quantization errors are included.

We can see that the previous ORB-ST technique [17] is derived for a case in which reconstructions are done in the sample domain by interpolations. As our proposed scheme is based on frequency-domain reconstructions and uses duplication as the reconstruction method, the previous technique cannot be directly applied. For our new scheme, we have to re-formulate this problem and derive a new ORB-ST transform.

4.3.2 Modified ORB technique for frequency-based reconstructions

Before we re-formulate this problem, we first overview in Figure 4.5 the basic building blocks of our modified ORB-ST system. This structure contains three parts: the encoder, the UDP transmission over the Internet, and the decoder. The encoder has three stages: wavelet transform, down-sampling, and quantization (denoted as **Q.** in the figure). Here the quantization stage also includes entropy coding. The decoder has five stages: de-quantization (denoted as **I.Q.** in the figure), reconstruction, up-sampling, wavelet transform, and de-interleaving, in which de-quantization, up-sampling, and wavelet transform are the reverse procedures of those at the encoder. Reconstruction is the step in which the decoder reconstructs the lost code blocks; de-interleaving is the step in which the two subdescriptions are merged into a single image; and the quantization and de-quantization processes are assumed to be perfect and lossless.

We show an example to illustrate the reconstruction procedure before we give the derivations.

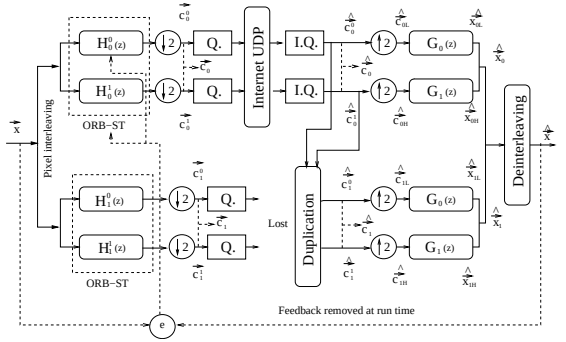


Figure 4.5 Architecture of our modified ORB scheme. The dotted boxes are the ORB matrices derived.

We assume that subdescription 1 is lost during transmission as shown in Figure 4.5, and that the lost data is reconstructed by duplication at the decoder. Note that we make this assumption only to demonstrate the reconstruction procedures. In the later derivation, we do not need this assumption.

In our derivation, we assume column interleaving, where $\vec{x}$ is the original column vector of an image. After $\vec{x}$ is interleaved, wavelet transformed, and down-sampled, we obtain $\vec{c}_i^j$, which is the original subband coefficients in subband j of subdescription i. We use $\vec{c}_i$ to denote a combined vector of $\vec{c}_i^0$ and $\vec{c}_i^1$ (See Table 4.2). These subband coefficients are then quantized, grouped into code blocks, and transmitted in UDP packets. Some of the UDP packets are received by the decoder, while others are lost during the transmission.

At the decoder side, the received code blocks of subdescription 0 are inverse quantized into the subband coefficients $\hat{\vec{c}}_0^j$. As the quantization and de-quantization are assumed to be lossless, we have $\hat{\vec{c}}_0^j = \vec{c}_0^j$. The lost code blocks of subdescription 1 are duplicated from the received code blocks, followed by an inverse quantization; hence, $\hat{\vec{c}}_1^j = \hat{\vec{c}}_0^j = \vec{c}_0^j$. As before, we use $\hat{\vec{c}}_i$ to denote a combined vector of $\hat{\vec{c}}_i^0$ and $\hat{\vec{c}}_i^1$ for each subdescription i.

After reconstruction, $\hat{\vec{c}}_i^0$ and $\hat{\vec{c}}_i^1$ are up-sampled, and the results are denoted as $\hat{\vec{c}}_{iL}$ and $\hat{\vec{c}}_{iH}$. From the definition of $\hat{\vec{c}}_i$ in Table 4.2, we can see that $\hat{\vec{c}}_{iL} + \hat{\vec{c}}_{iH} = \hat{\vec{c}}_i$. The terms $\hat{\vec{c}}_{iL}$ and $\hat{\vec{c}}_{iH}$ are inverse wavelet transformed to produce $\hat{\vec{x}}_{iL}$ and $\hat{\vec{x}}_{iH}$ (see Table 4.2 for their definitions). They

summed up to $\hat{\vec{x}}_i$, as expressed in (4.2), which is similar to (4.1).

$$\begin{aligned}\hat{\vec{x}}_i &= \hat{\vec{x}}_{iL} + \hat{\vec{x}}_{iH} \\ &= \mathbf{G}\hat{\vec{c}}_{iL} + \mathbf{G}\hat{\vec{c}}_{iH} \\ &= \mathbf{G}\hat{\vec{c}}_i. \end{aligned} \tag{4.2}$$

Note that an important interpretation for $\hat{\vec{x}}_{iL}$ is that it is actually the low frequency part of $\hat{\vec{x}}_i$ because it can be obtained by setting the high frequency coefficient $\hat{\vec{c}}_{iH}$ to zero in (4.2). Thus $\hat{\vec{x}}_{iL}$ and $\hat{\vec{x}}_{iL}$ are orthogonal to each other, and so are $\vec{x}_{iL}$ and $\vec{x}_{iL}$. Finally, $\hat{\vec{x}}_i s\ , i = 1, 2$ are de-interleaved to reproduce $\hat{\vec{x}}$, which is the reconstructed column vector.

We introduce some extra terms that can help in presenting our derivation: $\vec{c}_{iL}$, $\vec{c}_{iH}$, $\vec{x}_{iL}$, and $\vec{x}_{iH}$. They denote the value of $\hat{\vec{c}}_{iL}$, $\hat{\vec{c}}_{iH}$, $\hat{\vec{x}}_{iL}$, and $\hat{\vec{x}}_{iH}$, respectively, in a situation in which they are decoded perfectly with no loss. Their definitions can be found in Table 4.2.

We now state our assumptions on channel statistics before we give the derivation of the modified ORB-ST. Generally, we do not know the loss patterns ahead of time. Since we like to know whether channel statistics can help in better reconstruction quality using ORB, we derive our modified ORB-ST with known channel statistics. We use two parameters p and q to describe the channel situations. We assume that two packets are in an interleaved set of size 2, and that their loss patterns are described by $L = [a_1 a_2]$, where a_1 and a_2 represent the loss pattern of each packet respectively, with 1 being a lost packet and 0 to be a received one. We define the term $p = \Pr(L = [00])$ and $q = \Pr(L = [01]) + \Pr(L = [10])$; therefore $1 - p - q = \Pr(L = [11])$. If the exact p and q are known ahead of time, the results of our proposed ORB-ST can be considered as an upper bound.

We further make the following assumptions:

Assumption 4.4: We assume we can obtain previous p and q via receiver feedbacks.

Assumption 4.5: We further assume that, conditioned on the case including one packet is lost, this lost packet can be either a_1 or a_2 with the same probability. Hence, $\Pr(L = [01]) = \Pr(L = [10]) = q/2$.

Note that we do not have to assume independent packet losses in our derivation. When correct p and q are not available we may use the most recent fed-back p and q as the estimated parameters for the channel. In cases when channel feedbacks are not available, a conservative guess on losing half of the subdescriptions ($p = 0$ and $q = 1$) can be made.

Our objective is to find $\vec{c}_i$ in order to minimize $\mathcal{E}_r = ||\hat{\vec{x}} - \vec{x}||^2$, which we expand in (4.3). From this point on, we discuss the general situation and do not assume that subdescription 1 is lost.

Here $\mathbf{G}$ is the same matrix we have shown in the previous ORB-ST design.

$$\begin{aligned}\mathcal{E}_r &= \| \hat{\vec{x}} - \vec{x} \|^2 \\ &= \| \hat{\vec{x}}_0 - \vec{x}_0 \|^2 + \| \hat{\vec{x}}_1 - \vec{x}_1 \|^2 \\ &= \| (\hat{\vec{x}}_{0L} - \vec{x}_{0L}) + (\hat{\vec{x}}_{0H} - \vec{x}_{0H}) \|^2 + \| (\hat{\vec{x}}_{1L} - \vec{x}_{1L}) + (\hat{\vec{x}}_{1H} - \vec{x}_{1H}) \|^2 \\ &= \left(\| \hat{\vec{x}}_{0L} - \vec{x}_{0L} \|^2 + \| \hat{\vec{x}}_{0H} - \vec{x}_{0H} \|^2 \right) + \left(\| \hat{\vec{x}}_{1L} - \vec{x}_{1L} \|^2 + \| \hat{\vec{x}}_{1H} - \vec{x}_{1H} \|^2 \right) \\ &= \| \mathbf{G}\hat{\vec{c}}_{0L} - \vec{x}_{0L} \|^2 + \| \mathbf{G}\hat{\vec{c}}_{1L} - \vec{x}_{1L} \|^2 + \| \mathbf{G}\hat{\vec{c}}_{0H} - \vec{x}_{0H} \|^2 + \| \mathbf{G}\hat{\vec{c}}_{1H} - \vec{x}_{1H} \|^2 \end{aligned} \tag{4.3}$$

In (4.3), we first decompose the total distortions into distortions of two subdescriptions. The distortions of each subdescription is further decomposed into distortions in subbands. In the last line of (4.3), we can see that $\mathbf{E}_r$ depends on the decoded subband coefficients in each subdescription, namely, these four terms: $\hat{\vec{c}}_{iL}, i = 0, 1$ and $\hat{\vec{c}}_{iH}, i = 0, 1$. They have different forms depending on which subdescriptions are received. As we use duplication to be the reconstruction method, there are four possibilities in total:

- **Case 1**: Both subdescriptions are received: $\hat{\vec{c}}_{iL} = \vec{c}_{iL}$, $\hat{\vec{c}}_{iH} = \vec{c}_{iH}$ and $\mathcal{E}_{r1} = \| \mathbf{G}\vec{c}_{0L} - \vec{x}_{0L} \|^2 + \| \mathbf{G}\vec{c}_{1L} - \vec{x}_{1L} \|^2 + \| \mathbf{G}\vec{c}_{0H} - \vec{x}_{0H} \|^2 + \| \mathbf{G}\vec{c}_{1H} - \vec{x}_{1H} \|^2$
- **Case 2**: Subdescription 1 is lost and subdescription 0 is received: $\hat{\vec{c}}_{iL} = \vec{c}_{0L}$, $\hat{\vec{c}}_{iH} = \vec{c}_{0H}$ and $\mathcal{E}_{r2} = \| \mathbf{G}\vec{c}_{0L} - \vec{x}_{0L} \|^2 + \| \mathbf{G}\vec{c}_{0L} - \vec{x}_{1L} \|^2 + \| \mathbf{G}\vec{c}_{0H} - \vec{x}_{0H} \|^2 + \| \mathbf{G}\vec{c}_{0H} - \vec{x}_{1H} \|^2$
- **Case 3**: Subdescription 0 is lost and subdescription 1 is received: $\hat{\vec{c}}_{iL} = \vec{c}_{1L}$, $\hat{\vec{c}}_{iH} = \vec{c}_{1H}$ and $\mathcal{E}_{r3} = \| \mathbf{G}\vec{c}_{1L} - \vec{x}_{0L} \|^2 + \| \mathbf{G}\vec{c}_{1L} - \vec{x}_{1L} \|^2 + \| \mathbf{G}\vec{c}_{1H} - \vec{x}_{0H} \|^2 + \| \mathbf{G}\vec{c}_{1H} - \vec{x}_{1H} \|^2$
- **Case 4**: Both subdescriptions are lost: $\hat{\vec{c}}_{iL} = \hat{\vec{c}}_{iH} = 0$ and $\mathcal{E}_{r4} = \| \vec{x} \|^2$

What we minimize is the expectation of the distortion:

$$\begin{aligned} E[\mathcal{E}_r] &= p\mathcal{E}_{r1} + 0.5q\mathcal{E}_{r2} + 0.5q\mathcal{E}_{r3} + (1 - p - q)\mathcal{E}_{r4} \\ &= \big((p + 0.5q) \| \mathbf{G}\vec{c}_{0L} - \vec{x}_{0L} \|^2 + 0.5q \| G\vec{c}_{0L} - \vec{x}_{1L} \|^2\big) + \\ &\quad \big((p + 0.5q) \| \mathbf{G}\vec{c}_{0H} - \vec{x}_{0H} \|^2 + 0.5q \| \mathbf{G}\vec{0}_{H} - \vec{x}_{1H} \|^2\big) + \\ &\quad \big(0.5q \| \mathbf{G}\vec{c}_{1L} - \vec{\mathbf{x}}_{0L} \|^2 + (p + 0.5q) \| \mathbf{G}\vec{c}_{1L} - \vec{x}_{1L} \|^2\big) + \\ &\quad \big(0.5q \| \mathbf{G}\vec{c}_{1H} - \vec{x}_{0H} \|^2 + (p + 0.5q) \| \mathbf{G}\vec{c}_{1H} - \vec{x}_{1H} \|^2\big) + \\ &\quad ((1 - p - q)\mathcal{E}_{r3}) \\ &= A_1 + A_2 + A_3 + A_4 + A_5 \end{aligned} \tag{4.4}$$

The solution of (4.4) can be obtained by setting its derivative to zero and solving for $\vec{c}_{0L}, \vec{c}_{0H}, \vec{c}_{1L}$, and $\vec{c}_{1H}$ from the derivatives of A_1, A_2, A_3, and A_4, respectively (A_5 is a constant). We are going to demonstrate how $\vec{c}_{0L}$ is solved, and the rest of the derivations are the same. We first prove a lemma:

Lemma 4.1: Assume $\vec{z}$, $\vec{y}_1$ and $\vec{y}_2$ are n-dimensional vectors, and C_1 and C_2 are constants. The solution of the following:

$$\min_{\{\vec{z}\}} C_1||\vec{z}-\vec{y}_1||^2 + C_2||\vec{z}-\vec{y}_2||^2$$

is:

$$\vec{z} = \frac{C_1}{C_1+C_2}\vec{y}_1 + \frac{C_2}{C_1+C_2}\vec{y}_2$$

Proof of Lemma 4.1: Let $z(i)$, $y_1(i)$ and $y_2(i)$ be the i-th element of $\vec{z}$, $\vec{y}_1$ and $\vec{y}_2$, respectively. The objective can be rewritten as:

$$\min_{\{z(i)\}} \left(C_1 \sum_{i=1}^{n} (z(i)-y_1(i))^2 + C_2 \sum_{i=1}^{n} (z(i)-y_2(i))^2 \right).$$

By taking the derivative of the objective function of $z(i)$ for every i, we have:

$$2C_1(z(i)-y_1(i)) + 2C_2(z(i)-y_2(i)) = 0,$$

which gives $z(i) = \frac{C_1}{C_1+C_2}y_1(i) + \frac{C_2}{C_1+C_2}y_2(i)$. Therefore $\vec{z} = \frac{C_1}{C_1+C_2}\vec{y}_1 + \frac{C_2}{C_1+C_2}\vec{y}_2$.

To minimize A_1 in (4.4):

$$\begin{aligned} A_1 &= \left((p+0.5q) \parallel \mathbf{G}\vec{c}_{0L} - \vec{x}_{0L} \parallel^2 + 0.5q \parallel \mathbf{G}\vec{c}_{0L} - \vec{x}_{1L} \parallel^2\right) \\ &= \left((p+0.5q) \parallel \vec{x}^* - \vec{x}_{0L} \parallel^2 + 0.5q \parallel \vec{x}^* - \vec{x}_{1L} \parallel^2\right), \end{aligned} \tag{4.5}$$

where $\vec{x}^* = \mathbf{G}\vec{c}_{0L}$, and the solution is $\vec{x}^* = a\vec{x}_{0L} + b\vec{x}_{1L}$ and $a = \frac{2p+q}{2(p+q)}$ and $b = \frac{q}{2(p+q)}$. This equals $\vec{c}_{0L} = a\mathbf{G}^{-1}\vec{x}_{0L} + b\mathbf{G}^{-1}\vec{x}_{1L}$. Applying the same technique, we have:

$$c_{0L} = a\mathbf{G}^{-1}\vec{x}_{0L} + b\mathbf{G}^{-1}\vec{x}_{1L}, \quad c_{0H} = a\mathbf{G}^{-1}\vec{x}_{0H} + b\mathbf{G}^{-1}\vec{x}_{1H}$$

$$c_{1L} = b\mathbf{G}^{-1}\vec{x}_{0L} + a\mathbf{G}^{-1}\vec{x}_{1L}, \quad c_{1H} = b\mathbf{G}^{-1}\vec{x}_{0H} + a\mathbf{G}^{-1}\vec{x}_{1H}$$

The complete final solutions are given in (4.6) and (4.7):

$$\vec{\mathbf{c}}_0 = \vec{c}_{0L} + \vec{c}_{0H} = a\mathbf{G}^{-1}\vec{x}_0 + b\mathbf{G}^{-1}\vec{x}_1 \tag{4.6}$$

$$\vec{\mathbf{c}}_1 = \vec{c}_{1L} + \vec{c}_{1H} = b\mathbf{G}^{-1}\vec{x}_0 + a\mathbf{G}^{-1}\vec{x}_1 \tag{4.7}$$

Next, we show an example to illustrate the performance when different values of p and q are used. The experiment is done by transmitting a *lena* image, compressed at 0.125 bpp, from Urbana to Thailand in 8 UDP packets. The pattern of packet losses are [00001001], where 1 stands for a packet loss. These 8 packets can be grouped into 4 interleaving pairs: [00], [00], [10] and [01].

In the first case, we assume the unrealistic case that we know p and q obtained from the trace ahead of time: $p = 0.5$ and $q = 0.5$, hence we have $a = 0.75$ and $b = 0.25$.

In the second case, we assume we do not have any information of the channel, so we conservatively assume half of the subdescriptions are lost: $p = 0$ and $q = 1$; hence, we have $a = 0.5$ and $b = 0.5$.

In the third case, we test the case without the modified ORB-ST in order to obtain a lower bound for comparison. This case equals setting $p = 1$ and $q = 0$; thus, $a = 1$ and $b = 0$.

We substitute these values in (4.6) and (4.7) calculate the PSNR for each case. Here, PSNR is defined in (4.8), where x_i and $\hat{x}_i$ are, respectively, the original and the reconstructed pixel values:

$$PSNR = 10\log\frac{255^2}{\sum_i (x_i - \hat{x}_i)^2}. \tag{4.8}$$

Our experiments show that for the first case, the PSNR is 25.25 dB; for the second case, the PSNR is 25.23 dB; for the last case, the PSNR is 25.21 dB. The improvement in this example is not very large because we choose a low bitrate image, which has only a bandwidth of 8 packets and is easy to show its loss pattern. Due to the heavy quantization, the improvement here is not significant. The experimental results for high bitrate and non-quantized images are shown later.

4.3.3 Generalization of the modified ORB technique

In the above derivation, we have shown the frequency-domain ORB technique using duplication to reconstruct lost subdescriptions. It is also possible to derive a frequency domain ORB-ST using interpolation to reconstruct the lost subdescriptions. We generalize the objective function in (4.3) by rewriting it in (4.9):

$$\begin{aligned} \mathbf{E}_r &= \parallel \hat{\vec{x}} - \vec{x} \parallel^2 \\ &= \parallel \mathbf{RG}\hat{\vec{c}}_{0L} - \vec{\mathbf{x}}_{0L} \parallel^2 + \parallel \mathbf{RG}\hat{\vec{c}}_{1L} - \vec{x}_{1L} \parallel^2 + \\ &\quad \parallel \mathbf{RG}\hat{\vec{c}}_{0H} - \vec{x}_{0H} \parallel^2 + \parallel \mathbf{RG}\hat{\vec{c}}_{1H} - \vec{x}_{1H} \parallel^2, \end{aligned} \tag{4.9}$$

where R is the identity matrix I if the reconstruction is done by duplication, and is in the following form, when using interpolations for reconstruction:

$$R = \begin{pmatrix} 1/2 & 1/2 & \cdot & \cdot & \cdot & \cdot \\ 0 & 1/2 & 1/2 & \cdot & \cdot & \cdot \\ 0 & 0 & 1/2 & 1/2 & 0 & \cdot \\ 0 & 0 & 0 & 1/2 & 1/2 & \cdot \\ \vdots & \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix}$$

Note that R is different from $\mathbf{U}$ because R only accounts for interpolation, while U accounts for both interpolation and de-interleaving. Hence, R does not contain those rows in U that have only one nonzero element. The general solution to (4.9) can be represented as follows by substituting $\mathbf{G}$ by $\mathbf{GR}$ in (4.6) and (4.7):

$$\hat{\vec{c}}_0 = \mathbf{G}^{-1}\mathbf{R}^{-1}(a\vec{x}_0 + b\vec{x}_1); \quad \hat{\vec{c}}_1 = \mathbf{G}^{-1}\mathbf{R}^{-1}(b\vec{x}_0 + a\vec{x}_1).$$

($\mathbf{R}^{-1}$ exists because the determinant of $\mathbf{R}$ is the product of its diagonal elements, which is non-zero).

The ORB of two-way MDC can be used in the four-way MDC case by recursive doing row-after-column interleaving. We show the recursive approach in Figure 4.6. The image pixels are first interleaved in the column direction further interleaved in the row direction. Reconstructions are also done in a recursive way. We show an example in Figure 4.4. We assume that subdescriptions 1, 2, and 3 are lost. The first step of reconstruction is to reconstruct subdescription 1 from subdescription 0 by duplication and de-interleave their rows. The second step is to further duplicate the de-interleaved subdescriptions in order to reconstruct subdescription 2 and 3 and obtain the decoded image finally. By using this recursive approach, we can avoid deriving $2^4 = 16$ ORB-ST matrices for each loss pattern and we can reuse the matrices derived in the 2-way case.

4.4 Subband Decomposed Reconstruction

In this section, we propose a subband decomposed reconstruction and prove that it is better that reconstructions in the full frequency band. We also develop a method that can approximate this subband decomposed reconstruction and make it practical to be applied in the real situations.

