JOINT SOURCE/CHANNEL CODING FOR IMAGE AND VIDEO TRANSMISSION

by

Zhenyu Wu

A Dissertation Submitted to the Faculty of the

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

In Partial Fulfillment of the Requirements
For the Degree of

DOCTOR OF PHILOSOPHY

In the Graduate College

THE UNIVERSITY OF ARIZONA

2005

INFORMATION TO USERS

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleed-through, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

ProQuest Information and Learning Company
300 North Zeeb Road
P.O. Box 1346
Ann Arbor, MI 48106-1346

The University of Arizona ®
Graduate College

As members of the Final Examination Committee, we certify that we have read the dissertation prepared by **Zhenyu Wu**

entitled **Joint Source/Channel Coding For Image And Video Transmission**

and recommend that it be accepted as fulfilling the dissertation requirement for the Degree of **Doctor of Philosophy**

Michael W. Marcellin, Ph.D.	12/9/04 date
Ali Bilgin, Ph.D.	12/9/2004 date
William E. Ryan, Ph.D.	12/9/04 date
John N. Palmer, Ph.D.	12/9/04 date
	date

Final approval and acceptance of this dissertation is contingent upon the candidate's submission of the final copies of the dissertation to the Graduate College.

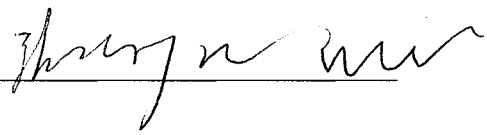
I hereby certify that I have read this dissertation prepared under my direction and recommend that it be accepted as fulfilling the dissertation requirement.

Dissertation Director: **Michael W. Marcellin, Ph.D.** 12/9/04 date

STATEMENT BY AUTHOR

This dissertation has been submitted in partial fulfillment of requirements for an advanced degree at The University of Arizona and is deposited in the University Library to be made available to borrowers under rules of the Library.

Brief quotations from this dissertation are allowable without special permission, provided that accurate acknowledgment of source is made. Requests for permission for extended quotation from or reproduction of this manuscript in whole or in part may be granted by the copyright holder.

SIGNED: 

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude towards my dissertation advisor, Dr. Michael W. Marcellin. It is him that gave me this interesting but challenging topic at the beginning and guided me into the research area. Throughout my study, his insightful thoughts have helped me overcome numerous difficulties. Without his support and guidance, this dissertation would not have been possible.

I would also like to express my appreciation to Dr. Ryan and Dr. Vasic. They provide me with helps and suggestions on many of the channel coding problems I encountered in the dissertation.

I would like to thank for Dr. Ryan, Dr. Bilgin, and Dr. Palmer for accepting to be members of my dissertation committee.

I am indebted to the group members of the SPACL lab. I especially owe much to Ali Bilgin for helping me get familiar with JPEG2000 and many useful discussions.

I would also like to thank Tami Whelan for handling all the paperwork necessary for the completion of my degree.

Finally, I would like to thank my parents. It is their endless and unconditional love that encourages me all along. This dissertation is dedicated to them.

To my parents,

Yaoming Wu and Wanzhen Shao

TABLE OF CONTENTS

TABLE OF CONTENTS, Continued

TABLE OF CONTENTS, Continued

LIST OF FIGURES

LIST OF FIGURES, Continued

LIST OF TABLES

LIST OF TABLES, Continued

ABSTRACT

With the rapid growth of data communication infrastructure, there has been an increasing demand for multimedia communication services over the past years. This rapid growth has posed new requirements and challenges in many related research areas. This dissertation studies the problem of efficient and robust transmission of image and video content over noisy channels. Firstly, joint source/channel coding algorithms are proposed for image transmission based on the new image coding standard JPEG2000. By combining the forward error correction capability provided by channel coding, together with the scalability and error resilience (ER) features provided by JPEG2000, the algorithms can achieve unequal error protection gains as well as robust source decoding for single image transmission over band-limited channels. Secondly, joint source/channel coding algorithms for multiple sources transmission over a common channel sharing a given total bandwidth are proposed. The algorithms exploit the rate-distortion diversity among multiple sources to optimally distribute a total bit rate given by the channel. It is demonstrated that both improved quality and reduced quality variance can be achieved at the receiver by the proposed algorithms. Finally, a robust image transmission scheme is proposed for packet erasure channels. The algorithm is based on the ER functionalities provided by JPEG2000

for robust source decoding. Together with the proposed interleaving scheme, some erasures can be recovered without channel coding.

CHAPTER 1

INTRODUCTION

1.1 Motivation

With the explosive growth of the technology and deployment of data communication infrastructure, there has been an increasing demand for digital multimedia communication services over the past years. Multimedia data, including text, audio, image, video and so forth, are being spread and transmitted at unprecedented speed, scope and convenience around the world. For example, the Internet has become a pervasive means for disseminating multimedia data. Meanwhile, mobile communications are emerging from primarily voice-based services toward data-oriented services, giving rise to a new, fast-growing wireless multimedia industry.

Communication channels are usually limited in bandwidth, and multimedia sources usually contain a significant amount of redundancy (such as images and video). Therefore data compression, also called *source coding*, is an indispensable component for efficient transmission of such information. Source coding removes source redundancy by using techniques such as transform coding and predictive coding, which exploit spatial, temporal, or spatio-temporal correlations among samples. While redundancy is reduced, data dependency is usually created among the bits from a coded

source bitstream. This leads to the vulnerability of source bitstreams to transmission errors. When such a bitstream is transmitted over an error-prone channel, a single bit error from the channel could derail the source decoder and thus render the whole bitstream useless. To combat errors introduced by noisy channels, *channel coding* is often employed to add controlled redundancy. Removing redundancy by source coding and adding redundancy by channel coding are two apparently conflicting goals. Therefore in designing a multimedia communication system, questions arise as: how much redundancy is necessary, and how to effectively add the redundancy.