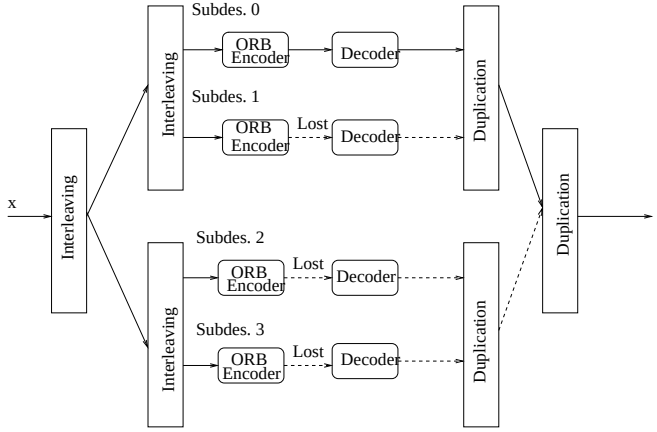


Figure 4.6 Four-way ORB-ST by recursive two-way ORB-ST.

4.4.1 Reconstruction in subbands

Reconstructions using either padding by zeroes or duplication can be viewed as a solution to a linear estimation problem. Given two general 2-D signals X and Y, with zero means, i.e. $\bar{x} = \bar{y} = 0$, a linear reconstruction problem is to find a linear transform of X, denoted as $\hat{Y} = aX + b$, in order to minimize the distortion e_0^2 defined as:

$$e_0^2 = E[(Y - \hat{Y})^2] = \frac{1}{M \times N} \sum_{m=1}^{M} \sum_{n=1}^{N} (y(m,n) - ax(m,n) - b)^2 \tag{4.10}$$

The solution to this problem can be obtained by setting the derivatives of e_0^2 with respect to a and b to be zero. That is:

$$a = \frac{E[XY]}{E[X^2]} = \frac{\sum_{m=1}^{M} \sum_{n=1}^{N} (x(m,n)y(m,n))}{\sum_{m=1}^{M} \sum_{n=1}^{N} (x(m,n)^2)}, \quad b = 0. \tag{4.11}$$

After substituting a and b into (4.10), we have:

$$\begin{aligned} e_0^2 &= E[(Y-aX)^2] \\ &= E[Y^2]+a^2E[X^2]-2aE[XY] \\ &= E[Y^2]+\frac{(E[XY])^2}{E[X^2]}-2\frac{(E[XY])^2}{E[X^2]} \\ &= E[Y^2]-\frac{(E[XY])^2}{E[X^2]} \end{aligned} \tag{4.12}$$

Note that the above optimal solution is obtained in the whole frequency band. In this subsection, we propose a scheme that does optimal reconstruction (by optimal, we mean with least errors in the linear estimation) in each subband separately. We prove that this approach has a smaller distortion than doing reconstruction in the whole frequency band.

Before we give our derivation, we first explain the symbols we use. The symbols X_w^o and X_w^e are the odd and even subdescriptions in the whole frequency band, respectively. The symbol j is the frequency band index, which consists of subband *orientation* and subband decomposition level. The *orientation* of a subband is where the subband is located in a subband decomposed structure. It can take one of the four values: LH, HL, LL, or HH. Subband decomposition level for a subdescription, denoted as l, can take the value from 1 to 4 by default. A special case of j is that when its value is w, it stands for the whole frequency band. The terms X_j^o and X_j^e is the odd and even subdescriptions in subband j. We use I to denote a set of all valid subband indices. Note that w is not in I, as I only contains indices for subbands. The term $s_j^2 = E[X_j^o X_j^e]$ is the cross-correlation between the odd and even subdescriptions in subband j, and $(\sigma_j^o)^2 = E[(X_j^o)^2]$ or $(\sigma_i^e)^2 = E[(X_j^e)^2]$ is the variance, or the energy, of the odd or even subdescriptions in subband j. The term ρ_j is the correlation between X_j^o and X_j^e, and r_j is the ratio of $(\sigma_j^o)^2$ to $(\sigma_j^e)^2$. We list these symbols in Table 4.3. Other symbols in this table will be further introduced when they are used.

In our derivation, we make the following assumptions:

Assumption 4.6: We assume that $E[X_{j_1}^o X_{j_2}^o] = E[X_{j_1}^e X_{j_2}^e] = E[X_{j_1}^o X_{j_2}^e] = 0,\ j_1 \neq j_2$. This assumption is valid because signals in different frequency bands are orthogonal.

Assumption 4.7: We assume that subdescriptions in the same subband has the same energy; i.e., $(\sigma_j^o)^2 = (\sigma_j^e)^2$. We test this hypothesis using a number of natural and texture images. In our test, we calculate the confidence intervals (CI) with 99% confidence of r_j and show them in Table 4.4. Later in this section, we use the term $(\sigma_j^*)^2$ to denote either $(\sigma_j^o)^2$ or $(\sigma_j^e)^2$ as they are equal.

Table 4.3 A list of symbols used in Section 4.4.

Symbols	**Definitions**
X, Y	Two general 2D signals
$X(m,n),\ Y(m,n)$	The value of X or Y at position (m,n)
$\bar{x},\ \bar{y}$	$\bar{x} = \frac{1}{M\times N}\sum_{m=1}^{M}\sum_{n=1}^{N} x(m,n),\ \bar{y} = \frac{1}{M\times N}\sum_{m=1}^{M}\sum_{n=1}^{N} y(m,n)$
$m,\ n$	Row and column coordinates
$M,\ N$	Row and column dimensions
e_0^2	Distortion when recontruction is done in the whole frequency band
e_1^2	Distortion when recontruction is done in each subband
l	decomposition level of a subband, $l \in \{1,2,3,4\}$ by default
j	Frequency band index, j can be LHl, HLl, HHl, $(l = 1,2,3,4)$ or $LL4$ j also can be w to indicate the whole frequency band,
	in that case, there is no suffix (LL,LH,HL, or HH) to it
I	Set of all subband indices $I = \{LH1, HL1, HH1.....\}$, note that $w \notin I$
X_j^o	Odd subdescription of which all subband coefficients are set to zero, except subband j; when $j = w$, it is the odd subdescription in the whole frequency band
X_j^e	Even subdescription of which all subband coefficients are set to zero, except subband j; when $j = w$, it is the even subdescription in the whole frequency band
s_j^2	$s_j^2 = E[X_j^o X_j^e] = \frac{1}{M\times N}\sum_{m=1}^{M}\sum_{n=1}^{N}(X_j^e(m,n)X_j^o(m,n))$
$(\sigma_j^e)^2$	$(\sigma_j^e)^2 = E[(X_j^e)^2] = \frac{1}{M\times N}\sum_{m=1}^{M}\sum_{n=1}^{N}(X_j^e(m,n)^2)$
$(\sigma_j^o)^2$	$(\sigma_j^o)^2 = E[(X_j^o)^2] = \frac{1}{M\times N}\sum_{m=1}^{M}\sum_{n=1}^{N}(X_j^o(m,n)^2)$
$(\sigma_j^*)^2$	$(\sigma_j^*)^2 = (\sigma_j^e)^2 = (\sigma_j^o)^2$
ρ_j	$\rho_j = Corr(X_j^o, X_j^e)$
r_j	$r_j = (\sigma_j^o)^2/(\sigma_j^e)^2$
d_j	$d_j = X_j^o - X_j^e$
d_j^2	$d_j^2 = E[(X_j^o - X_j^e)^2]$
b	$b = d_{LL4}^2$
c_1	$c_1 = d_{LH4}^2/d_{LL4}^2$
c_2	$c_2 = d_{HL4}^2/d_{LL4}^2$
c_3	$c_3 = d_{HH4}^2/d_{LL4}^2$

Table 4.4 Confidence intervals (CI) with 99% confidence of r_j for six natural and seven texture images.

Image	CI	Image	CI
barbara	1.05 ± 0.05	*cloth*	1.06 ± 0.01
boat	1.04 ± 0.02	*grape*	1.03 ± 0.01
goldhill	1.04 ± 0.01	*pines*	1.03 ± 0.01
lena	1.05 ± 0.03	*smoke*	1.03 ± 0.01
peppers	1.04 ± 0.01	*teeth*	1.03 ± 0.03
zelda	1.03 ± 0.01	*thumb*	1.08 ± 0.03
		trick	1.17 ± 0.04

Assumption 4.8: In our derivation, we assume that the even subdescription X_w^e is used to linearly reconstruct the lost odd subdescription X_w^o. The proof for the case using X_w^o to reconstruct X_w^e is exactly the same.

The previous linear reconstruction only use the correlation in the whole band. However, correlations differ very much in each individual subband. Thus, we propose to do linear reconstruction in each subband, using its individual correlations. We first prove two lemmas based on our assumptions.

Lemma 4.2:

$$\sum_{j\in I} s_j^2 = s_w^2, \quad \sum_{j\in I} (\sigma_j^e)^2 = (\sigma_w^e)^2, \quad \sum_{j\in I} (\sigma_j^o)^2 = (\sigma_w^o)^2.$$

Proof Since the proofs for the above three equations are very similar, we only show the proof for the first equation here.

$$\begin{aligned} s_w^2 &= E[X_w^e X_w^o] \\ &= E\left[\left(\sum_{j_1\in I} X_{j_1}^e\right)\left(\sum_{j_2\in I} X_{j_2}^o\right)\right] \\ &= E\left[\sum_{j_1=j_2} (X_{j_1}^e X_{j_2}^o)\right] + E\left[\sum_{j_1\neq j_2} (X_{j_1}^e X_{j_2}^o)\right] \\ &= \sum_{j_1=j_2} E[X_{j_1}^e X_{j_2}^o] + \sum_{j_1\neq j_2} E[X_{j_1}^e X_{j_2}^o] \\ &= \sum_{j\in I} s_j^2 + 0 \\ &= \sum_{j\in I} s_j^2 \end{aligned}$$

Lemma 4.3:

$$\rho_j = s_j^2/(\sigma_j^*)^2$$

.

Proof

$$\begin{aligned} \rho_j &= Corr(X_j^e, X_j^o) \\ &= \frac{E[X_j^e X_j^o]}{(E[(X_j^e)^2])^{1/2}(E[(X_j^o)^2])^{1/2}} \\ &= \frac{s_j^2}{((\sigma_j^e)^2)^{1/2}((\sigma_j^o)^2)^{1/2}} \\ &= \frac{s_j^2}{((\sigma_j^*)^2)^{1/2}((\sigma_j^*)^2)^{1/2}} \\ &= \frac{s_j^2}{(\sigma_j^*)^2} \end{aligned}$$

Lemma 4.4: The optimal reconstruction from X_j^e to X_j^o is:

$$\hat{X}_j^o = \rho_j X_j^e.$$

Proof From (4.11), we know the optimal reconstruction from X_j^e to X_j^o is:

$$\hat{X}_j^o = \frac{E[X_j^e X_j^o]}{E[(X_j^e)^2]} X_j^e + 0 = \frac{s_j^2}{(\sigma_j^e)^2} X_j^e = \frac{s_j^2}{(\sigma_j^*)^2} X_j^e.$$

From **Lemma 4.3**, this is exactly $\hat{X}_j^o = \rho_j X_j^e$.

Now let e_0^2 be the reconstruction error when we reconstruct in the whole frequency band. By using the results derived in (4.12), we have:

$$\begin{aligned} e_0^2 &= E[(X_w^o)^2] - \frac{(E[X_w^e X_w^o])^2}{E[(X_w^e)^2]} \\ &= (\sigma_w^*)^2 - s_w^4/(\sigma_w^*)^2. \end{aligned} \tag{4.13}$$

We further let e_1^2 be the sum of reconstruction errors in all subbands when we do reconstruction separately. We then have:

$$e_1^2 = \sum_{j\in I} \left((\sigma_j^*)^2 - s_j^4/(\sigma_j^*)^2\right). \tag{4.14}$$

Next, we prove that e_1^2 is smaller than e_0^2 by calculating the difference between e_0^2 and e_1^2:

$$\begin{aligned} e_0^2 - e_1^2 &= ((\sigma_w^*)^2 - s_w^4/(\sigma_w^*)^2) - \sum_{j\in I}\left((\sigma_j^*)^2 - s_j^4/(\sigma_j^*)^2\right) \\ &= \sum_{j\in I}\left(s_j^4/(\sigma_j^*)^2\right) - s_w^4/(\sigma_w^*)^2 + \left((\sigma_w^*)^2 - \sum_{j\in I}(\sigma_j^*)^2\right) \\ &= \left((\sigma_w^*)^2\sum_{j\in I}(s_j^4/(\sigma_j^*)^2) - s_w^4\right)/(\sigma_w^*)^2 + 0 \\ &= \left(\sum_{j\in I}((\sigma_j^*)^2)\sum_{j\in I}(s_j^4/(\sigma_j^*)^2) - s_w^4\right)/(\sigma_w^*)^2 \\ &\geq \left((\sum_{j\in I}s_j^2)^2 - s_w^4\right)/(\sigma_w^*)^2 = 0. \end{aligned} \tag{4.15}$$

The inequality in (4.15) is derived based on the Cauchy-Schwartz inequality:

$$\left(\sum_{i=1}^N p_i^2\right)\left(\sum_{i=1}^N q_i^2\right) \geq \left(\sum_{i=1}^N p_i q_i\right)^2,$$

where equality holds when p_i/q_i is constant for all i. Hence in (4.15), equality holds when $s_j^2/(\sigma_j^*)^2 = \rho_j$ is a constant. This means that reconstructing in decomposed subbands will improve over reconstructing in the whole frequency band unless all ρ_i are the same across all subbands. As we already show that ρ_i differs in different subbands, we conclude, that between two subdescriptions, reconstructing across the same subbands is better than reconstructing in the whole frequency band.

The limitation of this method is that we require knowing ρ_j for each subband j, which is image-dependent and require a large overhead to send them to the receiver. We solve this problem in the next subsection by proposing an approximate subband decomposed reconstruction.

4.4.2 Approximating correlations in subbands

In order to make our method work, we still have to calculate ρ_j for each subband j at the sender and transmit them to the receiver. Since this will incur a lot of overhead, we derive a model that can produce ρ_j approximately with a few parameters, thereby allowing us to duplicate these parameters in each subdescription. We notice that calculating ρ_j from its definition can be difficult because it requires knowing the image-dependent s_j^2. Instead of doing this, we begin by rewriting

the definition of ρ_j in (4.16):

$$\begin{aligned} \rho_j &= Corr(X_j^o, X_i^e) \\ &= \frac{E[X_j^o X_i^e]}{E[(X_j^o)^2]^{1/2}E[(X_j^e)^2]^{1/2}} \\ &= \frac{E[(X_j^e)^2] + E[(X_j^o)^2] - E[(X_j^o - X_j^e)^2]}{2(\sigma_j^*)^2} \\ &= 1 - \frac{d_j^2}{2(\sigma_j^*)^2}. \end{aligned} \tag{4.16}$$

Here d_j is defined as $d_j = X_j^o - X_j^e$; in particular $d_w = X_w^o - X_w^e$ is called the *difference signal* of the two subdescriptions. As d_w is the difference between two subdescriptions, it carries very little information about the image content and the reconstruction of X_w^o or X_w^o. This is proved as follows. From (4.11), we know that the optimal reconstruction from d_w to X_w^o can be obtained by setting the term a as:

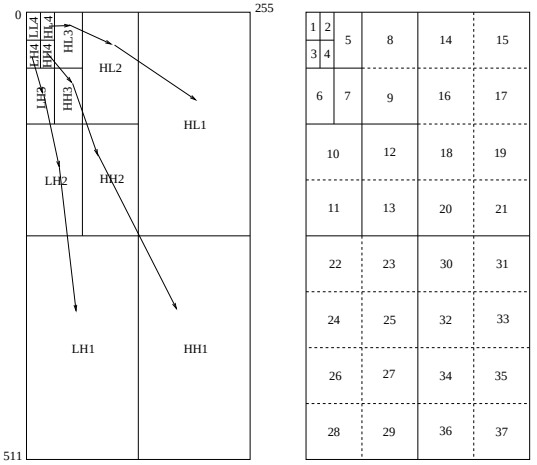
$$\begin{aligned} a &= \frac{E[d_w X_w^o]}{E[(X_w^o)^2]} \\ &= \frac{E[(X_w^o - X_w^e)X_w^o]}{E[(X_w^o)^2]} \\ &= \frac{E[(X_w^o)^2 - X_w^o X_w^e]}{E[(X_w^o)^2]} \\ &= \frac{E[(X_w^o)^2] - E[X_w^o X_w^e]}{E[(X_w^o)^2]} \\ &= \frac{E[(X_w^o)^2] - \rho_w E[(X_w^o)^2]}{E[(X_w^o)^2]} \\ &= 1 - \rho_w \\ &= 1 - Corr(X_w^o, X_w^e). \end{aligned} \tag{4.17}$$

As we know that X_w^e and X_w^o are highly correlated, a in (4.17) is very close to 0; thus, the difference signal d_w has almost no information about X_w^o. For the same reason, it has no information about X_w^e as well.

The above conclusion that the difference signal d_w has no information about the image content motivates us to assume that it is like a random noise, of which the spectrum is white; i.e., each frequency coefficients in the frequency domain has the same value. This is equivalent to assuming that the energy of d_w in a subband j, denoted as d_j^2, is proportional to the number of frequency coefficients it contains. In JPEG 2000, the number of frequency coefficients in a subband is 4 times

Table 4.5 Results of k_i found by linear regression on each image, each group of images, and all the images together. In each case, the R^2 measure of linear regression is shown.

Image	HL Band		LH Band		HH Band		Image	HL Band		LH Band		HH Band	
	k_1	R_1^2	k_2	R_2^2	k_3	R_3^2		k_1	R_1^2	k_2	R_2^2	k_3	R_3^2
barbara	3.76	0.97	3.86	0.98	4.61	0.85	*cloth*	4.13	0.80	1.31	0.52	0.97	0.23
boat	3.95	0.99	2.28	0.99	2.54	0.88	*grape*	2.49	0.98	1.97	0.99	1.56	0.80
goldhill	2.88	0.99	2.80	0.99	2.70	0.99	*pines*	3.95	0.97	2.46	0.95	2.50	0.90
lena	2.36	0.99	2.16	0.99	2.04	0.99	*smoke*	2.54	0.99	2.31	0.99	1.97	0.99
peppers	1.82	0.98	2.57	0.96	2.08	0.99	*teeth*	3.79	0.95	2.53	0.87	2.03	0.54
zelda	2.07	0.98	2.27	0.98	1.92	0.82	*thumb*	3.59	0.68	1.75	0.19	1.40	0.02
Group 1	2.69	0.79	2.60	0.80	2.53	0.68	*trick*	3.79	0.99	1.42	0.86	2.05	0.96
Groups 1 & 2	3.05	0.80	2.20	0.68	2.05	0.40	**Group 2**	3.40	0.82	1.91	0.55	1.71	0.22



(a) Subband decomposed structure

(b) Code block organization

Figure 4.7 The relationship of d_j^2 in the frequency domain. The left figure shows a four-level decomposed frequency-based structure of a 512×255 image. The three arrows show the grouping of the different subbands. The right figure shows the structure when subbands are further divided into code blocks.

that of the subband of the same orientation in a higher level. Hence, d_j^2 are in the following form:

$$d_{LHl}^2 = c_1 b(4)^{(4-l)}; \quad d_{HLl}^2 = c_2 b(4)^{(4-l)}; \quad d_{HHl}^2 = c_3 b(4)^{(4-l)},$$

where $b = d_{LL4}^2$ and c_1, c_2, and c_3 are energy of d_j in subband $LH4$, $HL4$, and $HH4$, normalized by b, respectively. This relationship is illustrated in Figure 4.7.

For real images, we can expect that the ratio of the energy of a subband to that of the subband at a higher level will not exactly be 4; hence, we modify the above assumption for real images.

Assumption 4.9:

$$d_{LHl}^2 = c_1 b k_1^{(4-l)}; \quad d_{HLl}^2 = c_2 b k_2^{(4-l)}; \quad d_{HHl}^2 = c_3 b k_3^{(4-l)}. \tag{4.18}$$

As subbands of different orientations carry different details of the original signal, we do not assume that there is any relationship across them. In other words, k_i are independent of each other.

To verify this assumption and to calculate k_i for real images, we select six natural and seven textural images. We first obtain d_j^2 for an image, transform them into the log domain, and use linear regression to extract k_i from them (Table 4.5). We also show the commonly used R^2 measure [21] in linear regression in the table, which is close to 1 if two sets of data are linearly correlated. We can see that this assumption holds for most single images, as their R^2 are very close to 1. Moreover, the values of k_i do not vary significantly across images. We also calculate k_i for all natural (shown as **Group 1** in the figure) and textural images (shown as **Group 2**), respectively. Finally, we calculate k_i of the entire thirteen images (**Group 1 & 2**). These k_i parameters for groups of images can be kept as *a priori* information at each receiver and used when reconstruction is necessary.

Now we show how to obtain ρ_j by these k_i. In order to calculate ρ_j, we need d_j^2 and $(\sigma_j^*)^2$ according to (4.16). The term $(\sigma_j^*)^2$ can be obtained from subband j of the received subdescription, and d_j^2 can be obtained using (4.19), which means the energy of a signal in each subband sums up to its total energy in the entire band:

$$\begin{aligned} d_w^2 &= d_{LL4}^2 + \sum_{l=1}^{4} d_{LHl}^2 + \sum_{l=1}^{4} d_{HLl}^2 + \sum_{l=1}^{4} d_{HHl}^2 \\ &= b + \sum_{i=1}^{3} c_i b(1 + k_i + k_i^2 + k_i^3). \end{aligned} \tag{4.19}$$

By using the four initial values c_1, c_2, c_3 and d_w^2 transmitted from the sender, together with the k_i parameters (for certain groups of images) kept at the receiver beforehand, the receiver can determine b first and then obtain d_j^2 from (4.18).

Table 4.6 Performance gain in decibels of our four proposed reconstruction methods as compared to duplication, both under no quantization loss and with quantization. G_u is the gain when unified k_i's are used across all images; G_t is the gain when the k_i from a specific group is used; G_i is the gain when k_i from a single image is used; and G_0 is the gain when the actual ρ_i is used.