We can design such a communication system as a combination of two parts, i.e., source coding and channel coding. Source codes are designed to represent source information in the most efficient form. Channel codes are designed separately and independently to approach the channel capacity. Shannon's separation theorem [1] says that the combination will be as efficient as anything that we could design by considering both problems together.

The separation theorem applies only in the information-theoretic sense: as long as the entropy of the source is less than the channel capacity, there exist a separable source and a channel coding scheme that allows transmission over the channel with arbitrarily low error probability. The theorem assumes infinitely long codewords, infinite coding complexity and infinite delay. Furthermore, it requires that the channel state be known to the transmitter prior to encoding. Also the separation theorem is applicable only in point-to-point communication.

For most practical systems, however, these assumptions cannot be fulfilled. For example, for applications with real-time requirements, long delay is not tolerable. For applications that are complexity conscious, complicated coding schemes are prohibitive. Meanwhile, communication channels such as those experienced in the Internet and wireless networks are characterized by constant changes. It is difficult to precisely predict the channel conditions prior to and during transmission. Moreover, for channels suffering extensive noise, interference, fading and shadowing, it is often too expensive in terms of bandwidth and delay to implement perfect channel decoding. Finally the separation theorem does not hold for broadcast channels, where a source communicates with a multitude of receivers. Based on these reasons, a *joint source/channel coding* approach is often preferred for practical systems.

Fig. 1.1 shows a diagram of connections among the modules of a practical data communication system. These modules include a source coder, channel coder, transmission channel, channel decoder and source decoder. This diagram is used to illustrate the design of a joint source/channel coding system.

- Source significance information (SSI) and channel coding information (CCI) are passed between the source coder and the channel coder for the purpose of *joint source/channel encoding.* In addition, channel state information (CSI) is necessary for schemes adopting channel adaptive rate allocation. For example, static and dynamic unequal error protection can be achieved based on this information.

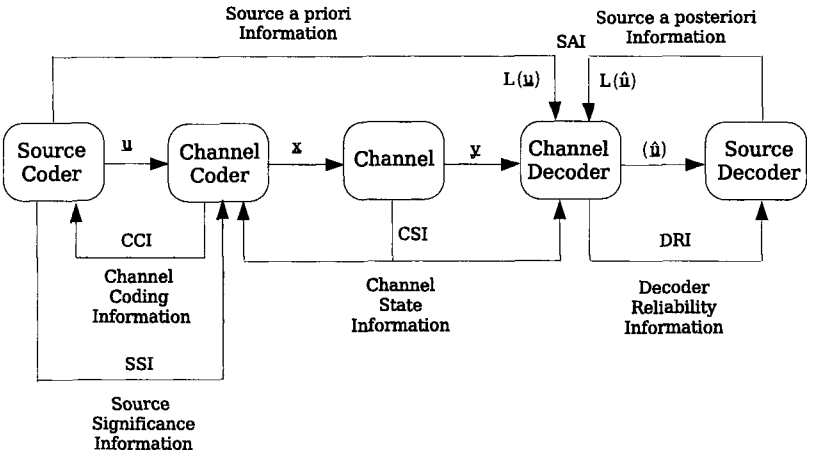


Figure 1.1: A diagram of connections among the modules of a practical system.

- The channel decoder and demodulator can have access to hard decision outputs from the demodulator/detector. Furthermore, soft decisions and CSI can be passed on to improve channel decoding performance.

- At the receiver, when a channel decoding failure occurs, some source bits may be lost or discarded. One of the common techniques in error concealment is to interpolate these lost source samples. More sophisticated methods use the decision on the current bit and its reliability. This requires that the source decoder receives the "soft outputs" from the channel decoder, i.e., decisions and "decision reliability information" (DRI) generated from channel decoding. This leads to a *joint source/channel decoding* solution.

- Another approach connects source and channel decoding to a greater extent in joint source/channel decoding. Source-controlled channel decoding uses *a priori* and *a posteriori* information about the source bits. For reasons such as delay, complexity and nonstationary channel conditions, many source coding schemes intentionally embed some redundancy in their bitstreams. As pointed out by Shannon [1], "$\cdots$ any redundancy in the source will usually help if it is utilized at the receiving point. In particular, if the source already has redundancy and no attempt is made to eliminate it in matching to the channel, this redundancy will help combat noise." Therefore, the residual redundancy from a source bitstream should be utilized jointly by the channel and source decoders. The

source *a priori/a posteriori* information (SAI) tells the channel decoder, with certain probability, the value of the next bit to be decoded. Therefore in source-controlled channel decoding, the channel decoder error rate can be reduced by supplying the channel decoder with source information.

In the rest of this dissertation, both joint source/channel encoding and decoding are generally referred to as joint source/channel coding.

The broad area of joint source/channel coding has been under active research for many years. Many successful schemes, both theoretical and practical, with different target sources, like audio, image and video, have been proposed in the literature.

In [2], the authors gave a theoretical analysis of the tradeoff between lossy source coding and block channel coding for a vector quantizer cascaded with a channel coder and a binary symmetric channel. Tight bounds for optimal channel code rates that minimize average distortion were derived. In [3], the evaluation of operational rate-distortion behavior for different combined source and channel coding approaches was investigated. This evaluation procedure was then applied to images coded with a wavelet-based coder and demonstrated performance close to theoretical bounds. The problem for optimal distribution of source and channel bits among subbands for transmission of scalable video over noisy channels was studied in [4]. The proposed algorithm adapts to the estimated channel and optimally transmits video for the current channel state. The schemes discussed in this paragraph utilize SSI and CCI at the transmitting end in Fig. 1.1.

In [5], the Viterbi decoding algorithm was modified to utilize *a priori* or *a posteriori* information about the source bit probability for better decoding. The source information comes from the correlation in certain bits after source coding. The scheme was applied to a GSM speech codec over Gaussian and Rayleigh channels. In [6], source redundancy is embedded into the source coding by reserving space for a symbol that is not in the source alphabet for arithmetic codes. The redundancy is exploited by the arithmetic coder and a sequential decoder for image transmission. No explicit channel coding was used in that work. Redundancy was embedded in a similar fashion in [7]. The redundancy was used for error detection in a continuous fashion to increase the throughput of an ARQ-based image transmission system. Iterative joint source/channel decoding was considered in [8] in the spirit of serially concatenated codes. The dependencies between the variables involved in arithmetic coding was analyzed by means of a Bayesian network. The proposed soft decoding framework can provide high error resilience and was applied to practical systems with context-based arithmetic coding such as JPEG2000. All the schemes discussed in this paragraph utilize CSI, DRI and SAI at the receiving end in Fig. 1.1.

Recent progress made in the source coding area has seen an increasing emphasis on source scalability and error resilience aspects. A compressed data stream is said to be scalable if it consists of an embedded collection of smaller streams, each representing an efficient compression of the original source or some portion thereof [9]. With a scalable source coder, a source can be compressed once but decompressed in many

ways according to various application requirements. Such requirements may include display resolutions, region of interest, communication capacities and so forth. For the channels experienced by real-world communications, error-free delivery of information is not always guaranteed. Therefore, it is important that a source coder can generate bitstreams that are resilient to transmission errors, in order to provide graceful quality degradation in the presence of residual errors. Error resilience can be realized in a number of ways. It can be implemented at the source and channel encoder to make the bitstream more resilient to potential errors, or can be invoked at the decoder upon error detection to conceal the effects of errors, or the source encoder and decoder can interact adaptively [10].

Remarkable progress has been made in the channel coding area during the last decade. Specifically, the discovery of Turbo codes in 1993 [11] and the rediscovery of LDPC codes in 1995 [12] have spurred great research interest to construct good codes and decoding algorithms based on their underlying principles. Both Turbo codes and LDPC codes have been shown to be able to approach the Shannon limit asymptotically [13] [14]. The new concepts brought by these codes, such as codes defined on graphs and message passing decoding, are currently being studied in great depth and applied to broader areas.

JPEG2000 is the latest international standard for still image compression [15]. Besides providing state-of-the-art compression performance, JPEG2000 offers a number of functionalities to address the requirements from emerging multimedia applications.

In particular, JPEG2000 offers scalability and error resilience properties which are highly desirable for visual communication purposes.

The combination of the excellent source coding properties provided by source coders such as JPEG2000, together with the capacity-approaching channel coding schemes such as Turbo and LDPC codes, poses a promising but challenging problem for practical multimedia communication system design. This is the motivation of this dissertation.

1.2 Organization and Contributions

This dissertation studies the problem of joint source/channel coding for image and video transmission over noisy communication channels.

In Chapter 2, relevant background information is provided. The background includes some state-of-the-art image and video coding schemes, some channel coding schemes that are adopted for joint source/channel coding applications, and the channel models that are employed or discussed in the dissertation.

In Chapter 3, a joint source/channel coding scheme is proposed for image transmission over memoryless channels. It is tailored for JPEG2000 with the goal to minimize the end-to-end image distortion within a given total transmission rate. It provides unequal error protection by combining the forward error correction capability from channel codes and the error detection/localization functionality from JPEG2000 in an

effective way. The proposed scheme generates quality scalable and error-resilient bitstreams. It gives competitive performance with other existing schemes for JPEG2000 in matched channel condition cases, and provides more graceful quality degradation for mismatched channel condition cases. Furthermore, both fixed length source packets and fixed length channel packets can be efficiently formed with the same algorithm.

In Chapter 4, a joint source/channel coding scheme is proposed for image transmission based on JPEG2000 and LDPC codes over channels with memory. The capability of combating bursty errors from LDPC codes is combined with the error resilience functionality from JPEG2000. The same source bitstream structure and joint rate allocation algorithm from Chapter 3 are employed. It provides significantly improved distortion performance over the existing schemes. At the same time, it offers more flexibility in packetization and re-packetization during transmission. In this chapter, an irregular LDPC code construction method is also proposed to have efficient representation of the constructed LDPC codes.

In Chapter 5, a joint source/channel coding scheme is proposed for the transmission of multiple sources sharing the total bandwidth given by a common channel. The algorithm optimally distributes the available bit rate to each source by exploiting the rate-distortion diversity among the sources. The algorithm is applied to different sources, including image, video and HDTV sequences. It is demonstrated that by

coding multiple sources jointly, both improved quality and a reduction in quality variation can be achieved for the reconstructed sources at the receiver.

In Chapter 6, a robust image transmission scheme based on JPEG2000 is proposed for packet erasure channels. The error resilience functionalities provided by JPEG2000 are utilized to control the source coding efficiency and robustness according to the channel conditions. Furthermore, together with the proposed interleaving scheme, some erasures can be recovered.

In summary, the contributions of this dissertation include:

- A joint source/channel coding algorithm is proposed for image transmission over channels with or without memory. The proposed algorithm generates quality scalable and error resilient bitstreams. For memoryless channels, the proposed algorithm gives performance competitive with other existing schemes for matched channel condition cases and provides more graceful degradation for mismatched channel condition cases. For channels with memory, together with the LDPC codes, the algorithm provides significant performance improvement over other existing schemes and at the same time gives more flexibility in packetization/re-packetization during transmission.

- A joint source/channel coding algorithm is proposed for the transmission of multiple sources including images, video and HDTV sequences, sharing a common channel. By coding multiple sources jointly with the proposed algorithm,

significant gains can be achieved in both quality and quality variation reduction for the reconstructed sources at the receiver.