Image	No Quantization				With Quantization											
	8bpp				0.5 bpp				0.25 bpp				0.125 bpp			
	G_u	G_t	G_i	G_0	G_u	G_t	G_i	G_0	G_u	G_t	G_i	G_0	G_u	G_t	G_i	G_0
barbara	2.28	2.39	2.57	2.81	2.02	2.02	2.15	2.35	1.56	1.61	1.70	1.78	0.77	0.78	0.78	0.84
boat	1.44	1.32	1.42	1.46	1.09	1.00	1.08	1.11	0.65	0.67	0.62	0.68	0.28	0.28	0.28	0.29
goldhill	1.02	1.14	1.13	1.15	0.50	0.50	0.49	0.59	0.28	0.27	0.27	0.32	0.17	0.17	0.17	0.18
lena	0.90	0.98	0.98	0.98	0.64	0.69	0.70	0.70	0.37	0.42	0.44	0.46	0.16	0.17	0.17	0.18
peppers	0.86	0.96	1.09	1.15	0.64	0.64	0.66	0.76	0.45	0.44	0.39	0.52	0.25	0.25	0.25	0.28
zelda	1.13	1.25	1.25	1.28	0.70	0.76	0.77	0.78	0.45	0.49	0.53	0.55	0.20	0.20	0.20	0.22
cloth	0.26	0.21	0.09	0.49	0.17	0.08	0.00	0.38	0.06	0.06	0.06	0.12	0.06	0.06	0.06	0.11
grape	0.68	0.60	0.73	0.75	0.41	0.36	0.45	0.52	0.19	0.14	0.25	0.32	0.06	0.06	0.07	0.08
pines	1.06	1.05	1.05	1.08	0.48	0.42	0.41	0.50	0.23	0.23	0.23	0.27	0.07	0.07	0.07	0.07
smoke	0.92	0.71	1.02	1.02	0.19	0.19	0.19	0.26	0.06	0.06	0.06	0.06	0.03	0.03	0.03	0.03
teeth	0.73	0.69	0.67	0.77	0.48	0.44	0.43	0.51	0.14	0.14	0.13	0.22	0.04	0.04	0.04	0.04
thumb	0.05	-0.03	-0.03	0.49	0.08	0.11	0.11	0.39	0.14	0.16	0.16	0.25	0.05	0.06	0.06	0.10
trick	1.48	1.61	1.63	1.65	0.79	0.76	0.76	0.83	0.22	0.22	0.22	0.23	0.06	0.06	0.07	0.08

Table 4.6 shows how accurately this model can work. We send the four parameters of each image (c_1, c_2, c_3 and d_w^2) to a sender and assume that one subdescription is totally lost. The receiver keeps several groups of k_i values, one for each type of images. If the sender does not specify the image type, the group of unified k_i values will be used; otherwise, k_i from the specified group will be used. For comparison, we also include the case the exact k_i from each image is used and the case when the actual ρ_j are available. Cases with no quantization and quantization at three bit rates are included. We see from the table that our approximation will only degrade the performance by less than 1 dB.

In our implementation, c_1, c_2, c_3 and d_0^2 are converted to integers before transmission in order to save bandwidth: $N_1 = \lfloor c_1/c_2 + 0.5 \rfloor$, $N_2 = \lfloor 1/c_2 + 0.5 \rfloor$, $N_3 = \lfloor c_3/c_2 + 0.5 \rfloor$, and $N_e = \lfloor d_0^2 + 0.5 \rfloor$. We normalize c_1 and c_3 by c_2 because c_2 is always the smallest value of the three. As c_2 itself is always smaller than 1, we convert c_2 be taking its reciprocal. Table 4.7 shows the four transmitted parameters for each of the thirteen images. We see that N_1 and N_2 can be encoded in 4 bits, N_3 in 5 bits and N_e in 10 bits. Therefore, the additional overhead in bandwidth is small.

Table 4.7 Transmitted parameters of each of the thirteen images.

Natural Images	Parameters				Textural Images	Parameters			
	N_1	N_2	N_3	N_e		N_1	N_2	N_3	N_e
barbara	13	37	20	311	*cloth*	3	4	12	212
boat	3	2	8	102	*grape*	5	3	8	67
goldhill	5	2	9	68	*pines*	4	2	5	106
lena	10	2	11	65	*smoke*	3	2	5	47
peppers	15	2	16	79	*teeth*	3	3	7	127
zelda	7	2	11	28	*thumb*	15	3	14	106
					trick	1	1	2	1734

4.5 Experimental Results

In this section, we first discuss several alternatives of MDC schemes. Next, we show the experimental results under both controlled-loss scenarios and trace-driven Internet simulations. Finally, we show the time-quality trade-offs of our proposed method and compare it with the traditional schemes.

Before we show our experimental results, we first describe our experiment settings. The images we use are *lena*, *barbara*, *goldhill*, *peppers*, *smoke* and *cloth*. The last two images are textural images. We calculate PSNR defined in (4.8) as our evaluation metric.

4.5.1 Comparison of Alternatives

In our experiments, we compare several alternatives of MDC schemes. As pointed out both in Chapter 1 and in Assumptions 4.2 and 4.3, MDC schemes have four different attributes, which form a total of 24 combinations. The first attribute is the availability of channel statistics. We use STAT to represent the case when exact channel loss patterns is known ahead of time. Notice that this is not realizable in practice, and we include it as an upper bound to performance. We use NOSTAT to represent the case when no loss patterns are available, and LTSTAT to represent the case when long term statistical information is available via receiver feedbacks. The second attribute is the reconstruction method. We use DUPL and INTPL to represent the cases when duplication and interpolation operations are used to reconstruct lost subdescriptions, respectively. The third attribute is the domain where reconstruction is done. We use TIME and FREQ to represent the cases that reconstruction is done in the sample and the frequency domains, respectively. Finally, we use ORB and NOORB to represent the cases on whether ORB is used. Here, we do not explicitly specify the reconstruction method that ORB is based on but it is implicit that it is based on the

same reconstruction method specified in the experiment. Moreover, when ORB technique is applied with the case of NOSTAT, the encoder always assumes that half of the subdescriptions are lost in deriving the ORB transforms. When ORB is applied with the case of STAT or LTSTAT, the encoder will adjust the ORB transforms according to the actual or estimated channel statistics.

Although there is a total of 24 cases, we eliminate some cases according the following three observations:

1: When reconstruction is done in the sample domain, interpolations will be better than duplications in the reconstruction quality. Hence, cases with TIME and DUPL together are eliminated.

2: Channel statistics are only helpful when they are used with ORB. Hence, cases with NOORB and STAT, or, NOORB and LTSTAT, are eliminated.

3: As interpolations are very computationally expensive when applied in the frequency domain, we only include two cases (FREQ, INTPL, ORB, STAT) and (FREQ, INTPL, NOORB, NOSTAT) for comparison. Other cases with INTPL and FREQ together are eliminated.

Table 4.8 shows the number of cases eliminated by each observation, as well as some examples of eliminated cases. In total, 17 cases are eliminated, leaving seven cases to be evaluated:

- Case I (TIME, INTPL, NOORB, NOSTAT)
- Case II (TIME, INTPL, ORB, NOSTAT)
- Case III (FREQ, DUPL, NOORB, NOSTAT)
- Case IV (FREQ, INTPL, NOORB, NOSTAT)
- Case V (FREQ, DUPL, ORB, LTSTAT)
- Case VI (FREQ, DUPL, ORB, STAT)
- Case VII (FREQ, INTPL, ORB, STAT)

In later subsections, we test the performance of each of these cases under both controlled-loss scenarios and trace-driven Internet experiments. In the seven cases, Cases I and II are previously proposed MDC schemes; Cases III to VI are our proposed frequency-based schemes; and Case VII is included as an upper bound of performance.

4.5.2 Reconstruction quality under controlled losses

In this subsection, we present our experimental results based on controlled loss scenarios. The first scenario is that both subdescriptions are received. The second and third are based on either the even or the odd subdescription is lost. We do not test Case V because in controlled loss scenarios, the exact loss pattern is known, and long term statistics (LTSTAT) is implied. Each of the other

Table 4.8 Three observations to eliminate impractical cases. In each observation, the number of cases eliminated and two example cases eliminated are shown.

Observations	# of cases eliminated	Two examples of cases eliminated
1	7	(TIME, DUPL, ORB, NOSTAT), (TIME, DUPL, ORB, LTSTAT)
2	8	(FREQ, DUPL, NOORB, STAT), (FREQ, DUPL, NOORB, LTSTAT)
3	2	(FREQ, INTPL, ORB, LTSTAT), (FREQ, INTPL, ORB, NOSTAT)

six cases is carried out at no quantization, as well as at three different bit-rates. Both two-way and four-way MDC are tested.

Table 4.9 shows the results of two-way MDC. The first four rows show the results under no quantization, and consequently, no quantization. Hence, Cases I and IV will give the same result, as well as Cases II and VII, because they have the same reconstruction methods. Case III is not as good as Case I or II because duplication is usually not as good as interpolation in terms of reconstruction quality under no segmentation loss. When both subdescriptions are received, perfect reconstruction is achieved in all cases.

The remaining rows in Table 4.9 show the performance when quantization is included. We use three bit-rates: 0.5 bpp, 0.25 bpp and 0.125 bpp, respectively. When one of the subdescriptions is lost, all frequency-domain reconstruction methods, namely, Cases III, IV, VI, and VII, outperform sample-domain reconstruction methods in Cases I and II. Among the frequency-based methods, Case VII gives the best performance, as it uses interpolation to reconstruct lost subdescriptions together with ORB. To further show how Cases III and IV are compared to the upper bound (Case VII), we have shown the degradations of both cases from Case VII. Note here when both subdescriptions are received, they have the same quality for Cases III, IV and VII so there are no degradations under such situations.

We observe that the degradations from Case III to Case VII are mostly between 1 to 2 dB, and the degradations from Case IV to Case VII are mostly from 0.5 to 1.5 dB, depending on the bit-rates and image. The difference between these two degradations is usually around 0.5 dB, which is due to the fact that interpolations have better reconstruction ability than duplications. An exception to the above conclusion is *cloth*, which shows large degradations in Cases III and IV, as compared to the upper bound. However, this image benefits a lot from the ORB technique, as the performance of Case VI is high. The reason is that for certain textural images like *cloth*, very periodic patterns

Table 4.9 Reconstruction quality in PSNR (dB) when transformed by frequency-based algorithm along the horizontal direction and one or two of the subdescriptions received under two-way MDC. For Cases III and IV, we have shown their degradations from Case VII, to better reflect their performances compared with Case VII.

Image	Quant. Effects	bit rate	Odd Received								Even Received								Both Received					
			I	II	III	Degr.(III)	IV	Degr.(IV)	VI	VII	I	II	III	Degr.(III)	IV	Degr.(IV)	VI	VII	I	II	III	IV	VI	VII
barbara	No	-	25.21	26.21	23.19	-3.02	25.21	-1.00	26.19	26.21	25.26	26.27	23.19	-3.08	25.26	-1.01	26.19	26.27	perfect reconstruction					
goldhill		-	32.77	34.11	29.83	-4.28	32.77	-1.34	32.83	34.11	32.73	34.05	29.83	-5.22	32.73	-1.33	32.83	34.05	perfect reconstruction					
pepper		-	31.65	33.65	29.14	-4.51	31.65	-2.00	32.14	33.65	33.66	35.16	29.14	-6.02	33.66	-1.50	32.14	35.16	perfect reconstruction					
lena		-	34.81	35.92	30.03	-5.89	34.81	-1.11	33.03	35.92	34.69	35.82	30.03	-5.79	34.69	-1.13	33.03	35.82	perfect reconstruction					
smoke		-	37.75	38.74	33.24	-5.50	37.75	-0.99	33.24	38.74	37.72	38.00	33.04	-4.60	37.72	-0.28	33.04	38.00	perfect reconstruction					
cloth		-	37.98	38.01	33.55	-4.46	37.98	-0.03	33.03	35.92	37.56	37.67	33.23	-4.44	37.69	-0.33	33.03	35.82	perfect reconstruction					
barbara	Yes	0.125	21.73	21.91	22.80	-0.12	22.81	-0.13	22.88	22.93	21.71	21.84	22.79	-0.17	22.84	-0.14	22.88	22.96	21.90	21.89	23.43	23.43	23.43	23.43
		0.25	22.48	22.82	23.20	-0.91	23.58	-0.53	23.79	24.11	22.44	22.75	23.20	-1.03	23.51	-0.71	23.79	24.22	23.02	23.06	26.20	26.20	26.20	26.20
		0.5	23.16	23.75	24.70	-0.51	24.71	-0.50	25.04	25.21	23.19	23.84	24.73	-0.53	24.75	-0.51	25.04	25.26	25.84	24.67	29.84	29.84	29.84	29.84
goldhill	Yes	0.125	25.02	25.12	25.78	-0.53	26.00	-0.25	26.21	26.25	25.07	25.11	25.91	-0.40	25.99	-0.32	26.21	26.31	25.09	25.01	26.46	26.46	26.46	26.46
		0.25	26.23	26.25	27.19	-1.05	27.47	-0.77	28.01	28.24	26.22	26.29	27.21	-0.97	27.50	-0.68	28.01	28.18	26.45	26.30	28.74	28.74	28.74	28.74
		0.5	27.58	27.74	28.29	-1.16	29.33	-0.70	29.91	30.03	27.62	27.78	28.27	-1.15	29.41	-0.71	29.91	30.02	28.16	27.91	31.26	31.26	31.26	31.26
peppers	Yes	0.125	24.12	24.37	26.18	-0.69	26.07	-0.80	26.34	26.87	24.27	24.28	25.91	-1.03	26.01	-0.93	26.34	26.94	24.35	24.35	27.33	27.33	27.33	27.33
		0.25	26.19	26.37	27.69	-1.80	28.47	-1.02	28.60	29.49	26.46	26.24	27.62	-2.33	28.50	-1.45	28.60	29.95	26.76	26.49	30.68	30.68	30.68	30.68
		0.5	28.31	28.67	29.61	-1.91	30.10	-1.42	30.22	31.52	29.03	29.09	28.58	-3.76	30.11	-1.23	30.22	32.34	29.68	29.21	34.03	34.03	34.03	34.03
lena	Yes	0.125	25.42	25.42	26.70	-0.57	26.98	-0.29	27.13	27.27	25.47	25.43	26.73	-0.64	26.99	-0.38	27.13	27.37	25.48	25.48	27.77	27.77	27.77	27.77
		0.25	27.22	27.23	28.26	-1.90	29.01	-1.15	29.73	30.16	27.25	27.27	28.23	-1.98	29.02	-1.19	29.73	30.21	27.35	27.28	31.04	31.04	31.04	31.04
		0.5	29.00	29.06	29.33	-3.73	32.49	-0.57	32.59	33.06	28.90	29.07	29.33	-3.64	32.50	-0.47	32.59	32.97	29.40	29.13	34.93	34.93	34.93	34.93
smoke	Yes	0.125	24.50	24.58	25.48	-0.79	25.68	-0.57	25.93	26.27	24.51	24.54	25.47	-0.83	25.69	-0.61	25.97	26.30	24.70	24.74	26.66	26.66	26.66	26.66
		0.25	26.11	26.32	26.47	-1.74	27.21	-1.00	27.55	28.21	26.12	26.27	26.45	-1.94	27.32	-1.07	27.74	28.39	26.22	26.35	28.79	28.79	28.79	28.79
		0.5	27.93	28.06	29.33	-1.65	29.97	-1.01	30.59	30.98	27.91	28.07	29.31	-1.66	29.98	-0.99	30.59	30.97	28.10	28.33	31.16	31.16	31.16	31.16
cloth	Yes	0.125	17.84	18.42	19.99	-3.18	21.98	-1.19	22.43	23.17	17.84	18.43	20.03	-3.24	21.99	-1.28	22.33	23.27	17.96	18.48	23.93	23.93	23.93	23.93
		0.25	20.30	21.23	23.25	-3.89	24.41	-1.75	26.53	27.16	20.25	21.27	23.15	-4.06	24.32	-2.89	26.73	27.21	20.37	22.28	28.28	28.28	28.28	28.28
		0.5	22.86	23.06	26.72	-5.74	28.69	-3.87	30.59	32.56	22.87	23.07	27.71	-4.76	28.50	-3.97	30.19	32.47	22.98	23.13	33.44	33.44	33.44	33.44

Table 4.10 Reconstruction quality in PSNR (dB) when each image is divided into four subdescriptions by recursive two-way interleaving under lossy scenarios. For Cases III and IV, we have shown their degradations from Case VII, to better reflect their performances compared with Case VII.

(a) Case I to III

Image	Bit Rate	Subdes. 1Recvd.				Subdes. 2Recvd.				Subdes. 3Recvd.				Subdes. 4Recvd.				Subdes. 1-4Recvd.		
		I	II	III	Degr.(III)	I	II	III	Degr.(III)	I	II	III	Degr.(III)	I	II	III	Degr.(III)	I	II	III
barbara	-	25.14	25.89	22.21	-3.68	25.09	25.83	22.59	-3.24	25.13	25.87	22.60	-3.27	25.09	25.83	22.21	-3.62	perfect reconstruction		
goldhill	-	30.69	31.83	27.31	-4.52	30.70	31.87	27.41	-4.46	30.65	31.80	27.41	-4.39	30.66	31.84	27.31	-4.53	perfect reconstruction		
peppers	-	31.69	33.03	26.93	-6.10	30.33	32.05	27.01	-5.04	30.86	32.44	27.03	-5.41	29.68	31.56	26.91	-4.65	perfect reconstruction		
lena	-	33.44	34.55	28.29	-6.26	33.43	34.54	28.81	-5.73	33.53	34.63	28.82	-5.81	33.53	34.63	28.29	-6.34	perfect reconstruction		
smoke	-	36.14	36.95	32.77	-4.18	36.13	36.94	32.76	-4.18	36.23	36.86	32.72	-4.14	36.23	36.90	32.69	-4.21	perfect reconstruction		
cloth	-	36.44	37.25	32.39	-4.86	36.33	37.24	32.41	-4.83	36.33	37.23	32.42	-4.81	36.53	37.23	32.49	-4.74	perfect reconstruction		
barbara	0.125	20.95	21.00	21.31	-0.31	21.02	21.06	21.38	-0.23	21.02	21.00	21.38	-0.42	20.98	21.09	21.37	-0.33	21.03	21.03	21.79
	0.25	21.76	21.80	22.45	-0.30	21.87	21.91	22.60	-0.04	21.83	21.87	22.56	-0.11	21.87	21.88	22.45	-0.26	21.99	21.86	24.05
	0.50	22.67	22.87	23.68	-0.01	22.68	22.84	23.97	-0.01	22.67	22.93	24.02	-0.03	22.68	22.91	23.70	-0.05	23.71	23.36	27.02
goldhill	0.125	23.99	24.07	24.59	-0.18	24.03	24.03	24.57	-0.27	23.98	23.93	24.55	-0.26	24.04	23.92	24.59	-0.24	24.03	23.96	24.93
	0.25	25.14	25.17	25.64	-0.71	25.11	25.13	25.67	-0.65	25.10	25.15	25.61	-0.72	25.08	25.14	25.60	-0.76	25.21	25.12	26.76
	0.50	26.34	26.36	26.44	-1.49	26.38	26.44	26.49	-1.40	26.38	26.41	26.44	-1.37	26.30	26.34	26.38	-1.37	26.75	26.47	29.00
peppers	0.125	22.28	22.36	23.54	-0.50	22.42	22.43	23.64	-0.38	22.31	22.50	23.50	-0.41	22.36	22.43	23.70	-0.38	22.46	22.43	24.27
	0.25	24.29	24.46	25.27	-1.71	24.15	24.42	25.42	-1.17	24.41	24.47	25.31	-1.34	24.26	24.34	24.43	-2.02	24.60	24.41	27.57
	0.50	26.44	26.69	26.71	-2.92	26.33	26.58	26.52	-2.23	26.55	26.61	26.48	-2.52	26.27	26.49	26.41	-1.87	27.25	26.89	31.38
lena	0.125	23.62	23.68	24.85	-0.37	23.68	23.80	24.91	-0.38	23.83	23.77	24.77	-0.43	23.64	23.77	24.79	-0.45	23.68	23.72	25.22
	0.25	25.36	25.50	26.48	-1.30	25.35	25.58	26.62	-1.21	25.37	25.59	26.70	-1.22	25.39	25.42	26.45	-1.37	25.37	25.46	28.03
	0.50	27.33	27.29	27.44	-2.80	27.30	27.35	27.76	-2.60	27.39	27.46	27.76	-2.58	27.32	27.38	27.52	-2.90	27.45	27.44	31.40
smoke	0.125	22.80	22.81	23.78	-0.53	22.78	22.80	23.81	-0.52	22.73	22.81	23.64	-0.70	22.74	22.77	23.63	-0.74	22.90	22.95	23.92
	0.25	24.09	24.40	24.68	-1.30	24.08	24.48	24.62	-1.36	24.09	24.49	24.70	-1.29	24.07	24.52	24.65	-1.36	24.11	24.56	24.83
	0.50	26.63	26.59	26.85	-0.40	26.60	26.55	26.84	-0.53	26.59	26.56	26.83	-0.55	26.52	26.58	26.81	-0.58	26.75	26.84	26.94
cloth	0.125	17.82	18.02	18.08	-0.77	17.88	18.01	18.09	-0.80	17.83	17.97	18.07	-0.80	17.84	18.00	18.11	-0.73	17.98	18.07	18.12
	0.25	18.36	18.40	18.68	-1.20	18.35	18.38	18.72	-1.11	18.37	18.39	18.70	-1.22	18.39	18.42	18.65	-1.17	18.40	18.46	18.73
	0.50	20.21	20.29	20.34	-0.60	20.20	20.33	20.36	-0.60	20.19	20.32	20.36	-0.58	20.22	20.31	20.32	-0.67	20.22	20.34	20.40