- A robust image transmission scheme is proposed based on JPEG2000 for packet erasure channels. Error resilience functionalities provided by JPEG2000 are utilized to detect and recover erasures, and to localize residual erasure impact.

CHAPTER 2

BACKGROUND

This chapter presents the definitions, terminology and background information about the source coding, channel coding and the channel models that will be used and discussed in the following chapters of the dissertation. Section 2.1 discusses the image and video coding schemes, Section 2.2 describes the channel coding schemes and Section 2.3 gives the channel model definitions.

2.1 Source Coding

For a lossy image or video coder, its goal is to represent the original source with as few bits as possible to achieve good fidelity according to some measurements. This is usually achieved by trading the distortion with rate by a source coder. For an image source, the rate of a compressed bitstream is usually reported in bits per pixel (bpp). And for a video source, the units of bits per second (bps) together with frames per second (fps) are often adopted. There exist many ways to measure the quality of a reconstructed image or video source. A common objective distortion measure is mean squared error (MSE). For a reconstructed image or video frame $\hat{X}$ with dimension

$N \times M$, the MSE with respect to the original source X is defined as

$$\mathrm{MSE} = \frac{1}{\mathrm{NM}} \sum_{\mathrm{i=1}}^{\mathrm{N}} \sum_{\mathrm{j=1}}^{\mathrm{M}} (\mathrm{X_{i,j}} - \hat{\mathrm{X}}_{\mathrm{i,j}})^2 \tag{2.1}$$

The MSE value is often converted to a logarithmic scale to accommodate a large dynamic range. The most common of such measures is peak signal-to-noise ratio (PSNR) and it is the ratio of the peak amplitude squared to the MSE. For 8 bpp graylevel sources, it is given by

$$\mathrm{PSNR} = 10 \log_{10} \frac{255^2}{\mathrm{MSE}} \quad \mathrm{dB} \tag{2.2}$$

2.1.1 SPIHT

Set partitioning in hierarchical trees (SPIHT) [16] is an image compressor based on the zero-tree coding of wavelet subband samples. It produces state-of-the-art MSE performance as well as embedded bitstreams and has received tremendous attention by the image and video compression communities.

The SPIHT algorithm utilizes three basic concepts: (1) searching for sets in spatial-orientation trees in a wavelet transform; (2) partitioning the wavelet transform coefficients in these trees in a bit-plane representation of their magnitudes; and (3) coding and transmitting bits associated with the highest remaining bit planes first.

In SPIHT, the spatial orientation trees are groups of wavelet transform coefficients organized into trees rooted in the coarsest scale subband. Their offspring extend along

the same spatial orientation in the higher frequency subbands, as shown in Fig. 2.1. In the spatial orientation trees, each node consists of 2×2 adjacent pixels, and each pixel in the node has four offspring, except at the highest level of the pyramid, where one pixel (with "*" in the figure) of each node does not have any offspring. Spatial orientation trees were introduced to exploit self-similarity and magnitude localization properties in a 2D wavelet transformed image.

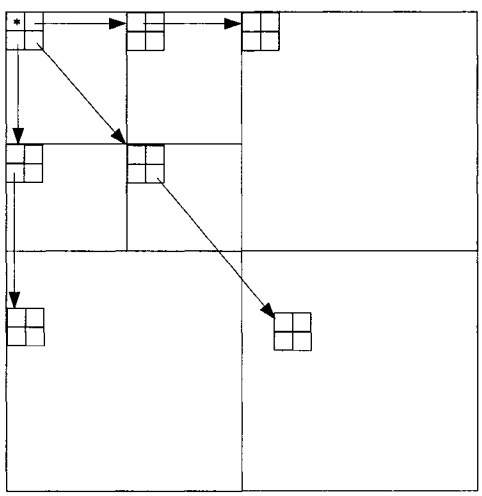


Figure 2.1: Spatial orientation trees in SPIHT for dyadic wavelet decomposition.

SPIHT consists of two main stages: sorting and refinement. In the sorting stage, SPIHT searches the spatial-orientation trees to identify entities satisfying some properties. It partitions the trees into sets of three entities and moves their coordinates into one of three lists: (1) the list of significant coefficients (LSC); (2) the list of isolated insignificant coefficients (LIS); and (3) the list of insignificant sets of coefficients (LIC). During the refinement pass, the nth bit of every member of the LSC is emitted to the bitstream, where n is the bit significance number and is successively lowered from the maximum of the largest magnitude coefficient. Then the three lists are tested and updated. This process terminates when the desired bit rate or quality level is reached.

A three dimensional extension of the SPIHT algorithm (3D-SPIHT) was proposed in [16] for video coding. In 3D-SPIHT, a group of contiguous frames (hereafter called GOF) are first temporally wavelet transformed with/without motion compensation. Then, each resulting frame is separately transformed in the spatial domain. Thus a GOF is decomposed temporally and spatially into three-dimensional subbands. A new 3D spatio-temporal orientation tree and its parent-offspring relationships are defined on the 3D subband structure. Based on the new 3D spatial-temporal tree structure, a 3D-SPIHT kernel sorts the data according to the magnitude along the spatio-temporal orientation trees and refines a bit-plane by adding necessary bits. Also, when motion-compensation is performed, the motion vectors are separately

coded losslessly and transmitted over the transmission channel with high priority. 3D-SPIHT provides complete embeddedness for progressive fidelity transmission, precise rate control for constant bit-rate traffic and high performance for video coding.

Despite their superior performance and full embeddedness, SPIHT and 3D-SPIHT do not explicitly consider the error resilience issue. To that end, an error resilient 3D-SPIHT was proposed in [17]. Its basic idea is to divide the 3D wavelet coefficients into some number P of different groups according to their spatial and temporal relationships, and then encode each group independently using the 3D-SPIHT algorithm, so that P independent embedded 3D-SPIHT substreams are created. Therefore, a bit error in one substream does not affect the decoding of other substreams. Because all substreams are embedded, the partial decoding of the error free part of a corrupted substream only results in a less accurate rendition of some specific region, while other regions corresponding to correct substreams are reconstructed to their full fidelity.

2.1.2 JPEG2000

JPEG2000 [15] [18] is based on the discrete wavelet transform (DWT) and embedded block coding with optimized truncation (EBCOT) [19]. Figures 2.2 and 2.3 show the diagram of a JPEG2000 encoder and some geometric structures employed by JPEG2000.

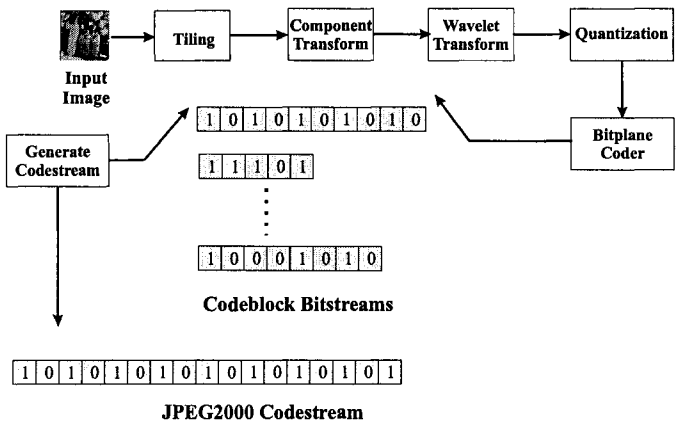


Figure 2.2: Block diagram of a JPEG2000 encoder.

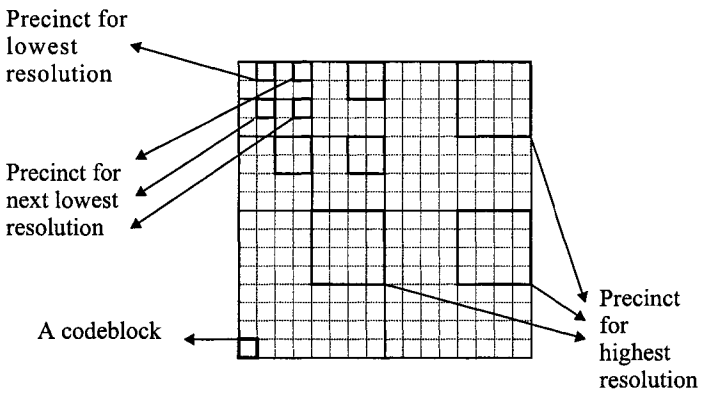


Figure 2.3: Geometric structures within JPEG2000.

An input image is first optionally subdivided into non-overlapping rectangular *tiles* prior to compression [18]. If the image has multiple components, these components may also be optionally subjected to a decorrelating color transform. Each image component is then wavelet transformed into a collection of subsampled spatial frequency bands known as *subbands*.

These subbands are further partitioned into rectangular blocks known as *codeblocks*. Codeblocks are the smallest geometric structure in JPEG2000, as shown in

Fig. 2.3. Each codeblock is an independent coding unit. The wavelet coefficients within a codeblock are quantized using a deadzone uniform scalar quantizer. The quantized coefficients are then sent to a bit-plane coder. The bit-plane coder makes three passes over each bit-plane of the samples of a codeblock. These passes are referred to as *coding passes* and are called *significance propagation pass*, *magnitude refinement pass* and *cleanup pass*. In the significant propagation pass, the bit-plane coder processes those samples which are not currently significant but have a significant neighbourhood. In the magnitude refinement pass, the bit-plane coder includes those samples which first became significant in a previous bitplane. Finally, in the cleanup pass, all samples that are passed over by the previous two coding passes are included. During the coding process, the encoder computes and stores the rate-distortion information corresponding to each coding pass. The entropy coding of the bit-plane coder is based on a context-dependent, binary arithmetic coder known as the MQ-coder.

The generated bitstream from a bit-plane coder for each codeblock is finely embedded. This means that truncating such a bitstream has the effect of quantizing the samples in that codeblock more coarsely. And there are numerous truncation points in a codeblock's bitstream.

As illustrated in Fig. 2.3, the codeblocks of particular wavelet resolutions are grouped together to form *precincts*. For the purpose of forming a final bitstream,

compressed data from each precinct are arranged to form *packets*. Each packet consists of the compressed bytes from a certain number of the coding passes from each codeblock in one precinct of one tile-component.

In order to obtain quality scalability, JPEG2000 provides the quality layer functionality. A *layer* is comprised of one packet from each precinct of each resolution of each tile-component. Each layer represents an incremental quality contribution from the embedded bitstream associated with each codeblock in the image. The sizes of these incremental layer contributions are optimized based on post-compression rate-distortion (PCRD) optimization [19].

Besides state-of-the-art compression efficiency, JPEG2000 provides scalability and error resilience functionalities. In JPEG2000, scalability is supported in four dimensions: Quality, Resolution, Spatial Location and Component. The error resilience aspect will be described in detail in Chapter 3.2.

Similar to 3D-SPIHT, JPEG2000 can be extended to 3D-JPEG2000 for video coding by performing a wavelet transform along the temporal dimension with/without motion compensation. Without motion compensation, temporal filtering produces visually disturbing ghosting artifacts in the low-pass temporal subband, especially at low bit rates. Motion compensation was realized in the JPEG2000 framework in [20] [21] based on a lifting-based invertible motion adaptive transform (LIMAT). Both 3D-SPIHT and 3D-JPEG2000 have applications with relatively low resolution and bit rate requirements, such as video conferencing, video streaming, etc.