(b) Case IV to VII

Image	Bit Rate	Subdes. 1 Recvd.				Subdes. 2 Recvd.				Subdes. 3 Recvd.				Subdes. 4 Recvd.				Subdes. 1-4 Recvd.		
		IV	Degr.(IV)	VI	VII	IV	Degr.(IV)	VI	VII	IV	Degr.(IV)	VI	VII	IV	Degr.(IV)	VI	VII	IV	VI	VII
barbara	-	25.14	-0.75	25.21	25.89	25.09	-0.75	25.59	25.83	25.13	-0.74	25.60	25.87	25.09	-0.74	25.21	25.83	perfect reconstruction		
goldhill	-	30.69	-1.14	30.31	31.83	30.70	-1.17	30.41	31.87	30.65	-1.15	30.41	31.80	30.66	-1.18	30.31	31.84	perfect reconstruction		
peppers	-	31.69	-1.34	29.93	33.03	30.33	-1.72	30.01	32.05	30.86	-1.58	30.03	32.44	29.68	-1.88	39.91	31.56	perfect reconstruction		
lena	-	33.44	-1.11	31.29	34.55	33.43	-1.11	31.81	34.54	33.53	-1.10	31.82	34.63	33.53	-1.10	31.29	34.63	perfect reconstruction		
smoke	-	36.14	-0.81	33.49	36.95	36.13	-0.81	33.51	36.94	36.23	-0.63	33.42	36.86	36.23	-0.67	33.48	36.90	perfect reconstruction		
cloth	-	36.44	-0.81	33.74	37.25	36.33	-0.91	33.76	37.24	36.33	-0.90	33.78	37.23	36.53	-0.70	33.67	37.23	perfect reconstruction		
barbara	0.125	21.45	-0.17	21.60	21.62	21.44	-0.17	21.60	21.61	21.44	-0.16	21.60	21.60	21.50	-0.20	21.60	21.70	21.79	21.79	21.79
	0.25	22.47	-0.28	22.56	22.75	22.49	-0.15	22.56	22.64	22.50	-0.17	22.56	22.67	22.51	-0.20	22.56	22.71	24.05	24.05	24.05
	0.50	23.57	-0.12	23.63	23.69	23.58	-0.40	23.63	23.98	23.55	-0.50	23.63	24.05	23.56	-0.19	23.63	23.75	27.02	27.02	27.02
goldhill	0.125	24.54	-0.33	24.66	24.87	24.57	-0.27	24.66	24.84	24.53	-0.28	24.66	24.81	24.54	-0.29	24.66	24.83	24.93	24.93	24.93
	0.25	26.22	-0.13	26.24	26.35	26.22	-0.10	26.24	26.32	26.25	-0.08	26.24	26.33	26.23	-0.13	26.24	26.36	26.76	26.76	26.76
	0.50	27.69	-0.24	27.73	27.93	27.76	-0.13	27.73	27.89	27.74	-0.07	27.73	27.81	27.73	-0.02	27.73	27.75	29.00	29.00	29.00
peppers	0.125	23.99	-0.05	24.01	24.04	23.97	-0.05	24.01	24.02	23.89	-0.02	24.01	23.91	23.98	-0.10	24.01	24.08	24.27	24.27	24.27
	0.25	26.32	-0.66	26.40	26.98	26.34	-0.25	26.40	26.59	26.33	-0.32	26.40	26.65	26.37	-0.08	26.40	26.45	27.57	27.57	27.57
	0.50	28.22	-1.41	28.24	29.63	28.21	-0.54	28.24	28.75	28.20	-0.80	28.24	29.00	28.20	-0.09	28.24	28.29	31.38	31.38	31.38
lena	0.125	25.07	-0.15	25.08	25.22	25.15	-0.14	25.08	25.29	25.10	-0.10	25.08	25.20	25.17	-0.07	25.08	25.24	25.22	25.22	25.22
	0.25	27.74	-0.04	27.77	27.78	27.76	-0.07	27.77	27.83	27.75	-0.17	27.77	27.92	27.75	-0.07	27.77	27.82	28.03	28.03	28.03
	0.50	30.03	-0.21	29.95	30.24	30.01	-0.35	29.95	30.36	30.00	-0.34	29.95	30.34	30.00	-0.42	29.95	30.42	31.40	31.40	31.40
smoke	0.125	24.17	-0.14	24.18	24.31	24.23	-0.10	24.15	24.33	24.20	-0.14	24.13	24.34	24.27	-0.10	24.16	24.37	24.32	24.32	24.42
	0.25	24.84	-1.14	25.17	25.98	24.83	-1.15	25.24	25.98	24.83	-1.16	25.24	25.99	24.85	-1.16	25.17	26.01	24.86	25.25	26.03
	0.50	26.93	-0.32	26.99	27.25	26.91	-0.46	27.05	27.37	26.92	-0.46	27.05	27.38	26.95	-0.44	27.05	27.39	26.98	27.06	27.30
cloth	0.125	18.37	-0.48	18.48	18.85	18.35	-0.54	18.50	18.89	18.30	-0.57	18.48	18.87	18.37	-0.47	18.49	18.84	18.38	18.52	18.92
	0.25	19.04	-0.84	19.32	19.88	19.05	-0.78	19.32	19.83	19.05	-0.87	19.33	19.92	19.05	-0.77	19.35	19.82	19.08	19.37	19.93
	0.50	20.53	-0.41	20.75	20.94	20.57	-0.39	20.76	20.96	20.54	-0.40	20.73	20.94	20.54	-0.45	20.76	20.99	21.59	20.77	21.00

generally make reconstruction by interpolations or duplications very difficult, thus leaving more space for improvement by the ORB technique.

When no subdescriptions is lost, Cases III, IV, VI, and VII have the same results, because no reconstructions are needed; therefore, Cases III and IV have no degradations as compared to Case VII. In this situation, the results of frequency-domain methods are also better than those of sample-domain methods, because they suffer no segmentation loss.

Table 4.10 shows the results when four-way MDC is used. For four-way MDC, we apply both column and row interleaving. Similar to the two-way case, our experiments include cases without quantization and those with quantization; for quantization, three bit-rates are used.

The results are similar to those of the two-way cases. When there is no quantization, Cases II and VII have the best performance because they use interpolation as the reconstruction method, and no segmentation is needed in Case II.

When quantization is included, Case III, IV, VI, and VII also have similar performance to the two-way case, which is better than that of Cases I and II. The degradations of Case III from Case VII are usually from 0.3 to 1 dB, whereas degradations from Case IV to Case VII are usually from 0.2 to 0.5 dB. The gap between these two degradations is due to their different reconstruction methods.

In short, frequency-based schemes are better than sample-based schemes under controlled loss scenarios.

4.5.3 Experiments using trace-driven simulations

In this experiment, we test our algorithms in trace-driven simulations using traces collected in the Internet. Based on the prototype discussed in Chapter 3, we first transmitted 2000 UDP packets to the echo port of a destination computer and recorded the time stamps and sequence numbers in the packets bounced back. The time stamps and sequence numbers were used to determine the delay time and packet loss of each packet. Considering that Internet traffic might vary in time, we did this experiment at the beginning of every hour over a 24-hour period. We chose three destinations in this experiments: Taiwan, Thailand, and Argentina, as they represent low, medium, and high loss connections, respectively. For the low- and medium-loss connections, we use a two-way MDC and for the high-loss connection, we use a four-way MDC. All the test images were quantized using three bit-rates.

Table 4.11 gives the average PSNR over 24 hours for each connection and image. Here, the

Table 4.11 Companions of 24-hour average performance of the seven schemes. 32, 16, and 8 packets were transmitted at 0.5 bpp, 0.25 bpp and 0.125 bpp, respectively. For Case III, we have shown its degradations from Case VII, to better reflect its performance compared with Case VII.

Urbana –	Image	0.5 bpp				0.25 bpp				0.125 bpp			
		I	II	III	Degr.III	I	II	III	Degr.III	I	II	III	Degr.III
Taiwan	*lena*	29.38	29.12	33.50	-0.08	27.33	27.27	30.53	-0.05	25.47	25.47	27.61	-0.08
	barbara	24.66	24.67	28.34	-0.12	23.00	23.04	25.77	-0.11	21.84	21.85	23.26	-0.04
	peppers	29.10	29.19	32.26	-0.07	26.43	26.48	30.12	-0.06	24.26	24.27	27.24	-0.04
	goldhill	28.05	27.90	30.52	-0.09	26.25	26.29	28.47	-0.11	25.07	25.10	26.26	-0.19
	smoke	28.05	28.25	30.78	-0.32	26.20	26.34	28.64	-0.13	24.65	24.68	26.30	-0.15
	cloth	22.95	23.13	30.54	-2.66	20.36	22.14	26.19	-1.35	17.90	18.47	22.63	-0.34
Thailand	*lena*	28.95	29.10	32.74	-0.12	27.33	27.24	30.05	-0.20	25.45	25.47	27.32	-0.07
	barbara	24.00	24.30	27.57	-0.24	22.80	22.91	25.32	-0.17	21.71	21.72	23.05	-0.29
	peppers	29.00	29.10	31.17	-0.62	26.20	26.24	29.51	-0.05	24.15	24.30	26.83	-0.10
	goldhill	27.28	27.45	30.07	-0.11	26.23	26.27	28.14	-0.11	25.03	25.04	26.18	-0.13
	smoke	27.99	28.12	30.42	-0.59	26.14	26.33	28.00	-0.45	24.51	24.62	26.25	-0.18
	cloth	22.89	23.06	29.18	-1.80	20.35	21.54	25.13	-1.74	17.86	18.43	21.70	-0.73
Argentina	*lena*	27.40	27.42	28.41	-0.35	25.37	25.44	26.98	-0.13	23.64	23.70	24.91	-0.13
	barbara	22.99	23.34	24.37	-0.13	21.80	21.86	22.77	-0.43	20.99	21.01	21.62	-0.12
	peppers	26.67	26.87	27.04	-0.08	24.28	24.29	26.15	-0.46	22.40	22.41	23.78	-0.21
	goldhill	24.43	26.45	27.20	-0.14	25.15	25.12	25.94	-0.21	24.00	23.96	24.46	-0.22
	smoke	26.70	26.75	26.90	-0.33	24.10	24.45	24.74	-1.25	22.85	22.87	23.86	-0.52
	cloth	20.21	20.32	20.34	-0.63	18.38	18.42	18.70	-1.22	17.90	18.03	18.05	-0.85

Urbana –	Image	0.5 bpp				0.25 bpp				0.125 bpp			
		IV	V	VI	VII	IV	V	VI	VII	IV	V	VI	VII
Taiwan	*lena*	33.50	33.52	33.55	33.58	30.53	30.54	30.55	30.58	27.61	27.62	27.66	27.69
	barbara	28.34	28.38	28.42	28.46	25.77	25.79	25.83	25.88	23.26	23.29	23.34	23.36
	peppers	32.26	32.28	32.30	32.33	30.12	30.14	30.17	30.18	27.24	27.25	27.26	27.28
	goldhill	30.52	30.55	30.57	30.61	28.47	28.50	28.51	28.58	26.36	26.38	26.43	26.45
	smoke	30.80	30.85	30.89	31.10	28.69	28.72	28.73	28.77	26.35	26.39	26.43	26.45
	cloth	31.01	31.74	31.90	33.20	26.54	27.07	27.32	27.54	22.71	22.83	22.94	22.97
Thailand	*lena*	32.74	32.76	32.80	32.86	30.05	30.11	30.19	30.25	27.32	27.33	27.36	27.39
	barbara	27.57	27.64	27.72	27.81	25.32	25.37	25.46	25.49	23.05	23.11	23.19	23.34
	peppers	31.17	31.54	31.35	31.79	29.51	29.52	29.54	29.56	26.83	26.86	26.89	26.93
	goldhill	30.07	30.10	30.14	30.18	28.14	28.16	28.20	28.25	26.18	26.22	26.26	26.31
	smoke	30.43	30.61	30.63	31.01	28.20	28.31	28.35	28.45	26.30	26.33	26.41	26.43
	cloth	30.62	30.65	30.74	30.98	25.73	26.31	26.65	26.87	22.11	22.23	22.37	22.43
Argentina	*lena*	28.41	28.48	28.59	28.76	26.98	27.01	27.08	27.11	24.91	24.95	25.01	25.04
	barbara	24.37	24.40	24.46	24.50	22.77	22.86	22.99	23.20	21.62	21.64	21.68	21.74
	peppers	27.04	27.06	27.10	27.12	26.15	26.17	26.23	26.61	23.78	23.82	23.92	23.99
	goldhill	27.20	27.23	27.27	27.34	25.94	25.98	26.02	26.15	24.46	24.51	24.59	24.68
	smoke	26.95	27.03	27.05	27.23	24.85	25.20	25.25	25.99	23.89	23.95	24.06	24.38
	cloth	20.56	20.76	20.87	20.97	19.02	19.35	19.89	19.92	18.37	18.49	18.53	18.90

degradation of Case III from Case VII is also shown. In this table, we do not show the degradation from Case IV to VII because Case IV is not a feasible method in real practice as it requires doing interpolations for each code-block so the computation costs for it are high. However, we still evaluate its performance in this experiment but its degradation from the upper bound is not important here.

For the Urbana-Taiwan connection, frequency-domain based Cases III to VII outperform sample-domain based Cases I and II at all bit-rates. The performance gain is larger when the bit-rate is higher. For instance, in *lena*, the performance improvement of Case V over Case II is about 2.2 dB at 0.125 bpp and 4.4 dB at 0.5 bpp. This is not surprising because the number of segments increases when bit-rate increases. These segments, if eliminated, can contribute significantly to performance gain. Another observation is that the performance of Cases III to VII is very close to each other and is usually less than 1 dB apart. The degradations are mostly between 0.1 to 0.3 dB, with image *cloth* having a larger value as we explained before. As Case VII has the upper bound performance, we argue that Case III is the most practical method.

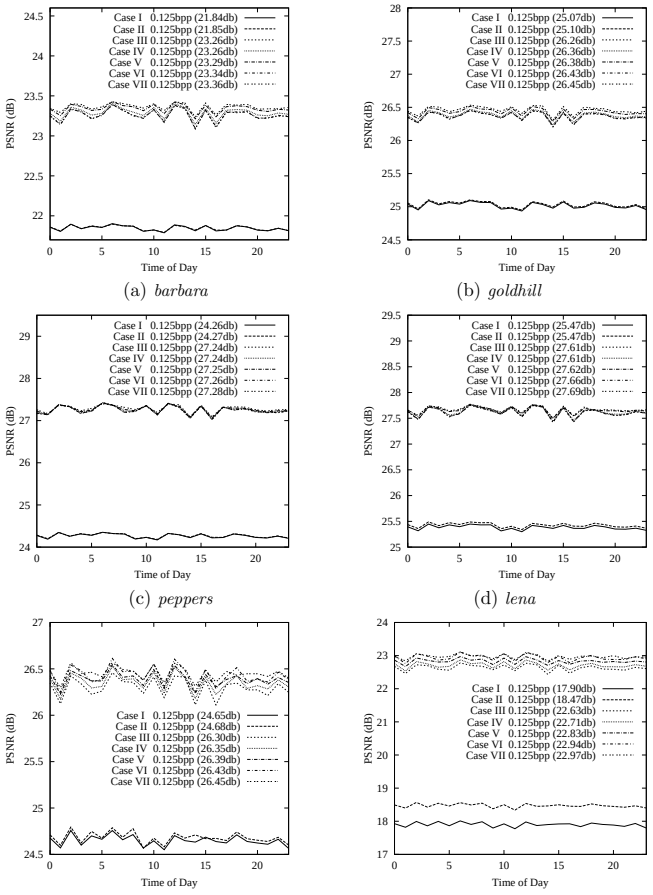
The result of the Urbana to Thailand connection is very similar to that of the Urbana to Taiwan connection. Case III outperforms Cases I and II and has close performances to that of Case VII with degradations between 0.1 to 0.6 dB.

For the Urbana to Argentina connection, four-way MDC is used, as the packet loss rate is too high for two-way MDC. The results of four-way MDC are also quite similar to those of the two-way MDC. The four frequency-domain based methods are clearly better than the two sample-domain based methods and have performance very similar to each other. Case III still outperforms Cases I and II and has close performance to that of Case VII with degradations between 0.1 to 0.3 dB.

In addition to showing the average PSNRs, we also show the PSNR at every hour in Figures 4.8 to 4.16. We can observe clearly that frequency-based schemes, are better than sample-based schemes and the performance of all frequency-based schemes are close. This indicates that Case III is a good alternative for real image transmissions.

4.5.4 Evaluating quality-delay tradeoffs of the proposed algorithms

After testing the performance of our proposed algorithms in a trace-driven simulations, we evaluate their quality-delay tradeoffs and compare them with those of the TCP. We notice that there is still performance degradation in multiple description coding (MDC) when packets are transmitted by UDP as compared to single description coding (SDC) when transmitted by TCP. Such tradeoffs should be further studied since quality-delay tradeoffs are what the users are interested in.

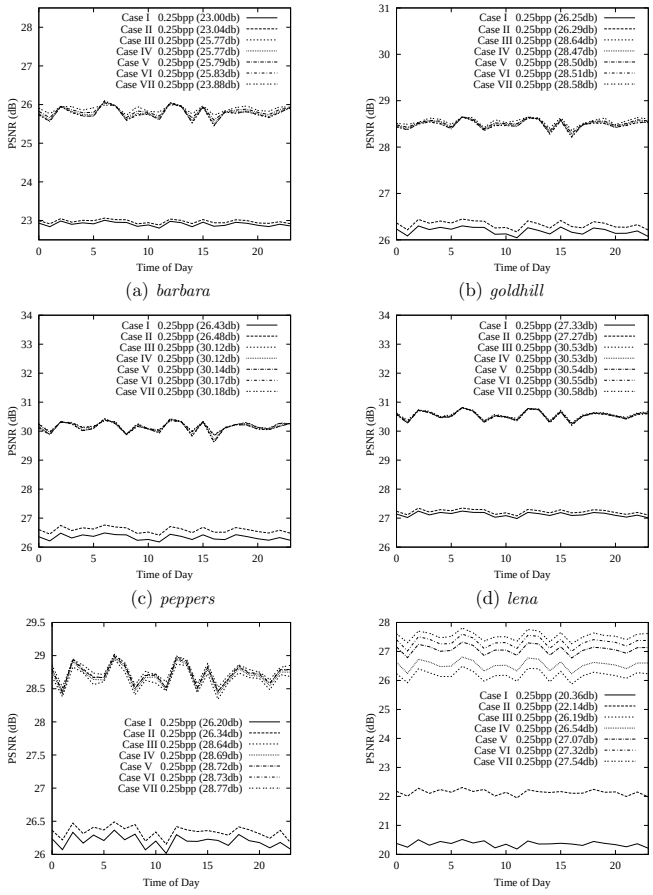


(a) *barbara* (b) *goldhill*

(c) *peppers* (d) *lena*

(e) *smoke* (f) *cloth*

Figure 4.8 Comparisons of reconstruction quality over a 24-hour period for the Urbana to Taiwan connection, when each image was coded at 0.125 bpp and placed into 8 UDP packets for transmission.



(a) *barbara* (b) *goldhill*

(c) *peppers* (d) *lena*

(e) *smoke* (f) *cloth*

Figure 4.9 Comparisons of reconstruction quality over a 24-hour period for the Urbana to Taiwan connection, when each image was coded at 0.25 bpp and placed into 16 UDP packets for transmission.

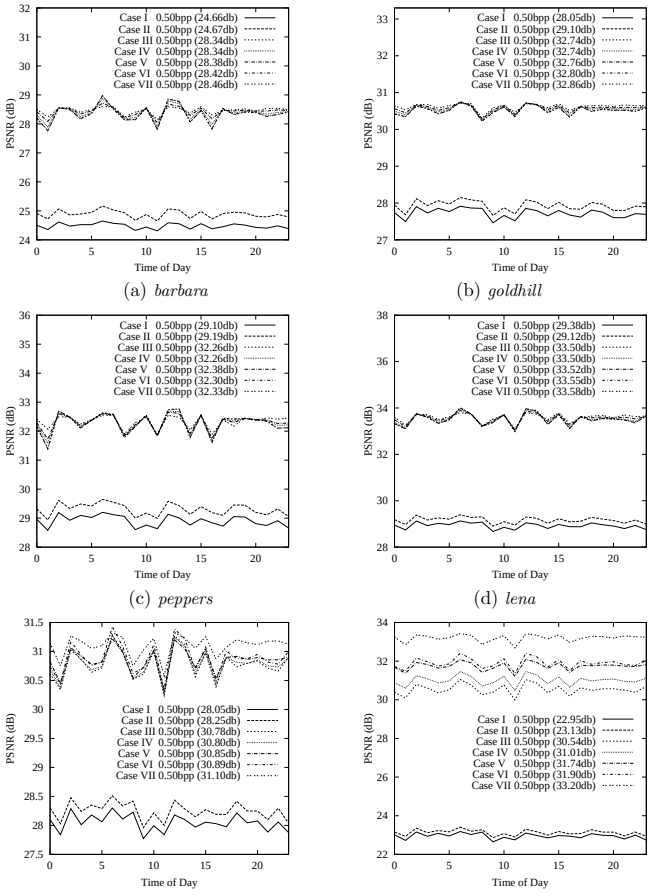


Figure 4.10 Comparisons of reconstruction quality over a 24-hour period for the Urbana to Taiwan connection, when each image was coded at 0.50 bpp and placed into 32 UDP packets for transmission.

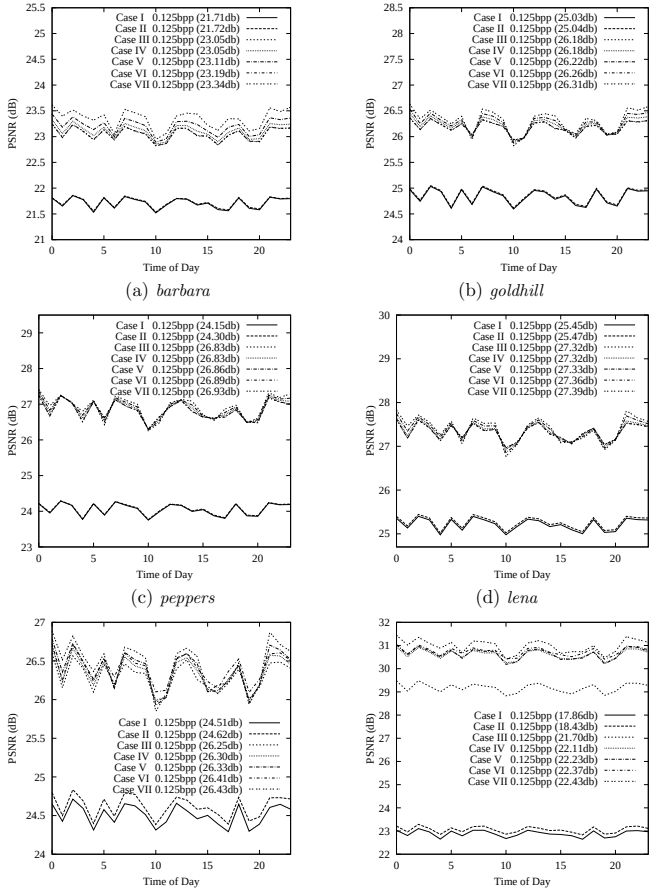


Figure 4.11 Comparisons of reconstruction quality over a 24-hour period for the Urbana to Thailand connection, when each image was coded at 0.125 bpp and placed into 8 UDP packets for transmission.