Motion JPEG2000 (MJP2) [22] [18] is an extension of JPEG2000 for intra-frame based video coding. With MJP2, each video frame is individually compressed by a JPEG2000 image coding engine. The resulting bitstreams for these frames are wrapped into a file having the MJP2 file format [18]. Although motion compensation (MC) is not employed by MJP2, experimental results show that MJP2 can still provide competitive performance to MC-based video coding standards, such as MPEG-4 and H.264/AVC, in applications requiring high quality or high resolution [23]. In addition, MJP2 can generate highly scalable bitstreams compared to most existing video standards. For these reasons, MJP2 has become a promising choice for applications like HDTV broadcasting [24].

2.2 Channel Coding

Channel coding has become an indispensable component in modern communication systems with stringent constraints such as delay and power. In this section, we focus on some of the codes that are widely used in the joint source/channel coding literature. These codes are favored because they can achieve good error protection performance, as well as the capability to easily or inherently provide different error protection levels.

2.2.1 Rate-Compatible Punctured Convolutional (RCPC) Codes

A family of RCPC codes can be described by a mother code of rate $R = 1/N$ and memory M, having a generator tap matrix

$$\mathbf{g} = (\mathbf{g_{ik}})_{\mathbf{N\times(M+1)}} \tag{2.3}$$

with the tap connection $g_{ik} \in \{0, 1\}$, where a 1 represents a connection from the kth shift register stage to the ith output. Together with N, a puncturing period P determines the range of the code rates

$$R = \frac{P}{P + l} \qquad l = 1, \cdots, (N-1)P \tag{2.4}$$

between $P/(P+1)$ and $1/N$. The RCPC codes are punctured codes of the mother code with puncturing matrices

$$\mathbf{a(l)} = (\mathbf{a_{ij}(l)})_{\mathbf{N\times P}} \tag{2.5}$$

with $a_{ij}(l) \in \{0, 1\}$, where 0 implies puncturing.

The rate-compatibility restriction implies the following rule:

$$\text{if} \quad \mathbf{a_{ij}(l_0) = 1} \quad \text{then} \quad \mathbf{a_{ij}(l) = 1} \quad \text{for all} \quad \mathbf{l \geq l_0 \geq 1} \tag{2.6}$$

or equivalently

$$\text{if} \quad \mathbf{a_{ij}(l_0) = 0} \quad \text{then} \quad \mathbf{a_{ij}(l) = 0} \quad \text{for all} \quad \mathbf{1 \leq l_0 \leq (N-1)P - 1} \tag{2.7}$$

This restriction on the puncturing tables ensures that all code bits of the high rate codes are used by the lower rate codes, i.e., the high rate codes are embedded in the lower rate codes.

On the receiving side, with the knowledge of the current puncturing rule **a**, an RCPC decoder can perform decoding based on the Viterbi algorithm (VA) or MAP decoding algorithm, as was proposed in [25]. During the decoding, $a_{ij} = 0$ means that the code symbol x_{ij} has not been transmitted and the receiver may simply insert an erasure.

RCPC codes are considered to be ideal for achieving unequal error protection. A set of channel codes with different strengths can be obtained with a single rate $1/N$ convolutional encoder and a Viterbi decoder sharing a puncturing table. This simplifies the hardware implementation of practical systems.

2.2.1.1 List Viterbi Decoding Algorithm (LVA)

The List Viterbi decoding algorithm (LVA) is one of the various generalizations of the Viterbi algorithm (VA), and RCPC codes can be decoded alternatively using LVA. The LVA produces a rank ordered list of the L globally best candidates after a trellis search. Two schemes are available to find these candidates. One is a parallel LVA that simultaneously produces the L best candidates. The other is a serial LVA that iteratively produces the kth best candidate based on the knowledge of the previously found $k-1$ candidates [26].

A framework that combines RCPC and LVA is shown in Fig. 2.4. Its encoder is a concatenated coder which consists of an inner forward error correction (RCPC) coder and an outer error detection (CRC) coder. During its encoding, a block of N_i data bits are augmented with N_c cyclic redundancy check (CRC) codes by the outer coder. M known information bits for terminating the trellis are added to the $N_i + N_c$ bits before being encoded by the inner RCPC coder. This block of $N_T = N_i + N_c + M$ bits are encoded by a family of RCPC codes with memory M, and rates from 1 to $1/N$. At the decoder, the inner LVA decoder releases the k ($1 \leq k \leq L$) best candidates. The outer CRC decoder performs error detection on these candidates. A decoding success is declared when a candidate passes the CRC check, and a decoding failure is declared when all L best candidates fail. Proper actions may be taken by the decoder upon failure, such as asking for transmitting more incremental redundancy at lower code rates, or just simply discarding an erroneous data frame.

2.2.2 Turbo Codes

Turbo codes were introduced in 1993 [11] and have been shown to be able to provide very close performance to Shannon limit (within 0.5 dB) [13].

One version of a turbo code, parallel concatenated convolutional code (PCCC), consists of a parallel concatenation of two recursive systematic convolutional (RSC) constituent codes, each separated by a pseudo-random interleaver of block size N. Each RSC is fed by the same information bits, but in different orders permuted

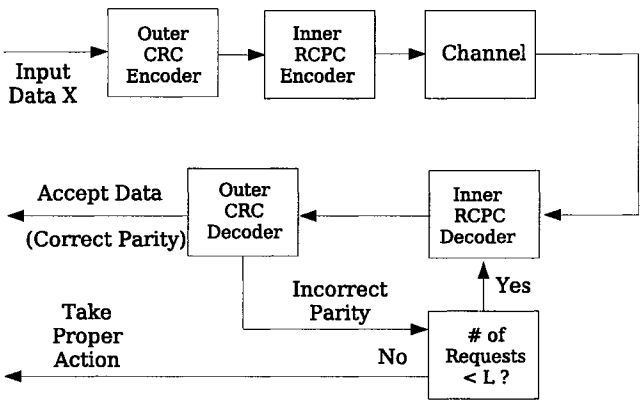


Figure 2.4: Block diagram of a data transmission system based on RCPC/CRC and LVA.

by the interleaver. For a block of information bits to be encoded, each RSC is generally terminated by appending a suitable number of dummy bits at the end of the information word to drive the encoder trellises to the zero states. Besides PCCC, other ways of concatenating encoders and interleavers are also possible, such as serial concatenation of two convolutional encoders (SCCC) separated by an interleaver acting on the code words of the outer code, or hybrid concatenation (HCCC), etc.