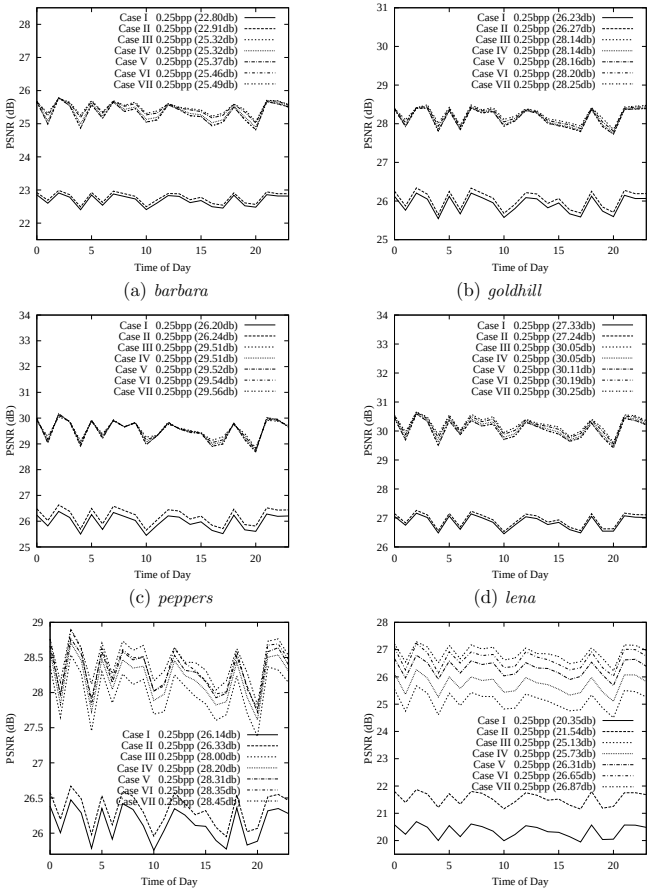


Figure 4.12 Comparisons of reconstruction quality over a 24-hour period for the Urbana to Thailand connection, when each image was coded at 0.25 bpp and placed into 16 UDP packets for transmission.

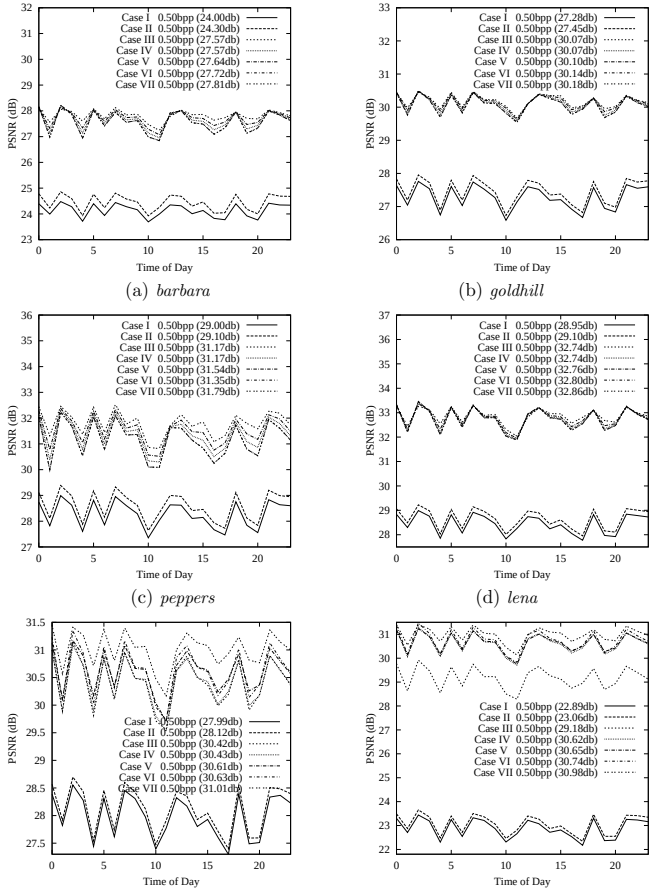
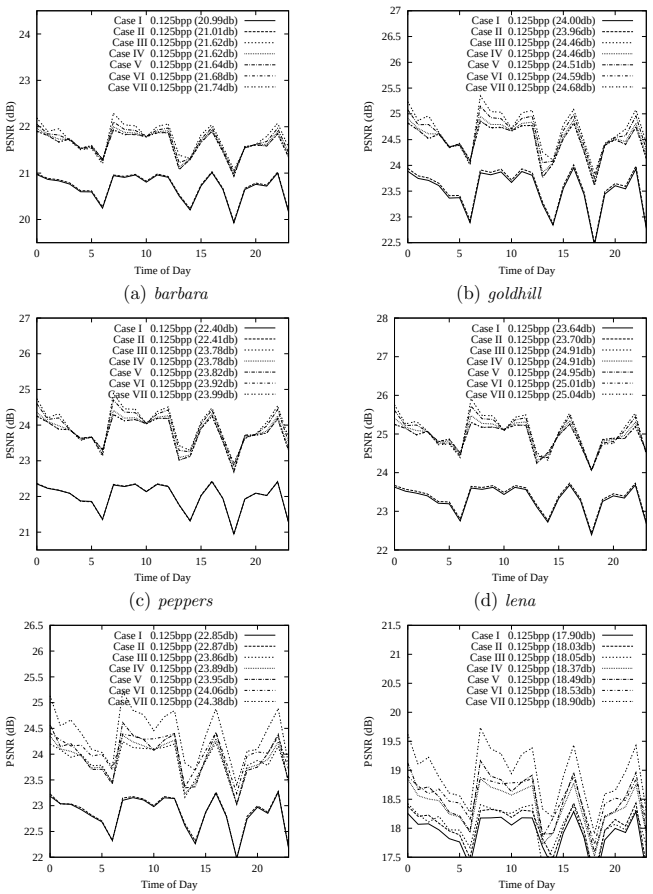


Figure 4.13 Comparisons of reconstruction quality over a 24-hour period for the Urbana to Thailand connection, when each image was coded at 0.50 bpp and placed into 32 UDP packets for transmission.

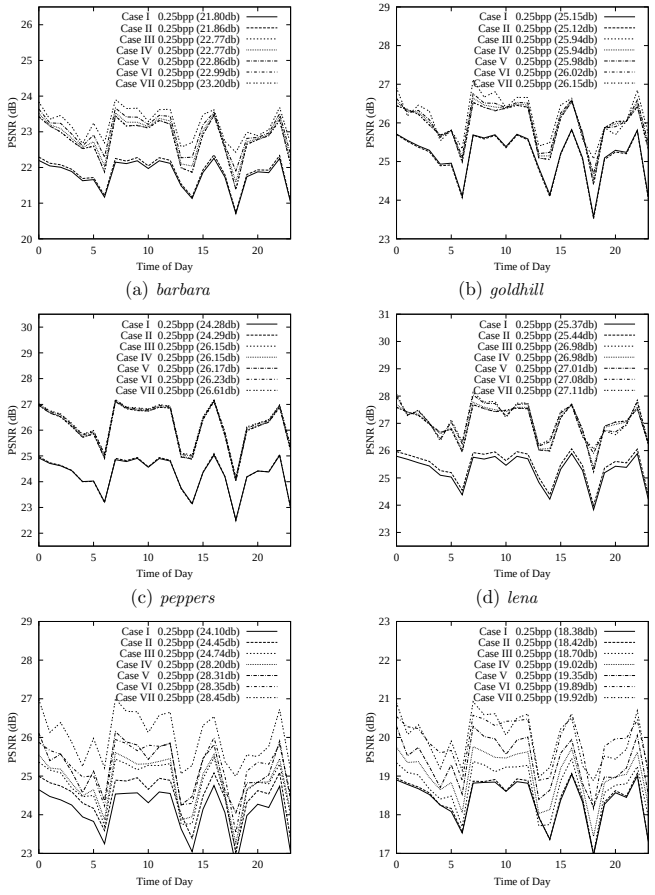


(a) *barbara* (b) *goldhill*

(c) *peppers* (d) *lena*

(e) *smoke* (f) *cloth*

Figure 4.14 Comparisons of reconstruction quality over a 24-hour period for the Urbana to Argentina connection, when each image was coded at 0.125 bpp and placed into 8 UDP packets for transmission.



(a) *barbara* (b) *goldhill*

(c) *peppers* (d) *lena*

(e) *smoke* (f) *cloth*

Figure 4.15 Comparisons of reconstruction quality over a 24-hour period for the Urbana to Argentina connection, when each image was coded at 0.25 bpp and placed into 16 UDP packets for transmission.

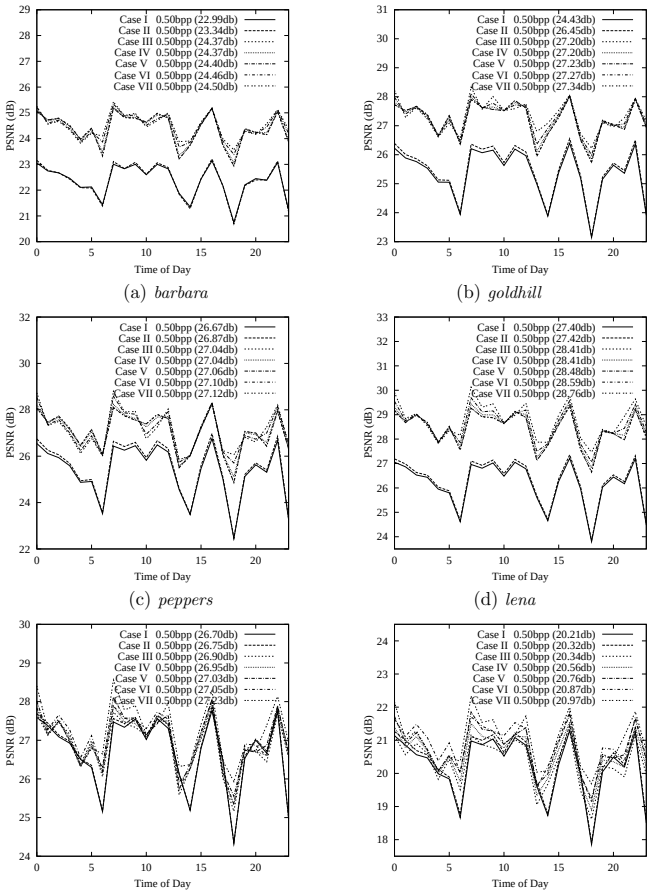


Figure 4.16 Comparisons of reconstruction quality over a 24-hour period for the Urbana to Argentina connection, when each image was coded at 0.50 bpp and placed into 32 UDP packets for transmission.

In this subsection, we test three delivery methods: TCP delivery of SDC images, TCP delivery of MDC images, and UDP delivery of MDC images.

For UDP delivery simulation, we follow the same method discussed in Chapter 3. The delay and loss of each packet are pre-stored. The receiver application performs reconstruction if there is loss after it receives the whole batch of UDP packets transmitted. Moreover, there are two alternatives in UDP delivery, one with retransmission and another one with no retransmission. As UDP retransmission has to be performed by the application, we consider this alternative not suitable as it requires all clients using an extra application level protocol to communicate with the server, which is not so simple to deploy. So for UDP delivery, we only discuss the case when no retransmission is performed.

For TCP delivery simulation, we also send each packet at a 30-ms interval at application level . However, these packets will be buffered by the TCP kernel and the exact sent time is controlled by TCP itself. On the receiver side, the hand-over of these packets from TCP buffer to the application is also controlled by the kernel. Only in-order packets will be returned after a read of the buffer. When an in-order packet is read by the application, we record its arrival time and calculate its delay from the time stamp in the packet. As the receiver and the sender are on the same computer in our simulation, the time is synchronized. Each time a TCP packet is received, the decoder will decode all available packets up to this point, setting all coefficients in later packets to zero. The quality of this decoded image is considered the quality we can get at this time.

We simulate the situation when each of the four images, compressed at 0.5 bpp, is transmitted to each of the destinations in 32 UDP packets. We choose the highest of the three bit-rates because it can provide 32 data points and can reflect better the quality-delay tradeoffs. We first use trace-driven simulations to send batches of 32 UDP packets to the destination at each destination's local time of 12 noon. The average arrival time of each packet is calculated for the three connections for each of the three methods used. Each time a new packet is received, the decoded image quality is evaluated. The delay time for each arrived packet is the end-to-end delay plus decoding time.

Figures 4.17-4.19 show the quality-time relationship for the three connections, respectively. The two curves are the quality-delay performance of TCP+SDC and TCP+MDC, respectively. TCP+SDC always has a better performance than TCP+MDC because MDC sacrifices coding efficiency in order to provide robustness. This performance gap can be considered as the tradeoff between quality and robustness.

As UDP packets may be received out-of-order, we assume that we can decode an image only when all its UDP packets have been received. Hence, the quality-delay behavior of MDC+UDP delivery is plotted as a single point. We see that UDP+MDC delivery has a much shorter delay than that of TCP+SDC case, although its performance is not as good as that of the TCP+SDC case. The delay of MDC+UDP delivery relative to that when minimum quality is achieved in TCP+SDC depends on the loss rate. In a low-loss connection, as shown in Figure 4.17, MDC+UDP delivery will have a longer delay than that when TCP+SDC achieves its minimum quality is achieved. As the loss rate increases, as shown in Figures 4.18 and 4.19, MDC+UDP will have a shorter delay than that when TCP+SDC achieves its minimum quality. In this case, MDC+UDP shows better quality-delay tradeoffs.

To better illustrate the performance of such tradeoffs, Figures 4.20 and 4.21 show the results of the transmitted *lena* and *smoke* to Thailand and Argentina, respectively, using three different methods. It shows that MDC+UDP provides a good image quality and a shorter delay.

In conclusion, TCP+SDC and TCP+MDC usually have long delays and our proposed MDC+UDP is a viable alternative delivery method with better quality-delay tradeoff.

4.6 Summary

In this chapter, we first give the problem statement and our assumptions. We then show an overall system architecture of our frequency-based MDC scheme and we study the correlations and reconstruction quality between two subdescriptions in individual subband to justify this architecture. We also propose a modified ORB-ST technique for this scheme and derive a subband decomposed reconstruction method to better improve the reconstruction quality. Finally, we showed some experimental results and the conclude from them that our proposed method can be a good alternative in transmitting images over the Internet.

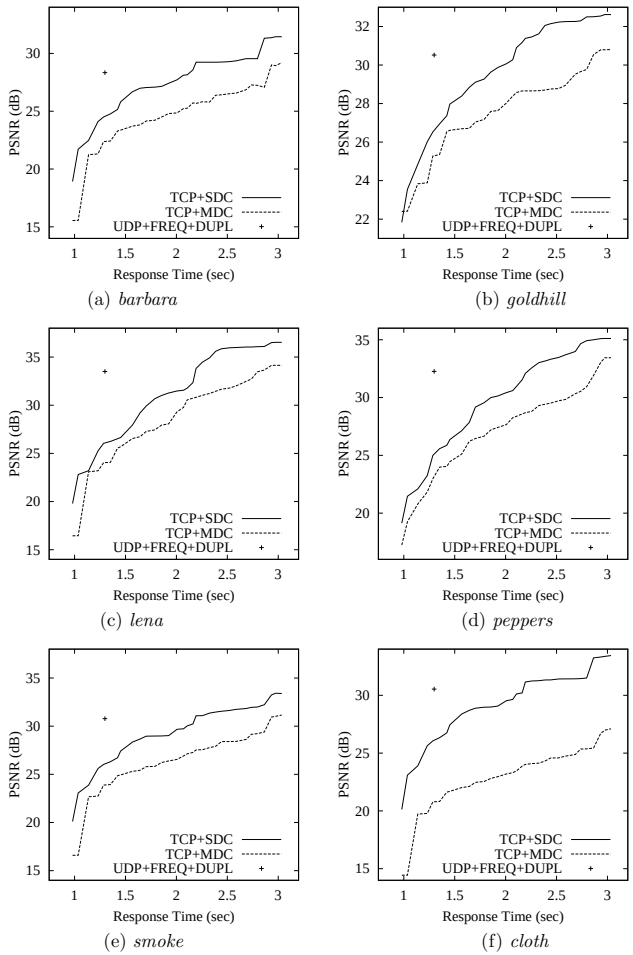
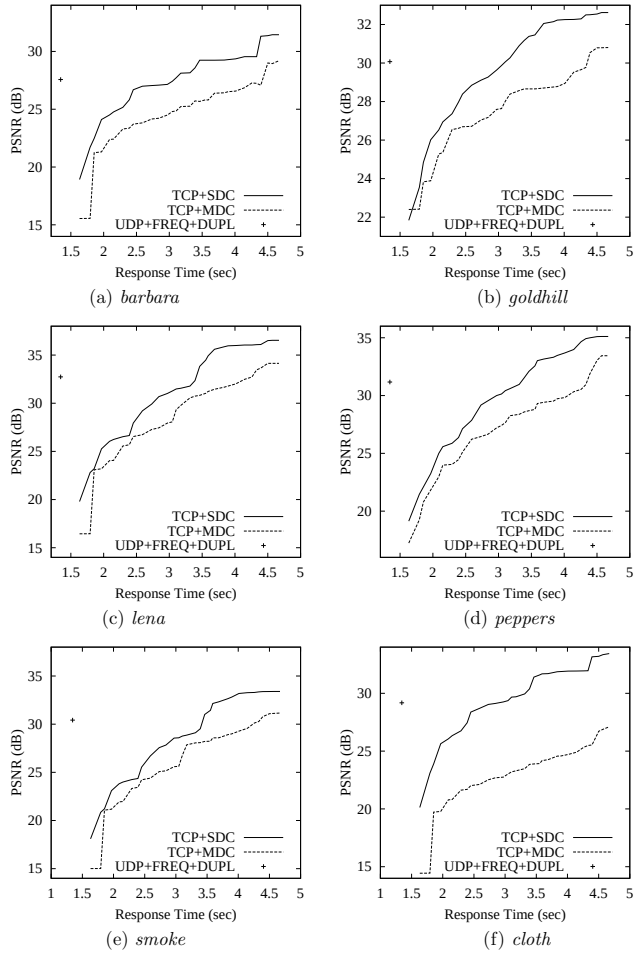


Figure 4.17 Quality-time trade-offs between TCP delivery of SDC/MDC image data and UDP delivery of MDC data for the Urbana-Taiwan connection at 12 noon Taipei local time. (The behavior at other times are similar and are not shown.)

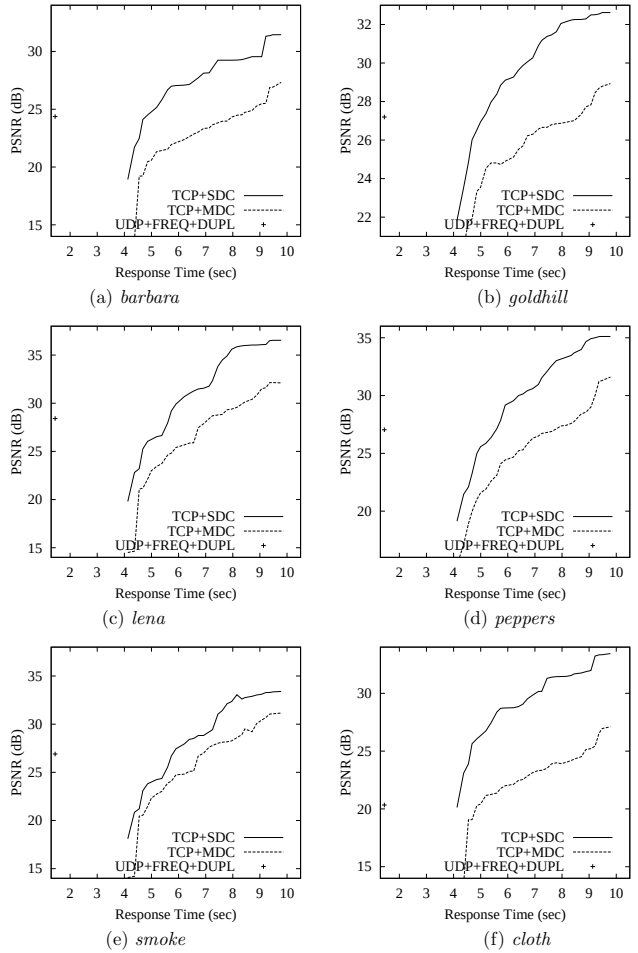


(a) *barbara* (b) *goldhill*

(c) *lena* (d) *peppers*

(e) *smoke* (f) *cloth*

Figure 4.18 Quality-time trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-Thailand connection at 12 noon Bangkok local time. (The behavior at other times are similar and are not shown.)



(a) *barbara* (b) *goldhill*

(c) *lena* (d) *peppers*

(e) *smoke* (f) *cloth*

Figure 4.19 Quality-time trade-offs between TCP delivery of SDC image data and UDP delivery of MDC data for the Urbana-Argentina connection at 12 noon Buenos Aires local time. (The behavior at other times are similar and are not shown.)

(a) SDC by TCP (b) SDC by UDP (c) Proposed MDC by UDP

Figure 4.20 Quality-delay trade-offs in round-trip transmissions of *lena* compressed at 0.125 bpp by JPEG2000 between UIUC and Thailand. In UDP transmissions, two out of the eight packets were lost. Image (a) has a quality of 30.97 dB and a delay of 4.01 sec. Image (b) has a quality of 20.51 dB and a delay of 0.71 sec. Image (c) has a quality of 25.21 dB and a delay of 0.71 sec.

(a) SDC by TCP (b) SDC by UDP (c) Proposed MDC by UDP

Figure 4.21 Quality-delay trade-offs in round-trip transmissions of *smoke* compressed at 0.25 bpp by JPEG2000 between UIUC and Argentina. In UDP transmissions, five out of the sixteen packets were lost. Image (a) has a quality of 30.96 dB and a delay of 13.03 sec. Image (b) has a quality of 22.03 dB and a delay of 0.46 sec. Image (c) has a quality of 28.72 dB and a delay of 0.46 sec.

5 UNEQUAL ERROR PROTECTION

In the previous chapter, we have proposed a new frequency-based MDC algorithm to protect image transmissions by UDP over the Internet. Although the proposed scheme performs quite well under both low-loss and high-loss situations, it is unable to control the amount of redundancy in the code stream adaptively to loss situations. The redundancy in a code stream in the proposed MDC algorithm is the implicit redundancy between the odd and the even subdescriptions and cannot be exploited during encoding. This implicit redundancy may be an over-protection for the code stream in low-loss situations, leading to unnecessary degraded quality. This motivates us to study the problem of protecting JPEG 2000 images adaptively according to the degree of channel loss, which we assume to be available via channel feedbacks.

One well-known approach for controlling redundancy in a code stream is through channel coding, or adding FEC codes. Many efforts have been made to study channel coding as a separate problem in the past decades. In recent years, there is more interest in joint channel source coding (JSCC) that optimizes channel loss and source distortion jointly in order to minimize the distortion at a receiver. It is especially useful in protecting multimedia data transmissions over the Internet. Although Shannon's separation theorem states that optimizing source coding and channel coding separately is equivalent to optimizing them together, the latter approach usually has an advantage of having computationally inexpensive heuristics.

The idea of JSCC leads to the unequal error protection (UEP) scheme, which protects different parts of the source with different amounts of protection. UEP scheme can be applied not only for real time communication over Internet ([22]), but for mobile communications ([23]) as well. There are a lot of open problems in this research area, including designing different codes for different networks ([24, 25]) or designing different packing schemes of code-streams for different applications ([26]). In this chapter, we study several UEP schemes for protecting JPEG 2000 images transmissions over Internet, and propose new UEP algorithm with better decoded image quality.

This chapter is organized as follows: Section 5.1 presents the formulation of our problem and introduces the basic components of UEP schemes. Section 5.2 discusses the serial and the parallel packing UEP schemes. Section 5.3, we proposes a new hybrid packing UEP scheme. Finally, Section 5.5 shows the experimental results of our proposed UEP schemes.

5.1 Overview of UEP Algorithms

In this section, we briefly overview the problem of applying unequal error protection (UEP) to image transmissions over the Internet. We first give the problem statement, followed by a discussion on three basic components of UEP schemes: channel models, FEC codes, and packing algorithms.

5.1.1 Problem Statement

In this subsection, we formulate the UEP problem by stating its goal and assumptions. We aim to send a compressed JPEG 2000 code stream to a receiver by UDP over the Internet, protected by FEC codes, in such a way that the receiver can decode it with certain quality when part of the code stream is lost during transmission.

The problem is solved under the following assumptions:

Assumption 5.1: We assume that a JPEG 2000 code stream can be divided into code blocks that are independent decodable units. These code blocks have different contributions to quality and have different sizes. The contribution of a code block to quality is measured by the distortion that losing this particular code block can bring to the decoded image quality.

Assumption 5.2: We assume that the bandwidth to transmit the entire image is fixed and depends on the image size and compression rate. For example, if the total size of the original compressed image is 4KB, then the code stream after applying UEP is also 4KB. In other words, we have to trade the space in the original code stream for FEC codes in applying UEP.

Assumption 5.3: We assume that the sender can obtain delayed feedbacks, in which the average delay time for feedbacks is connection-dependent.

Our problem is to design an efficient UEP scheme by putting suitable amount of FEC codes in the code stream in order to protect each part of the code stream according to its contribution to the final decoded image quality. We first introduce some of the symbols used (Table 5.1) before we formulate the problem. Let the encoder output be a compressed JPEG 2000 code stream with N_c code blocks, indexed by i, and the bandwidth is N_p UDP packets, each having a space of 548 octets (bytes). Code block i has size S_i, measured in bytes, and a value V_i that measures the contribution

of this code block to the final image quality. Each code block also has a weight W_i, defined as V_i/S_i.

Our goal in this chapter is to design an algorithm to assign FEC codes to the code stream within the given bandwidth in order to to achieve high average PSNR and low undecodable probability. Here, the undecodable probability is defined as follows:

Definition 5.1: The *undecodable probability* is the probability that the first code block, which contains the lowest frequency part, or the header, is either lost or cannot be recovered. Under such situations, either the code stream is undecodable or the decoded image is not meaningful as its lowest frequency part is missing.

The performance measures used are the average PSNR for each loss scenarios and the undecodable probability. There are undesirable cases in which the average PSNR is high but the undecodable probability is also high. For example, suppose one can receive a perfect image (50 dB) with an undecodable probability of 50%, and another can receive an image at 25 dB with an undecodable probability of 1%. The latter case is more preferred although it has a lower average PSNR because users usually would not tolerate frequent total losses.

There are three key issues in designing a UEP scheme to achieve our goal: the type of channel loss model, the type of FEC codes, and the type of packing strategy. These three factors will affect the performance of a UEP scheme. We discuss these issues in the following subsections.

5.1.2 Channel loss model

One of the fundamental issues in designing a UEP algorithm is the design of a good channel loss model. A good channel loss model should have two properties. The first is that it should be able to predict future channel losses accurately. The second is that the parameters of this model should be easily extracted by the receiver and fed back to the sender.

It is well-known that modelling Internet packet losses is very difficult because the Internet is a very large scale interconnection of many heterogeneous networks. A widely accepted model for modelling Internet packet losses is the multi-state hidden Markov model. However, it is usually difficult to extract the parameters for this model because the number of hidden states is hard to decide.

Although it is difficult to accurately predict whether a single packet will be lost or not, it is possible for a sender to predict $P(m, n)$, the probability that m packet are lost the transmission of n packets. Most channel models allow the sender to obtain $P(m, n)$ with some parameters fed

Table 5.1 Symbols used in this chapter.

Symbols	Definitions
N_c	Total number of code blocks generated by the encoder
N_p	Total number of packets allowed to use
i	Code block index
S_i	Size of code block i (in bytes), $i \in \{1, 2, ...N_c\}$
V_i	Value of code block i (in bytes), $i \in \{1, 2, ...N_c\}$
W_i	Weight of code block i (in bytes), $W_i = V_i/S_i$, $i \in \{1, 2, ...N_c\}$ $W_i = V_i/S_i$
$P(m, n)$	Probability that m packets are lost during a batch of transmission of n packets
$P_0(t)$	Probability that t packets are received when transmitting N_p packets $P_0(t) = P(t, N_p)$
$P_c(t)$	Probability that at least t packets are received when transmitting N_p packets $P_c(t) = \sum_{i=t}^{N_p} P_0(t)$
P_u	Maximum undecodable probability a user can tolerate
$L(i)$	Protection level of code block i
p	Average packet loss rate
p_o	Threshold of Over-estimated packet loss rate
p_u	Threshold of Under-estimated packet loss rate
$\hat{p}$	Estimated packet loss rate from feedback
V	Loss pattern
$\mathcal{V}$	All possible loss patterns
n_0	Number of lost packets in a loss pattern
n_1	Number of received packets in a loss pattern

back from the receiver. In this chapter, we assume that the receiver sends the loss rate of the most recent batch of packet transmissions, to the sender. The sender obtains $P(m, n)$ by assuming that each packet in the next batch of transmission has a probability of p to be lost; and within this batch, losses are independent from each other.

5.1.3 Introduction of Reed-Solomon code

One of the best choices of FEC codes for erasure channels is the Reed-Solomon (RS) code. An (N, k) RS code contains N symbol, of which k are useful information symbols and $N - k$ are protection symbols. One symbol in an RS code is generally not a bit, but a combination of n bits. RS codes cannot be created with an arbitrary length. The maximum length of an RS code is determined by $N_{max} = 2^n - 1$.

An (N, k) RS code can recover all lost symbols if the number of lost symbols does not exceed $N - k$. This is very useful in the case of erasure errors. In our design introduced later in this chapter, we choose to use RS codes.

5.1.4 Packing algorithms for error protection

We study in this subsection various packing algorithms, starting from equal error protection (EEP) to unequal error protection (UEP).

Packing algorithms usually determine the amount of protection on each part of a code stream. We illustrate six such packing algorithms in Figure 5.1. In this example, we assume that an image is encoded into $N_c = 14$ code blocks to be transmitted in $N_p = 5$ UDP packets. Assuming each is smaller than a UDP packet, the situation in which a single code block cannot fit into a UDP packet will not happen. The goal of the algorithm is to pack these N_c code blocks and extra RS codes efficiently into the N_p packets in such a way that we can recover the code blocks with as little distortions as possible when losses happen.

The first intuitive way of packing the code blocks is to put the N_c code blocks into the N_p packets sequentially and fill the rest of the space by extra RS codes (Figure 5.1a). This method usually has to segment code blocks at each packet boundary, which renders those code blocks undecodable when part of them are lost.

An improved method re-arranges the order of the code blocks and RS codes in the first method before putting them into packets in order to avoid the situation in which a code block spans across a packet boundary (Figure 5.1b). In this case, we do not have the issue in the first method. However,

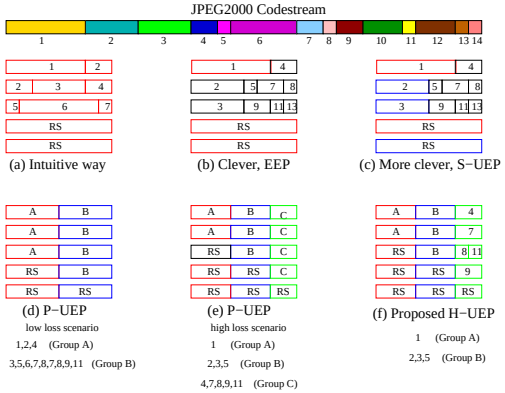


Figure 5.1 Various packing algorithms discussed in this chapter. From top left to bottom right, we list simple examples of EEP, S-UEP, P-UEP and H-UEP packing algorithms, respectively.

since the RS codes protects all the code blocks equally, this method is actually an EEP scheme. As a result, some parts of the code stream may not be protected adequately, while some parts of the code stream may be over-protected, as different code blocks have different contributions to the decoded image quality. This motivates us to find an effective UEP scheme for packing code blocks.

5.2 UEP Packing Algorithms

In this section we study two UEP packing algorithms in detail, namely, serial and parallel packing algorithms. Our goal is to maximize the average PSNR and minimize the undecodable probability of our packing algorithm.

5.2.1 Serial packing algorithm

The first algorithm studied is the serial UEP (S-UEP) packing algorithm shown in Figure 5.1c. In S-UEP, an entire information code block or a FEC code block will not be split and placed in different packets during packing. The goal of S-UEP is to achieve the maximum average PSNR by arranging a different amount of FEC codes to information code blocks of different importance.

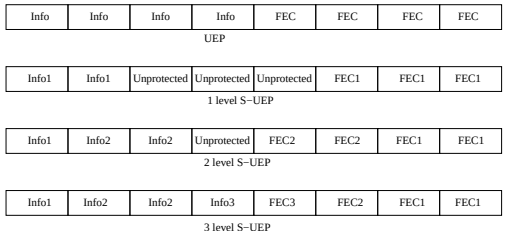


Figure 5.2 Illustration of serial packing algorithms.

Before introducing this algorithm in detail, we first define a *protection group*.

Definition 5.2: A protection group is a collection of information code blocks and the corresponding FEC code blocks that are used to protect these information code blocks.

Next, we illustrate our S-UEP algorithm. Let the number of total packets, N_p, be 8 and the packets be indexed from 1 to 8. Figure 5.2 shows the result of S-UEP packing. As a comparison, we also include an example of EEP in the figure.

In EEP, we have only one parameter a that controls the amount of FEC codes added to the code stream. Packets 1 to a contain information code blocks (labeled **Info** in the figure), and packets $a+1$ to 8 contain RS codes (labeled **FEC**) that are used to protect the information code blocks in packets 1 to a uniformly. In EEP, we only have one protection group.

In S-UEP, the packing algorithms can be further classified by the number of protection groups. In this chapter, we name an S-UEP algorithm with $n+1$ protection groups an n-level S-UEP. For instance, a 1-level S-UEP has two protection groups that are controlled by two parameters a and b. Packets 1 to a are information code blocks (labeled **Info1** in the figure) in the first protection group, whereas packets $8-b+1$ to b (labeled **FEC** in the figure) are the RS codes in the first protection group. Packets $a+1$ to $8-b$ are information code blocks in the second protection group with no RS protection codes (labeled **Unprotected**).

We also demonstrate in Figure 5.2 the 2-level and 3-level S-UEP cases that are similar to the 1-level S-UEP algorithm. In these multi-level S-UEP algorithms, information code blocks are divided into several protection groups, like **Info2** with **FEC2** and **Info3** with **FEC3**. Each protection group has a different amount of information code blocks and the corresponding RS protection codes.

The above example clearly shows that there are a quality-reliability trade-offs in S-UEP schemes. When we pack more information code blocks and less RS codes into packets, we have more contributions to the decoded image quality. On the other hand, when we pack more RS codes and less information code blocks, we have more ability to recover lost packet. To find the optimal number of protection groups and the packet an information code block should be assigned to require an enumeration. In a case with a small total packet number like 8, enumeration is computationally acceptable.

One of the limitations of S-UEP is that its undecodable probability may be high. The reason is that the most important code block, or the first code block, is not protected by all packets. For instance, in the third example in Figure 5.2, the first code block in the group labeled **Info1** is protected by code blocks labeled **FEC1**. If both **Info1** and **FEC1** (three packets) are lost, the code stream will be undecodable. In high-loss scenarios like an intercontinental connection, losing three out of eight packets can happen frequently; therefore, we want to design a packing algorithm that can produce a high average PSNR with low undecodable probability.

5.2.2 Parallel packing algorithm

In this subsection, we describe a parallel UEP (P-UEP) packing algorithm illustrated in Figures 5.1d and 5.1e. As mentioned at the end of the last subsection, our goal is to design a packing algorithm that can produce a high average PSNR with low undecodable probability.

Figure 5.3 shows the structure of P-UEP. Assuming the number of packets, N_p, to be 8, we first define the term *protection level*:

Definition 5.3: All information code blocks in a protection group with a *protection level* i are protected by an (N_p, i) RS code. In other words, receiving i packet in a batch of N_p packets allows the recovery of all lost code blocks.

In Figure 5.3, as an illustration, code block 1 is in a group with a protection level of 7. Hence either one of these eight code blocks received allows us to recover code block 1. As code block 1 is protected by seven other code blocks, it can be recovered when any of the 8 packets is received.

In P-UEP, we not only allow an entire information code block or an FEC code block be split into different packets, we further require that each protection group be split in every packet. In Figure 5.3, each row numbered T1 to T8 stands for a packet. A numbered box stands for an information code block(s). We see that the first information code block, with seven other RS code blocks, form the first protection group with a protection level of 7. The second and the third code

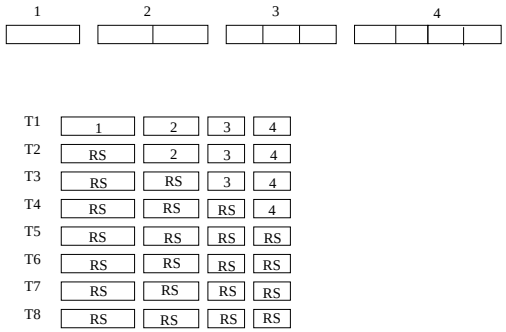


Figure 5.3 Illustration of parallel packing algorithm.

blocks, with six other RS code blocks, form the second protection group with protection level 6. The rest of the protection groups are formed in a similar way. Such a packing scheme allows the most important code block to be recovered as long as any packet is received.

In P-UEP algorithms, we need to determine the protection groups and the protection level of each. This is equivalent to assigning a protection level to each code block. Like S-UEP, the optimal assignment of a protection level to each code block has to be enumerated. In P-UEP, this enumeration is usually to computationally expensive, even in the case of 8 packets. Therefore, we choose to use constrained simulated annealing (CSA) [27] to find a solution.

P-UEP can decrease the undecodable probability as well as increase the decoded image quality. But for groups with lower protection levels (i.e. groups of less important code blocks), doing UEP actually leads to lower quality. This is true because a group with a protection level i needs i packets in order for it to be decoded. Hence, receiving less than i code blocks is not useful even if these received code blocks do carry information. Although this kind of code blocks may not carry fundamental information of an image, we still want to rearrange the code blocks in order for them to be decodable.

5.3 Proposed Hybrid Packing Algorithm

In this section, we propose a new hybrid UEP packing (H-UEP) algorithm. In order to solve the problem mentioned at the end of the last subsection, we combine S-UEP and P-UEP together.

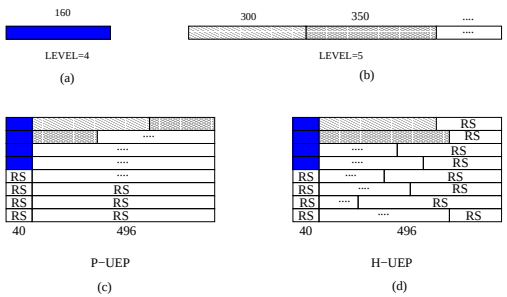


Figure 5.4 Illustration of hybrid packing algorithms.

In H-UEP, an entire information code block can be either put into one packet or split into different packets; each protection is then either split across every packet or distributed to some the packets. The idea of H-UEP is to use P-UEP for a few important code blocks in the beginning of a code stream and use S-UEP for the less important code blocks. It is shown as the sixth case in Figure 5.1.

Figure 5.4 further illustrated the structure of the proposed H-UEP. In Figures 5.4a and 5.4b, we show the original code stream. Figure 5.4a shows the code blocks that are assigned with a protection level 4, and Figure 5.4b shows the code blocks with a protection level 5. Figure 5.4c is the outcome of P-UEP algorithm. We can see that in this packing, the code blocks of protection group with level 5 are split to each packet. If only the second packet is received, the code blocks in protection group with level 5 cannot be recovered. This will lower the average PSNR. In Figure 5.4d, which is the outcome of H-UEP, the code blocks in the group with a protection level 5 are repacked from a parallel style into a serial style. When only the second packet is received, the 350-octet code block in level 5 can still be recovered, therefor increasing the average PSNR in this case.

Similar to P-UEP, H-UEP needs complicated optimization and is computationally prohibitive, we need to find out a feasible heuristic for it. As a result, we need to find an efficient heuristic for optimizing the code-block assignment. We propose a greedy algorithm that places code blocks with more quality contributions into groups having more protection, while keeping the undecodable probability under a predefined threshold. Note that in our H-UEP strategy, the greedy packing heuristic has to be combined with the repacking process.

The details of the algorithm is presented in Figure 5.5. The outputs of the algorithm are

$L(i), i = 1..N$, which are the protection levels of code-block i. Moreover, we assume that the code-blocks indices are sorted by $W(i)/S(i)$ in a descending order beforehand; hence, code-blocks with lower indices have more contributions per byte to the final image quality. In out algorithm, we first set the current protection level to 1 (**Line 4**), which is the most protection we can assign to a code-block. For each code-block, we calculate its gain and size if the current protection level is assigned to this code-block (**Line 7,9**) or if the new protection level (current protection level increased by 1) is assigned (**Line 8,10**). We compare the efficiency, which is gain divided by size, for both assignment (**Line 13**). The current protection level assignment is set to the new assignment if the new level leads to more efficient coding (**Line 19-21**). An exception of this is that at the beginning of a code stream, we will not reduce the amount of protection until the undecodable probability is lower than the pre-defined value. (**Line 17-18**).

After the algorithm is executed, each code-block is assigned a protection level. Our packing algorithm packs each packet and its RS protection codes from the beginning of the sorted code-block sequence according to the protection level it is assigned. However, for a bandwidth of N_p packets, the maximum possible protection level can be assigned is N_p, which actually represents the case that no protection is added to a code-block. So any code-blocks with a protection level larger than N_p is treated as an unprotected code-block. For these unprotected code-blocks, we pack them by their indice they have in the sorted sequence. As this sequence is sorted by their quality contributions, those packets are packed in such a way that more significant packets are packed first.

5.4 Packet loss estimation

In this section, we study the problem of estimating packet losses. The common method to estimate packet losses for a channel is to establish a loss model for the channel and extract the parameters from the channel. These parameters are then used to calculate the distributions of packet losses, which may be used by the applications.

However, as we discussed before, developing a good model for Internet packet loss is a very difficult problem. So in this section, we study a simple approach that uses the previous packet loss data to estimate the future packet loss directly without an underlying model.

We first describe the receiver feedback model and then present out estimation method. This

```
1   INPUT: N_c, N_p, S(i),V(i),P_0(t),P_c(t),P_u
2   OUTPUT: L(i), i = 1..N
3   BEGIN
4       LEVEL = 1;                                  /* Current protection level */
5       i = 1;                                      /* Current code block */
6       WHILE (i ≤ N_c)                             /* Loop over all code blocks */
7           G_old = V(i) * P_c(LEVEL);              /*Gain of current protection
                                                      level */
8           G_new = V(i) * P_c(LEVEL + 1);          /* Bandwidth of current
                                                      protection level */
9           B_old = S(i)/LEVEL;                     /* Gain if protection level
                                                      is increased */
10          B_new = S(i)/(LEVEL + 1);               /* Bandwidth if protection level
                                                      is increased */
11          IF i ≠ 1 THEN
12              IF G_old/B_old ≤ G_new/B_new
13                  change = 1;                     /* If the current code block is
                                                      not the 1st one, increase the
                                                      protection level, as the gain
                                                      per byte is bigger for the
                                                      increased protection level */
14              ELSE
15                  change = 0;                     /* If current code block is not
                                                      the 1st, do not increase
                                                      protection level, as the gain
                                                      per byte is smaller for the
                                                      increased protection level */
16          ELSE
17              IF (1 − P_c(LEVEL + 1) > P_u)       /* If it is the first code block,
                                                      check undec. prob. */
18                  change = 0;
19          IF (change == 1) THEN                   /* Make changes,
                                                      increase protection level */
20              change = 0;
21              LEVEL++;
22          L(i) = LEVEL;
23          i = i + 1;                              /* loop again */
24      END WHILE
25  END
```

Figure 5.5 H-UEP algorithm.

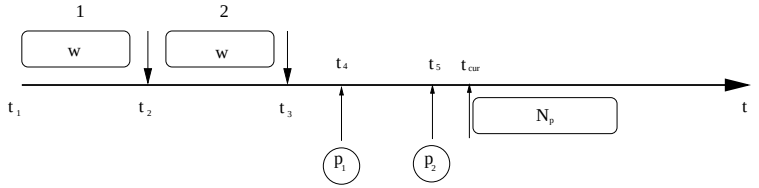


Figure 5.6 Basic feedback model

method is tested on trace-driven simulations and the effectiveness of this method is discussed at last.