Turbo decoding is based on the BCJR algorithm [27] proposed in 1974. BCJR is an algorithm that computes the *a posteriori* probability (APP) of each symbol based on the knowledge of the whole received sequence of the soft values. It computes the APP for each symbol through two recursions on the code trellis, one in the forward direction and the other in the backward direction, and it has a computational complexity per decoded symbol that is independent of block size and increases linearly with the number of trellis states. Within a Turbo decoder, there is a local decoder employing the BCJR algorithm for each RSC used at the Turbo encoder. These local decoders exchange extrinsic information iteratively, taking advantage of the progress made by the others to give performance close to maximum likelihood decoding.

Similar to RCPC codes, rate compatible punctured Turbo (RCPT) codes have been studied, such as in [28], [29]. RCPT codes follow the formulation of RCPC codes almost directly, and enjoy the same advantages in terms of rate compatibility as discussed in 2.2.1.

2.2.3 LDPC codes

Low density parity-check (LDPC) codes were originally introduced by Gallager in the 1960s [30], and were rediscovered by Mackay and Neal in 1995 [12].

An LDPC code is a special class of a linear block code whose parity-check matrix H has a low density of ones, i.e., is sparse. An LDPC code can be represented by a Tanner graph $\mathcal{G} = \{(\mathcal{V}, \mathcal{E})\}$, where $\mathcal{V}$ is a set of vertices and $\mathcal{E}$ is a set of edges

connecting the vertices. A Tanner graph $\mathcal{G}$ is a bipartite graph, where vertices $\mathcal{V}$ can be decomposed into two disjoint sets $\mathcal{V}_1$ and $\mathcal{V}_2$ such that no two vertices within either $\mathcal{V}_1$ or $\mathcal{V}_2$ are connected by an edge. These two disjoint sets collect the variable nodes and the check nodes, where each bit of a codeword is assigned a variable node and each parity-check equation is assigned a check node. In a Tanner graph, a cycle of length l is a path comprised of l edges which closes back on itself, and the girth of the graph is the minimum cycle length of the graph.

As an example, the following H matrix has 4 rows and 8 columns. It can be represented by a Tanner graph in which 4 check nodes C_1 – C_4 (starting from the top row to the bottom row) are connected with 8 variable nodes V_1 – V_8 (starting from leftmost column to the rightmost column) as shown by Fig. 2.5. Its girth is 4 as illustrated by the bold lines.

$$H = \begin{bmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

The decoding of LPDC codes is based on the sum-product algorithm, or more generally, the message passing algorithm [31]. It uses local information rules to compute the probability distributions of variables in graph-based models. The presence of cycles in a Tanner graph results in indefinite propagation of messages without natural termination in an iterative decoding. Furthermore, with cycles in a graph,

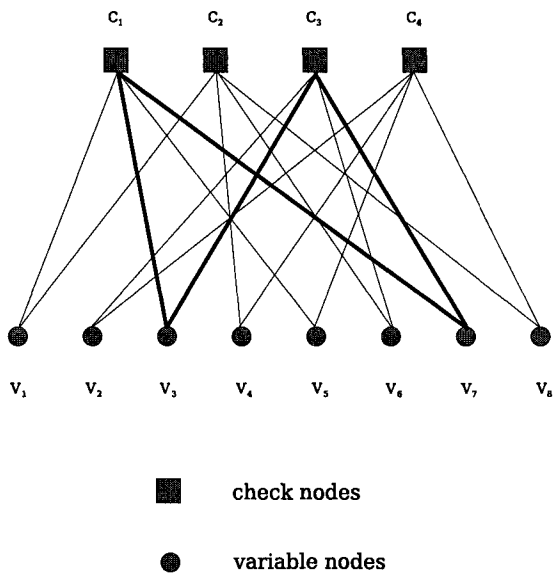


Figure 2.5: The Tanner graph of a parity-check matrix H.

the messages no longer represents the conditional probability mass functions given some subset of the evidence. This compromises the performance of the sum-product decoding algorithm. Also it has been shown [32] that the lower bound on the minimum distance d_{min} increases with girth of the code. Therefore, LDPC codes with large girth are preferred.

LDPC codes can be constructed from random parity-check matrices and give excellent BER performance on average. However, the asymptotic performance of random

graphs cannot be guaranteed for relatively short codes (for example, less than several thousand bits.) Another problem that arises from a practical point of view is the large memory required to specify the locations of ones in the parity-check matrix H of a randomly constructed LDPC code. To address these problems, structured construction of LDPC codes has received much attention lately, such as codes constructed based on Euclidean/projective geometries, or Steiner/Kirkman triple systems. LDPC codes thus constructed are easy to describe and their Tanner graphs are free of four cycles. Moreover, the explicit construction also gives hope to analyze the code distance properties.

2.3 Channel Models

Image or video communications may be required within different transmission environments, such as outdoor/indoor wireless, packet wireline, deep space, etc. Different environments impair the transmitted data in different ways and have a deep impact on the corresponding system design strategies. Therefore a model that depicts the exact nature of the channel is of necessity. This section provides background for some commonly used channel models, especially those discussed in this thesis.