5.4.1 Our feedback model

Figure 5.6 shows an example how a receiver performs a feedback. The t axis represents the time. The left two oval boxes stands for two previous batch transmissions 1 and 2, each having w packets. In each batch, the data packets are transmitted at a rate of 30ms per packet. The first(second) batch starts at time $t_1(t_2)$ and ends at $t_2(t_3)$. The right oval box stands for the next batch transmission of N_p packets, starting at t_{cur}, which is the current time. The receiver feedbacks are represented by circles in the figure under the time axis. The two shown fed back packets contain the loss rate of batch transmission 1 and 2, p_1 and p_2, respectively. As there exist transmission delays, the arrival times of these two feedbacks, denoted by t_4 and t_5, are some time later than t_2 and t_3.

Although this figure shows that one feedback is generated for every w data packets, which is an interval of a batch transmission ($30w$ ms), it is usually not necessary to set the feedback interval T_f to be the same as the interval of a batch transmission, in order to avoid too many feedbacks. Typical values for T_f are 20-, 30- and 40-packet intervals, which are equivalent to 0.6, 0.9 or 1.2 seconds in our settings, respectively.

As feedbacks only arrive at an interval of T_f ms, and the time the next batch transmission occurs, t_{cur}, is not deterministic, t_{cur} may not fall exactly at the time when the latest feedback (t_5) arrives at the sender. There is usually a gap $u(t_{cur}) = t_{cur} - t_5$ between them. Therefore, at time t_{cur}, only the feedback at time t_5, as well as feedbacks previous to t_5 is available to the sender. In other words, the sender only knows loss rates of batch transmissions previous to time t_3 at this

point. Considering that there is also loss for feedbacks, a sender might have less information than that when no feedbacks are lost. For instance, if feedback at time t_5 is lost, a sender will only know the loss rates of batch transmissions previous to time t_2.

5.4.2 A simple estimation method

Our problem is to estimate the loss rate $p(t_{cur})$ at time t_{cur} for the next transmission of N_p packets using currently available feedbacks. We first make two assumptions before we discuss our method:

Assumption 5.4: The time gap between the latest available feedback arrival and the next batch transmissions occurance, $u(t_{cur})$, is random, with a uniform distribution between $[0, T_f]$ ms.

Assumption 5.5: Only the latest feedback information will be used in the estimation. We currently do not discuss the strategy of using multiple previous feedbacks in estimation since there is no trivial solution on how to use multiple previous feedbacks. As our focus in this chapter is not on studying estimation methods throughly, we just study a simple case. In the previous example, this assumption means that if the loss rate of batch transmission 2 is known, we will not use the loss rate of batch transmission 1 in the estimation.

Assumption 5.6: On the start of the connection, there may be no feedbacks available. Under such conditions, we take a conservative guess of the loss rate of the next batch transmission, denoted by p_d. A typical value for p_d is 0.3, which is set by the application, and may be changed if *a prior* statistics are available to the application.

Obviously, with the latest feedback arrived at time $t_{cur} - u(t_{cur})$, which contains the loss rate of previous batch transmission starting at time $t_{cur} - u(t_{cur}) - d$, denoted by $p(t_{cur} - u(t_c ur) - d)$, (Here d is the delay between the start of a batch transmission and the receiving of the feedback.), a simple straight-forward estimation of the loss rate of the next batch transmission at time t_{cur}, $p(t_{cur})$ is given by:

$$\hat{p}(t_{cur}) = p(t_{cur} - u(t_{cur}) - d).$$

We define the estimation error for this estimation as:

$$e(t_{cur}) = \hat{p}(t_{cur}) - p(t_{cur}).$$

Note that $e(t_{cur})$ is not only a random variable on t_{cur}, but also depends (implicitly) on d and the type of the connection. The delay d of a feedback can be decomposed into feedback interval T_f and transmission delay (forward path delay for the data packet and the backward path delay for

the feedback), if the feedback is not lost. If feedbacks are lost, there is an additional $T_f$ms delay for each feedback loss. The transmission delay is a random variable on t_{cur} and is also determined by the type of connection, which an application cannot change. The application can only change T_f. Hence, the effects of T_f as well as the type of connection on $e(t_{cur})$ should be studied.

We use a trace-driven simulation to obtain the distribution of $e(t_{cur})$. The size of batch transmission w is fixed in each run of the simulation, as well as the size of next batch transmission N_p. They are set to the same value in the simulation, which is not necessary in general cases.

The sender first starts at time t_0 by sending multiple batch transmissions, and waiting for the feedbacks, of which the first is expected to arrive at time t^*. After t^* is decided using the trace data, a random time gap u is generated (uniformly), and $t_{cur} = t^* + u$ is set as the time for the next batch transmission. The estimation from feedback and the real loss rate is obtained and $e(t_{cur})$ is calculated. The sender stops sending packets 10 seconds after it has started. This is to simulate that a user will jump from one Web server to another Web server from time to time. This value may be changed if more accurate user behavior is known, but currently we heuristically set it to 10 seconds. In the simulation, if the first feedback is lost, we use p_d as stated in **Assumption 5.6**.

During one run of the simulation, all estimation errors are recorded for future analyses. We perform this simulation over the three destinations we chose in Chapter 3 and the results of our estimation is shown in the next subsection.

5.4.3 Distributions of estimation error

In this subsection, we study the distributions of estimation error e obtained in the simulation described in the previous subsection. In our simulation, we test different feedback intervals, different batch sizes and different traces that have different delays and losses. The feedback interval, T_f, is set at 20, 30 or 40 packets respectively and the traces we use are those discussed in Chapter 3. The batch size, w, is set as 8, 16 or 32, which corresponds to the bitrate of 0.125bpp, 0.25bpp or 0.5bpp, respectively. For each trace, we fix the feedback interval and batch size before we take 1000 runs of the simulation to obtain the distribution of e.

Figures 5.7 to 5.9 show the estimation error's cumulative probability distributions (CDF), $P(\alpha)$, which is defined as:

$$P(\alpha) = Prob\{\hat{p}(t) \in [p(t) - \alpha, p(t) + \alpha]\} = Prob\{|e| \leq \alpha\}.$$

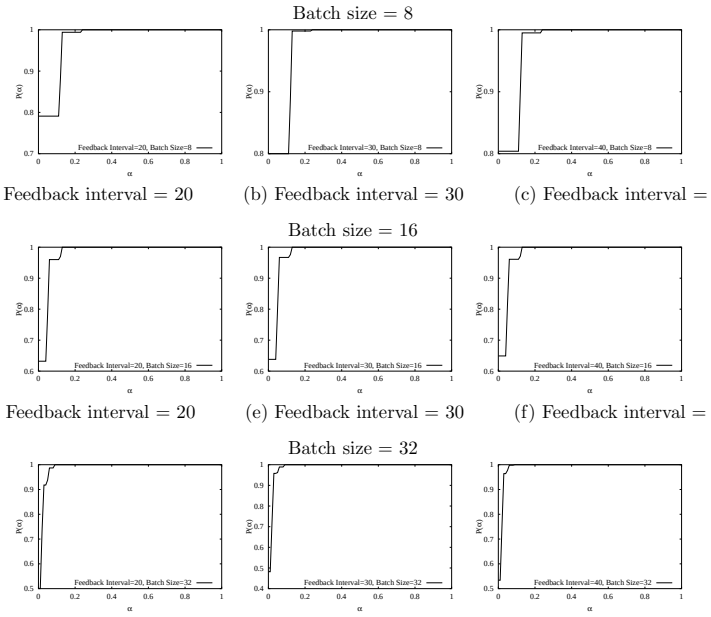
Figures 5.7 shows $P(\alpha)$ for UIUC-Taiwan connection, which has an average loss of 5% and an average on-way delay of 338ms. From the results, we first observe that the feedback interval does not affect the error distribution very much. This is true because that the network may be quite stable within a period of time. Although changing the feedback interval affects the total delay d between the time when past statistics were collected and the current time, it will not affect the estimation accuracy too much as long as d is within a certain range.

The second observation we can make is that the batch size has a significant effect on estimation accuracy. For a small batch size like 8, we can see that the probability of correct estimation $P(0)$ is high (around 0.8 in the figure), but it approaches 1 slowly. As the batch size increases, we can see that although the probability of correct estimation becomes smaller, $P(\alpha)$ quickly goes to 1. This is not surprising because it is more difficult to estimate the exact number of packets that will be lost for a larger batch size. Also, as batch size increases, the transient fluctuations of loss rate in small intervals will compensate each other and the average loss rate in this large batch will behave more stable; hence, it is easier to make an estimation with small errors in this case. That's why $P(\alpha)$ approaches to 1 quickly for large batch sizes.

For UIUC-Thailand (average loss 15% and average delay 384ms) and UIUC-Argentina (average loss 35% and average delay 487ms) connections, we observe similar results of $P(\alpha)$ in Figures 5.8 and 5.9, for both connections individually.

Besides that, we also find out that the type of connections is an important factor that can affect the error distributions. For the low-loss connection, $P(0)$ is large and $P(\alpha)$ approaches 1 quickly. As the loss rate increases, $P(0)$ becomes smaller and the speed of $P(\alpha)$ approaching 1 also becomes slower. This indicates that our estimation method will become less effective when loss rate increases. This can be explained qualitatively by multiple hidden-state model for the underlying network. When loss rates are higher, the network switches more frequently from "good" (non-congested) state to "bad" (congested) state, which makes the network's behavior less stationary and it is more difficult to estimate the loss rate under such conditions.

In the next section of experimental results, we will examine the sensitivity of our algorithm based on these error distributions.



(a) Feedback interval = 20 (b) Feedback interval = 30 (c) Feedback interval = 40

(d) Feedback interval = 20 (e) Feedback interval = 30 (f) Feedback interval = 40

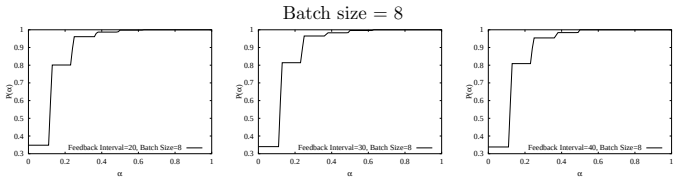
(g) Feedback interval = 20 (h) Feedback interval = 30 (i) Feedback interval = 40

Figure 5.7 $P(\alpha)$ for UIUC-Taiwan connection for three different batch sizes and feedback intervals.

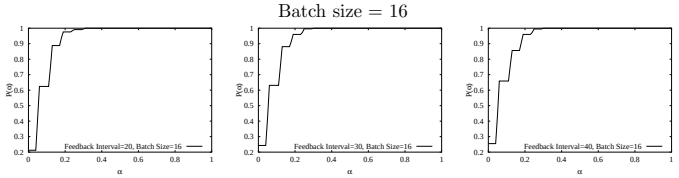
5.5 Experimental Results

Before we give our experimental results, we first discuss the metrics used in measuring performance. First, we use the common objective measure PSNR. Since we cannot list PSNR under all possible loss scenarios, we use the average PSNR over all possible cases. For each scenario, the PSNR calculated is weighted by the probability of the scenario. A second metric used in the undecodable probability of each packing algorithm.

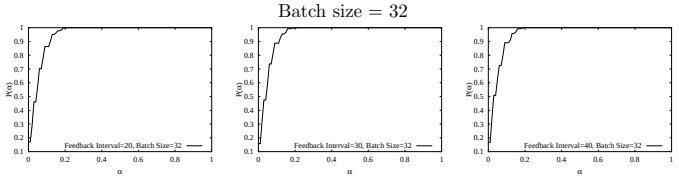
It is well known that packet losses in Internet are very difficult to model. Hence, we have chosen two loss scenarios to test our method. In the first scenario, we assume an independent packet loss model and generate all possible loss patterns given an average loss probability p. Our method is then tested under different average loss probability p. In the second scenario, we test our method



(a) Feedback interval = 20 (b) Feedback interval = 30 (c) Feedback interval = 40



(d) Feedback interval = 20 (e) Feedback interval = 30 (f) Feedback interval = 40



(g) Feedback interval = 20 (h) Feedback interval = 30 (i) Feedback interval = 40

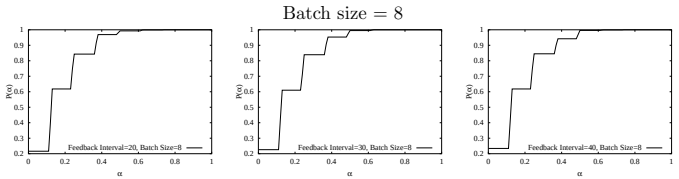
Figure 5.8 $P(\alpha)$ for UIUC-Thailand connection for three different batch sizes and feedback intervals.

using traces collected in the real Internet connections. This can give the performance of our method in more realistic cases. Based on a sensitivity test of our algorithm on an estimated average loss rate p, we use a simple method to estimate p from receiver feedbacks.

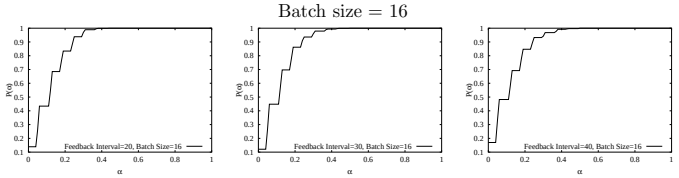
5.5.1 Artificial loss scenarios

In this subsection, we show the performance of serial packing (S-UEP), parallel packing (P-UEP) and our proposed hybrid packing (H-UEP) algorithm in artificial loss scenarios. We first introduce the concept of *loss patterns* and explain how the average PSNR is measured.

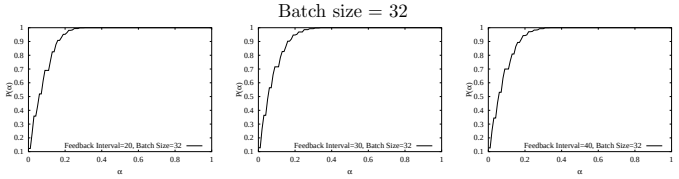
Definition 5.4: A *loss pattern* V of a batch of packets transmitted is a binary sequence that describes whether a specific packet is received or lost, where 1 denotes a received packet and 0



(a) Feedback interval = 20 (b) Feedback interval = 30 (c) Feedback interval = 40



(d) Feedback interval = 20 (e) Feedback interval = 30 (f) Feedback interval = 40

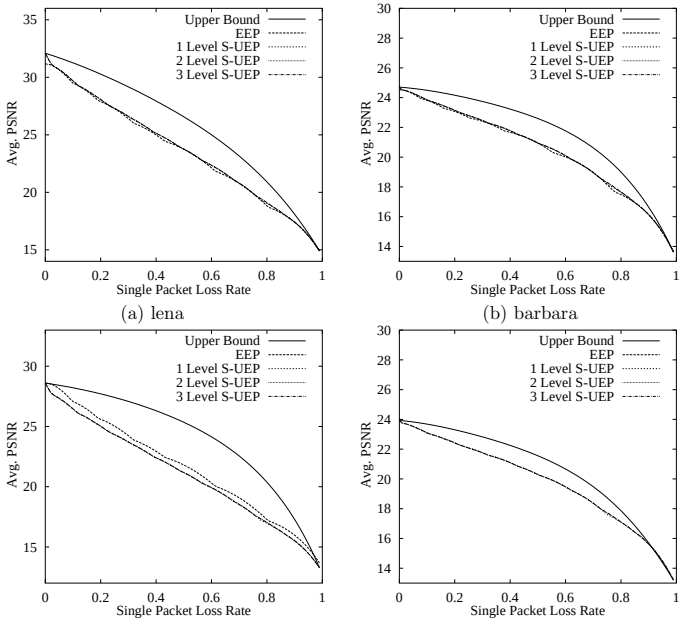


(g) Feedback interval = 20 (h) Feedback interval = 30 (i) Feedback interval = 40

Figure 5.9 $P(\alpha)$ for UIUC-Argentina connection for three different batch sizes and feedback intervals.

denotes a lost packet. For instance, a lost pattern $V = (11110110)$ means during a transmission of 8 packets, the fifth and the eighth packets are lost while other packets are received.

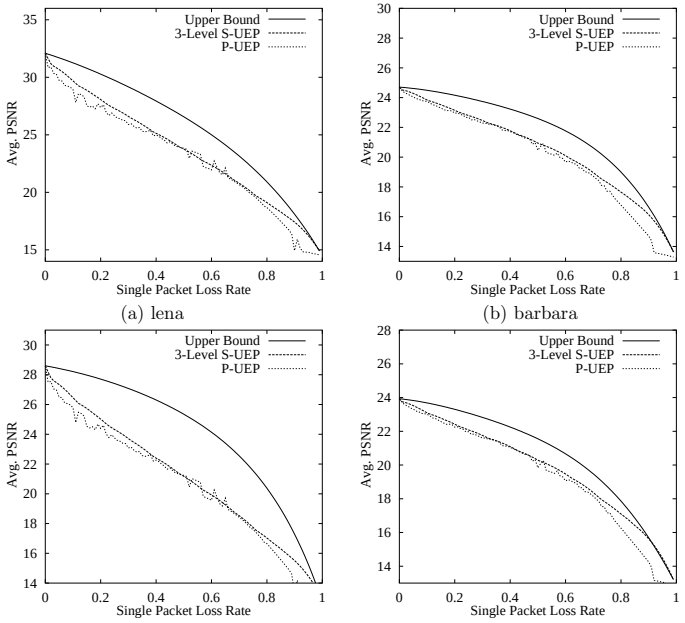
Given an average packet loss rate p and a loss pattern V, it is easy to calculate the probability that this pattern V will occur in transmission: $Pr(V) = p^{n_0}(1-p)^{n_1}$, where n_0 and n_1 are the number of 0's and 1's in V. For a certain packing algorithm, we first calculate its PSNR, denoted by $PSNR(V)$ under loss pattern V, and then obtain an average PSNR by finding $E[PSNR(V)] = \sum_{V in \mathcal{V}} PSNR(V)Pr(V)$, where $\mathcal{V}$ is the set of all possible loss patterns.



(a) lena (b) barbara

(c) smoke (d) cloth

Figure 5.10 Average PSNR of S-UEP with different levels of protection, compared with the upper bound and that of EEP.



(a) lena (b) barbara

(c) smoke (d) cloth

Figure 5.11 Average PSNR of P-UEP optimized by CSA, compared with the upper bound and that of 3 level S-UEP.

5.5.1.1 Average PSNR

We first demonstrate the performance of S-UEP for *lena*, *barbara*, *cloth* and *smoke*, all compressed at 0.125 bpp. The compressed image will be transmitted in 8 UDP packets, which is fixed over all schemes.

To obtain an upper bound of all UEP methods, we calculate the received image quality assuming the sender knows the specific packets lost before sending them. Under this assumption, the sender will only send a packet when it will not be lost. This assumption is, of course, not feasible in practice, but can provide an upper bound on quality.

We also include the performance of an EEP scheme. For both EEP and S-UEP, we enumerate all possible ways of placing FEC codes in the allowed bandwidth and obtain the optimal performance.

Figure 5.10 shows the average PSNR of S-UEP compared to that of EEP, where the x axis is the average packet loss rate p and the y axis is the average PSNR. For S-UEP, we have enumerated several levels of protections.

The results show that the performance improvement of S-UEP to EEP is not very significant, and S-UEP with multi-level protection does not lead to improved performance.

Next, we demonstrate the performance of P-UEP for the same images with the same compression rate in Figure 5.11. In this experiment, we find the packing for P-UEP using CSA [27].

In the figure, the upper bound is calculated in the same way as that in the experiment on S-UEP, and x and y axes also carry the same meaning as those in Figure 5.10. In order to compare the performance of P-UEP with S-UEP, we have included in Figure 5.11 the performance of the 3-level S-UEP scheme, as this gives the best performance of all multi-level S-UEP schemes.

In Figure 5.11, we observe that the performance of the 3-level S-UEP and P-UEP are very close. It is difficult to tell which one outperforms the other in terms of average PSNR. However, as the result of P-UEP was obtained by CSA, its computation cost of P-UEP is certainly much higher than that of S-UEP.

Next, we demonstrate the performance of our proposed H-UEP in Figure 5.12. The upper bound is calculated in the same way as in previous experiments, and the x and y axes are also the same as before. The packing of each code block is determined by our heuristic described in Figure 5.5. To compare our heuristic with other two packing methods, we include in Figure 5.12 the performance of S-UEP and CSA-optimized P-UEP.

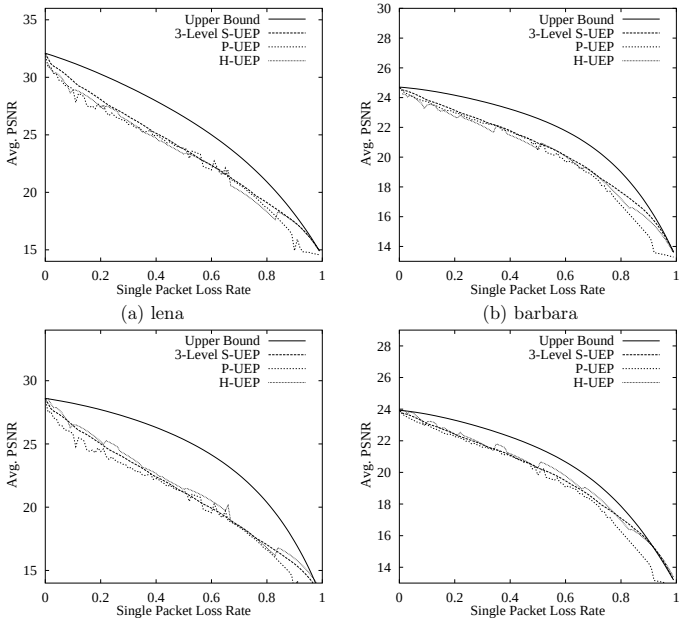
In this figure, we see that the performance of H-UEP does not degrade significantly from that of enumerated S-UEP and CSA-optimized P-UEP. However, the CPU time for enumerated S-UEP is about 5 minutes and the CSA optimization usually takes more than 2 hours, whereas the CPU time for H-UEP is less than 0.1 seconds. Considering that the high computation costs of both enumerated S-UEP and CSA-optimized P-UEP and the efficient H-UEP, our proposed heuristic of H-UEP is a more practical approach with good quality.

5.5.1.2 Undecodable probability

We now study the undecodable probability of S-UEP, P-UEP and our proposed H-UEP.

First, we show the results of S-UEP in Figure 5.13. The undecodable probability of EEP is also shown in this figure as a comparison. The test images are *lena*, *barbara*, *smoke* and *cloth*, all compressed at 0.125 bpp. The x axis is the average loss rate p, and the y axis is the undecodable probability.

From this figure, we observe that the undecodable probability of S-UEP does not improve very much from that of EEP. Both will have an undecodable probability that goes very high when the average loss rate p exceeds 0.4. This is obviously not preferred in real applications. Next, we show the undecodable probability of P-UEP under the same situations in Figure 5.14. As before, we include the undecodable probability of S-UEP for a comparison. The x and y axis are the same as those in the previous subsection.



(a) lena (b) barbara

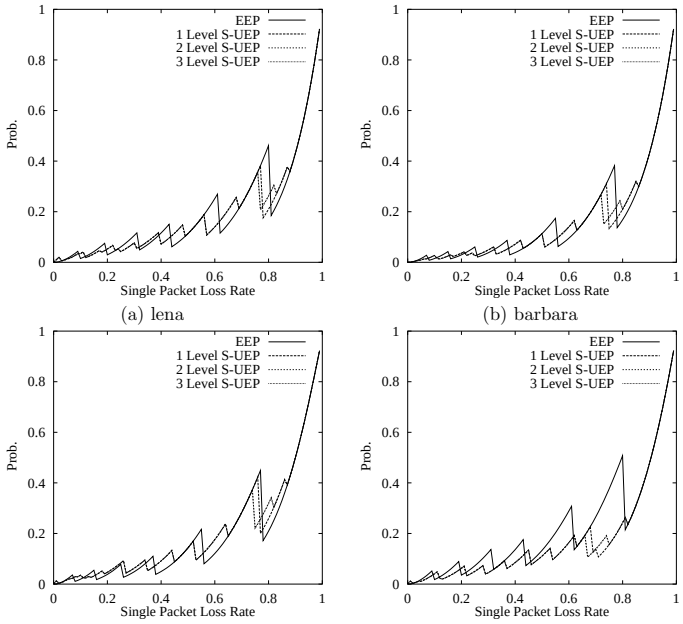
(c) smoke (d) cloth

Figure 5.12 Average PSNR of our proposed H-UEP, compared with the upper bound, that of 3-level S-UEP and that of P-UEP.

From this figure, it is obvious that P-UEP can improve a lot in undecodable probability than EEP and 3-level S-UEP. Especially when p exceeds 0.4, the total undecodable probability is still very low.

Now we show the undecodable probability of our proposed H-UEP in Figure 5.15. To further compare it with S-UEP and P-UEP, we include in this figure the performance of the 3-level S-UEP and CSA-optimized P-UEP methods. The x and y axes carry the same meaning as before.

In this figure, it is not surprising to see that H-UEP performs in the same way as P-UEP does in terms of undecodable probability. This is because in our H-UEP algorithm, it use P-UEP algorithm in packing the first several code blocks. As the undecodable probability is determined by



(a) lena (b) barbara

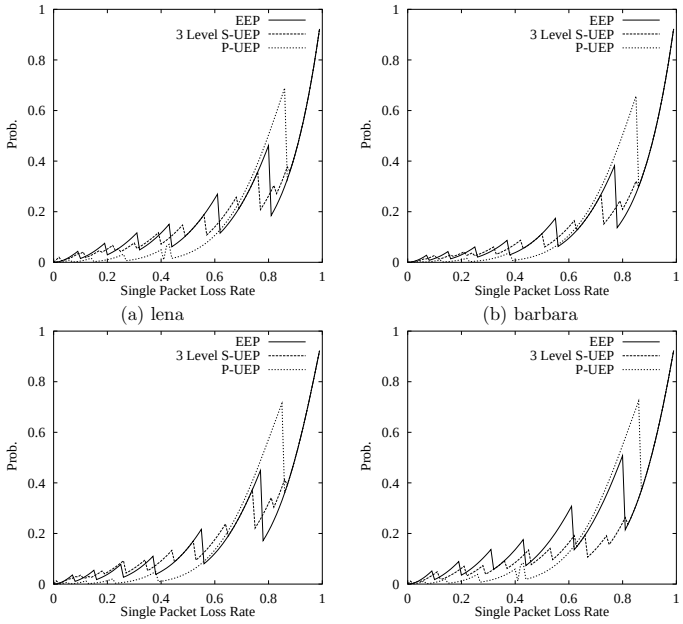
(c) smoke (d) cloth

Figure 5.13 Undecodable probability of S-UEP with different protection levels, compared with that of EEP.

the protection level of the first code block, H-UEP and P-UEP have the same results. When the loss rate goes very high, the undecodable probability of H-UEP differs a little from that of the P-UEP. This is because that in our proposed H-UEP algorithm, the protection level for a code-block is always increased as its index increases, whereas in the CSA-optimized P-UEP, the protection level is arbitrarily searched for a code-block. This has a subtle impact on the final protection level assignment of the first code-block, and therefore, the undecodable probability may differ a little.

5.5.2 Experiments using trace-driven simulations

In this section, we test the performance of our proposed H-UEP method in the trace-driven experiments. We use the traces collected from real Internet connections described in Chapter 3.



(a) lena (b) barbara

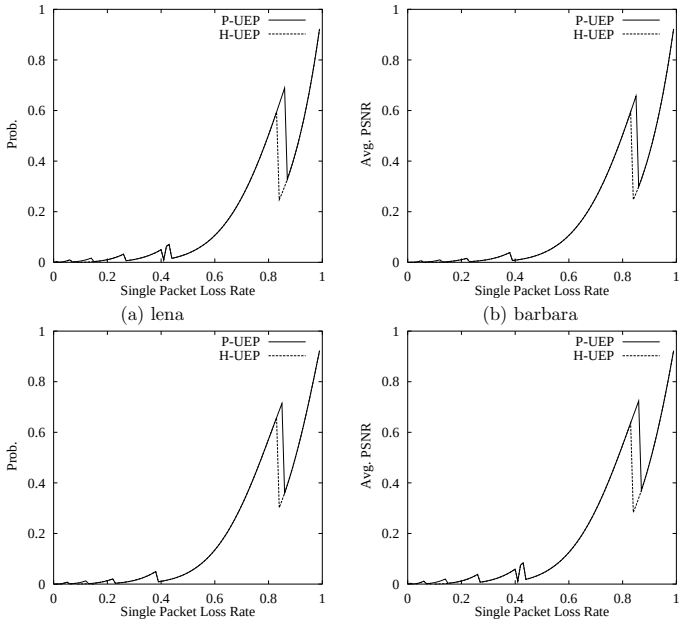
(c) smoke (d) cloth

Figure 5.14 Undecodable probability of P-UEP, compared with that of EEP and 3-level S-UEP.

Before testing the performance of our proposed H-UEP method, we first test the sensitivity of H-UEP based on the estimation error distributions we have obtained in the previous section. After that, we test H-UEP by using our simple estimation method with the three traces and discuss the results.

5.5.2.1 Sensitivity analysis

From the distributions we have obtained from the previous section, we know that our simple estimation method produces errors whose distribution may depend on several factors, such as batch size, feedback interval and type of connection. Therefore, we want to test the sensitivity of H-UEP under the situation that the loss rate is estimated with errors.



(a) lena (b) barbara

(c) smoke (d) cloth

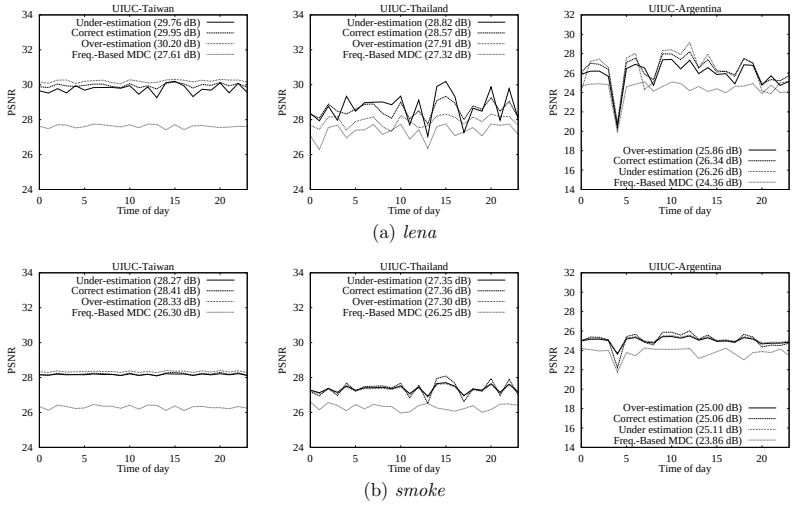
Figure 5.15 Undecodable probability of H-UEP, compared with that of P-UEP.

In each experiment, we feed H-UEP with three loss rates, each in one simulation: $p(t_0)$, $p_o(t_0) = \min\{p(t_0) + \alpha_0, 1\}$, and $p_u(t_0) = \max\{p(t_0) - \alpha_0, 0\}$. Here, $p(t_0)$ is the exact loss rate in the next batch transmissions at time t_0 in the trace, and $p_o(t_0)$ and $p_u(t_0)$ are the over- and under-estimated loss rates we are to test. We choose different α_0 for different connection, whose value satisfies:

$$P(\alpha_0) > 0.85.$$

For instance, with a batch size of 8 and a feedback interval of 30, the values of α_0 are 0.15, 0.20 and 0.30 for UIUC-Taiwan, UIUC-Thailand and UIUC-Argentina, respectively.

For each connection, we fix the bit-rate at 0.125 bpp, which corresponds to a batch size of 8. As we previously discussed, the error distribution approaches 1 more slowly when the batch



(a) *lena*

(b) *smoke*

Figure 5.16 Sensitivity analysis of sending *lena* and *smoke* to three different sites. Both images are encoded at 0.125 bpp.

size is small; hence, batch size of 8 is the worst case our estimation method may perform in our experiment. The feedback delay is fixed at 30, which means we generate a feedback in about 0.9 seconds.

We have tested these three cases for the three connections we described in Chapter 3, and Figure 5.16 shows the results for *lena* and *smoke*. We can see that even if the sender has an error in estimation, the performance of H-UEP will only degrade a small extent instead of dropping dramatically. The degradation is usually less than 1 dB, but is still better than that of the MDC+UDP case. With this, we conclude that H-UEP is robust to the estimation errors and we will carry out our simple estimation method in the trace-driven experiments discussed in the next subsection.

5.5.2.2 24-hour performance

Based on traces collected and pre-stored, we have tested our proposed H-UEP scheme. The performance of our algorithm under artificial loss scenarios cannot reflect the real performance of our

algorithm, because the models used to generate the artificial cases may be simple and inaccurate. As there is no good model to generate accurate Internet losses, we choose to test our algorithm using trace-driven simulations.

In our experiments, we use collected trace data from three connections and applied the proposed H-UEP packing algorithm. The three connections have an average loss rate p of about 5%, 15% and 35%, respectively.

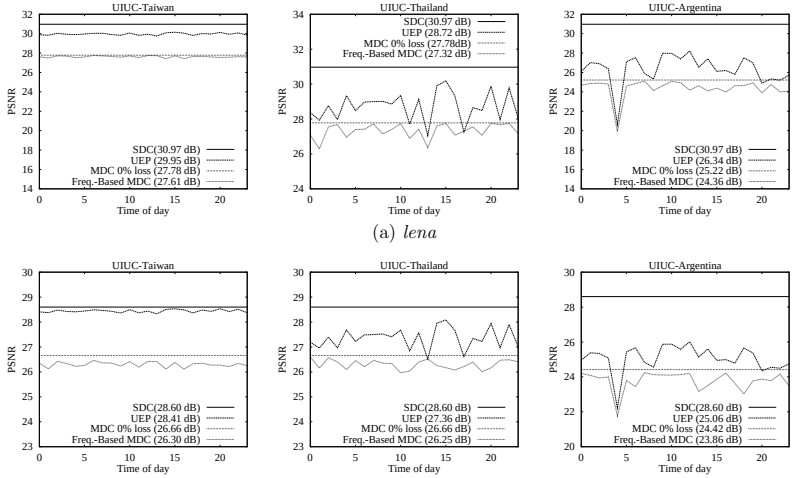
The test images are *lena* and *smoke*, both compressed at 0.125 bpp into 8 UDP packets. During the beginning of each hour, 250 batches of 8 UDP packets are transmitted, and the PSNR is computed from the packets received in the trace data. The feedback model is the same as we discuss in Section 5.4. The loss rate for the next batch transmission is obtained from a feedback of previous batch transmissions with a delay. For every 10 seconds, we assume that a new connection has started and the first several feedbacks are not available at the beginning of a new connection. For these cases, we use $\hat{p} = 0.3$ as a conservative guess.

Figure 5.17 shows the experimental results, as well as the performance of TCP+SDC, TCP+MDC, and the proposed frequency-based MDC. In the first connection, our proposed H-UEP method outperforms TCP+MDC for both *lena* and *smoke*. Further, there is a 1 to 2 dB increase in average PSNR as compared to the proposed frequency-based MDC. The result of H-UEP is also very close that of TCP+SDC, which is actually considered the upper bound of the performance. In the second and the third connections, H-UEP shows similar improvements over TCP+MDC in both natural and textural images.

To better illustrate the performance of our proposed H-UEP, Figures 5.18 and 5.19 show the results of two transmitted images to Thailand and Argentina using three different methods. It shows that either UEP+MDC or MDC+UDP provides a good image quality and a shorter delay. For UEP+MDC, it even provides a better image quality with the same delay, compared with that of MDC+UDP.

5.6 Summary

In this chapter, we first review the drawbacks of the MDC approach described in Chapter 4. After analysing our problem in terms of unequal error protection and stating our assumptions clearly, we have explored three major components of UEP schemes and concentrate on studying packing



(a) *lena*

(b) *smoke*

Figure 5.17 Twenty-four hour performance of sending image *lena* and *smoke* compressed at 0.125 bpp to three different sites.

algorithms. Two simple packing algorithms, namely, serial and parallel packing algorithms, are first studied and then we have proposed a hybrid packing algorithm that further takes advantages of both serial and parallel packing algorithms. In addition, we have proposed a simple estimation method that can estimate the packet losses from the receiver feedbacks, which is useful to our packing algorithm.

Our proposed algorithm is tested in both artificial loss scenarios and trace-driven simulations and its performance is better than the previous proposed MDC under low- and medium-loss situations and is similar to that of MDC in high loss situations. At the end of this chapter, we have studied the sensitivity of our proposed algorithm when the packet loss can not be correctly estimated in practice. Our experimental results show that our proposed algorithm can work well when the estimated average packet loss rate is not very accurate.

In general, a UEP scheme with a known distribution of packet losses can perform better than the MDC scheme as it can adjust the redundancy it adds to the code stream based on the distri-

(a) SDC by TCP (b) MDC by UDP (c) UEP by UDP

Figure 5.18 Quality-delay trade-offs in round-trip transmissions of *lena* compressed at 0.125 bpp by JPEG2000 between UIUC and Thailand. In UDP transmissions, two out of the eight packets were lost. Image (a) has a quality of 30.97 dB and a delay of 4.01 sec. Image (b) has a quality of 25.21 dB and a delay of 0.71 sec. Image (c) has a quality of 26.03 dB and a delay of 0.71 sec.

(a) SDC by TCP (b) MDC by UDP (c) UEP by UDP

Figure 5.19 Quality-delay trade-offs in round-trip transmissions of *smoke* compressed at 0.25 bpp by JPEG2000 between UIUC and Argentina. In UDP transmissions, five out of the sixteen packets were lost. Image (a) has a quality of 30.96 dB and a delay of 13.03 sec. Image (b) has a quality of 28.72 dB and a delay of 0.46 sec. Image (c) has a quality of 29.45 dB and a delay of 0.46 sec.

bution. However, this approach relies heavily on the packet loss distribution and its performance is also affected by the accuracy of the estimation model, whereas the MDC scheme does not need any information on packet losses. So UEP might be a better alternative than MDC for a network that has a slowly-changing behavior.

6 CONCLUSIONS AND FUTURE WORK

6.1 Conclusions

This thesis addresses the problem of protecting JPEG 2000 images during transmission over the Internet by both MDC and FEC approaches. First, we have studied the Internet loss bahavior. A new frequency-based reconstruction MDC algorithm has been proposed, and its performance is compared with that of previous methods. We have further shown the quality-delay trade-offs of this algorithm and have justified that MDC+UDP can be a viable for transmitting images over the Internet.

We have also studied UEP schemes for protecting real-time image transmissions under different loss conditions. Our experimental results show that UEP can also achieve good quality in protecting against losses.

In conclusion, with our proposed error-concealment algorithms, image delivery with UDP can be an attractive alternative to traditional TCP delivery.

6.2 Future Work

This research can be improved in better understanding and assessment of the channel behavior of the Internet. Our results have shown that the accuracy of the channel model is a key factor in enhancing UEP and can even help in the case of MDC schemes. In the future, we plan to analyze packet traces from real Internet connections. With more accurate prediction of channel losses, we expect to achieve better performance in both MDC and UEP.

REFERENCES

[1] A. E. Mohr, E. A. Riskin, and R. E. Ladner, "Unequal loss protection: Graceful degradation of image quality over packet erasure channels through forward error correction," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 6, pp. 819–828, 2000.

[2] S. D. Servetto, K. Ramchandran, V. A. Vaishampayan, and K. Nahrstedt, "Multiple description wavelet based image coding," *IEEE Transations on Image Processing*, vol. 9, no. 5, pp. 813–826, 2000.

[3] G. Yu, M. M. Liu, and M. W. Marcellin, "POCS-based error concealment for packet video using multiframe overlap information," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 8, no. 4, pp. 422–434, 1998.

[4] S. S. Hermami and R. M. Gray, "Subband coded image reconstruction for lossy packet networks," in *Conf. Record of the 28th Asilomar Conf. on Signals, Systems and Computers*, 1995, pp. 487–491.

[5] W. Kwok and H. Sun, "Multi-directional interpolation for spatial error concealment," *IEEE Transactions on Consumer Electronics*, vol. 39, no. 3, pp. 455–460, 1993.

[6] J. Suh and Y. Ho, "Error concealment based on directional interpolation," *IEEE Transactions on Consumer Electronics*, vol. 43, no. 3, pp. 295–302, 1997.

[7] W. Zeng and B. Liu, "Geometric-structure-based error concealment with novel application in block-based low-bit-rate coding," *IEEE Transactions on Circuit and System for Video Technology*, vol. 9, no. 4, pp. 648–6652, 1999.

[8] V. K. Goyal, "Multiple description coding: Compression meets the network," *IEEE Signal Processing Magazine*, vol. 18, no. 6, pp. 74–93, 2001.

[9] V. A. Vaishampayan, "Design of multiple description scalar quantizers," *IEEE Transactions on Information Theory*, vol. 39, no. 3, pp. 821–834, 1993.

[10] J. Cardinal, "Entropy-constrained index assignments for multiple description coding quantizers," *IEEE Transactions on Signal Processing*, vol. 52, pp. 265–270, Jan. 2004.

[11] Y. Wang, M. T. Orchard, V. Vaishampayan, and A. R. Reibman, "Multiple description coding using pairwise correlation transforms," *IEEE Transactions on Image Processing*, vol. 10, no. 3, pp. 351–366, 2001.

[12] V. K. Goyal and J. Kovacevic, "Generalized multiple description coding with correlating transforms," *IEEE Transactions on Information Theory*, vol. 47, pp. 2199–2224, Sept. 2001.

[13] H. Chen and Y. Yang, "Multiple description based wavelet image coding," *Proceedings of the Fourth International Conference on Parallel and Distributed Computing, Applications and Technologies*, pp. 805–808, 2003.

[14] S. Tian and P. Rajan, "Multiple description coding using transforms and data fusion," *International Workshop on Digitial and Computational Video*, pp. 192–199, Nov. 2002.

[15] I. Bajic and J. Woods, "Domain-based multiple description coding of images and video," *IEEE Transactions on Image Processing*, vol. 12, pp. 1211–1225, Oct. 2003.

[16] D. Lin and B. W. Wah, "LSP-based multiple-description coding for real-time low bit-rate voice over IP," *IEEE Transaction on Multimedia*, 2003, to be published.

[17] X. Su and B. W. Wah, "Reconstruction-based subband image coding for UDP transmissions over the Internet," *J. of VLSI Signal Processing*, vol. 34, no. 1-2, pp. 29–48, 2003.

[18] G. Wallace, "The jpeg still picture compression standard," *Communications of the ACM*, p. 35, 1991.

[19] ISO, *JPEG2000 Image Coding System, ISO IS 15444 Part I.* 2000.

[20] D. Taubman and M. Marcellin, *JPEG2000 Image compression fundamentals, standards and proactice*. Kluwer Academic Pub., 2002.

[21] R. Cook and S. Weisberg, *Applied Regression Including Computing and Graphics*. New York Wiley, 1999.

[22] J. Kim, R. Mersereau, and Y. Altunbasak, "Error-resilient image and video transmission over the internet using unequal error protection," *IEEE Transactions on Image Processing*, vol. 12, pp. 121–131, Feb. 2003.

[23] T. Fingscheidt, T. Hindelang, R. Cox, and N. Seshadri, "Joint source-channel (de-)coding for mobile communications," *IEEE Transactions on Communications*, vol. 50, pp. 200–212, Feb. 2002.

[24] C. Tanriover and B. Honary, "Image transmission using unequal error protection combined with two-fold turbo coding," *Joint First Workshop on Mobile Future and Symposium on Trends in Communications*, pp. i–v, 2003.

[25] B. Pettijohn, M. Hoffman, and K. Saywood, "Joint source/channel coding using arithmetic codes," *IEEE Transactions on Communications*, vol. 49, pp. 826–836, May 2004.

[26] A. Natu and D. Taubman, "Unequal protection of jpeg 2000 code-streams in wireless channels," *IEEE Global Telecommunications Conference*, vol. 1, pp. 534–538, Nov. 2002.

[27] T. Wang, *Global Optimization for Constrained Nonlinear Programming*. Urbana, IL: Ph.D. Thesis, Dept. of Computer Science, Univ. of Illinois, Dec. 2000.