2.3.1 Memoryless Channels

Memoryless channels are the predominant type of channel models used in communication system analysis and design. In such a channel, the noise affects each

transmitted symbol independently. Consider a channel denoted by ($\mathcal{X}$, $p(y|x)$, $\mathcal{Y}$). It consists of two sets $\mathcal{X}$ and $\mathcal{Y}$ and a set of conditional probability mass or density functions $p(y|x)$, for channel input-output pairs (x, y), $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

The additive white Gaussian noise (AWGN) channel is the most important continuous-output memoryless channel. In this channel model, the input typically comprises symbols selected from a finite and discrete set $\mathcal{X}$, and the output belongs to a continuous set $\mathcal{Y} = \{-\infty, +\infty\}$. The channel noise Z is drawn i.i.d. from a Gaussian distribution with zero mean and variance σ^2, and Z is assumed to be independent of the input signal X. An AWGN channel model can thus be described as

$$Y = X + Z, \qquad Z \sim N(0, \sigma^2). \tag{2.8}$$

The binary symmetric channel (BSC) is a special case of a discrete-input, discrete-output memoryless channel. Its input and output sets are both limited to $\{0, 1\}$. BSC can also be seen as a special case of a binary input AWGN channel where the output is quantized into two values. In this channel model, the input symbols are complemented with a crossover probability ϵ, i.e., a transmitted 0 is received as a 1 with probability ϵ and vice versa.

BSC may occur when the channel modulator employs binary waveforms at the input and the channel detector makes hard decisions at the output. It is the simplest model of a channel with errors, yet it captures most of the complexity of a general problem. For example, in many transmission systems, the application layer typically

is designed independent of the hardware level. The modularity of system design leaves some applications with access to the data after some form of hard decision decoding, and not the direct soft output from the channel observation.

The condition that a channel is memoryless can be expressed as

$$p(y^n|x^n) = \prod_{i=1}^{n} p(y_i|x_i) \tag{2.9}$$

for any given input sequence $\{x_i\}$, $i = 1, 2, \ldots, n$.

2.3.2 Channels with Memory

As opposed to memoryless channels, the outputs from a channel with memory exhibits correlation. Channels with memory are more realistic models for many real-world channels, such as the Internet and wireless channels. In these channels, errors tend to occur in clusters separated by relatively long error-free gaps.

2.3.2.1 Gilbert-Elliot Channel (GEC)

The Gilbert-Elliot channel (GEC) model is a simple model for a discrete channel with memory. It obeys a two-state Markovian model with a GOOD (G) state and a BAD (B) state, as shown in Fig. 2.6. In each state, the model generates a noise bit before making a state transition. Within each state the channel acts like a BSC of the appropriate error rate (ϵ_B for the bad state and ϵ_G for the good state). The channel changes state with probabilities governed by the state transition probabilities P_{GB} = Pr(G→B) and P_{BG} = Pr(B→G).

For a GEC, the average fade duration or mean holding time in the bad state is

$$\bar{\tau} = \frac{1}{P_{BG}} \tag{2.10}$$

and the average bit error probability is given by

$$\bar{\epsilon} = \epsilon_G \frac{P_{BG}}{P_{BG} + P_{GB}} + \epsilon_B \frac{P_{BG}}{P_{BG} + P_{GB}} \tag{2.11}$$

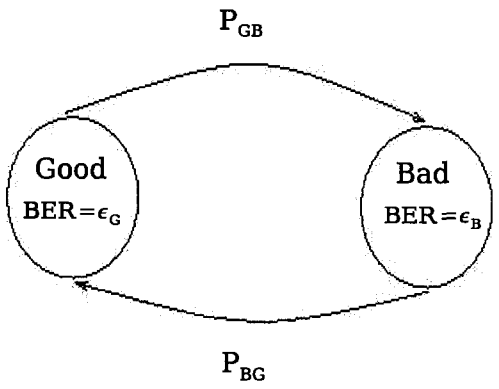


Figure 2.6: Gilbert-Elliot channel model.

2.3.2.2 Rayleigh Channel

The Rayleigh channel is a fading channel model in which the state varies over a continuous range as opposed to the two-state Gilbert-Elliot model. It is often used to simulate fading channels in mobile and wireless communication systems, and is characterized by two parameters, i.e., the average received signal-to-noise rate $\overline{SNR}$ and the normalized Doppler spread. In such a system, the received signal power varies due to the constructive and destructive interference between copies of the signal arriving from multiple paths. If the delay experienced by all paths is short relative to a bit interval, the channel is said to be a flat-fading channel. In this case, the received signal waveform shape is distorted. The effect of the multipath can be seen as a multiplicative gain α on the received signal. Only flat-fading Rayleigh channels are discussed in this thesis.

2.3.3 Packet Erasure Channels

A packet erasure channel is one whose data, prior to transmission, is partitioned into a sequence of packets, each of which either arrives at the destination without error, or is entirely lost. The packet erasure channel is a good model for modern packet-switched networks, such as ATM (Asynchronous Transfer Mode), IP (Internet Protocol) and its adoption into the wireless realm, e.g., GPRS (General Packet Radio Service). In these networks, when the number of packets sent exceeds transmission capacity, packets are discarded at random. With the underlying transport protocol,

each packet can be assigned a unique sequence number, so the receiver can sort the packets according to their transmission order and obtain the knowledge of which packets have been lost.

There exist several ways to model a packet erasure channel. One common approach is to assume that the channel packet loss process is i.i.d.. It means that every packet has the same erasure probability p and the loss of one packet does not influence the likelihood of losing other packets. This leads to a memoryless channel model adopted by authors such as of [33], [34], etc. Another more general approach is to assume that there exists an estimator that outputs a probability mass function (PMF) indicating the likelihood that a particular number of packets are lost, given the total number of packets to be transmitted. This estimator could be almost any model of expected packet erasure rates. A PMF can realize uniform, binomial, Zipf, Poisson, exponential distributions and so forth. This model can capture the burstness of packet loss events by adopting appropriate PMFs, thereby introducing memory effect into the channel model. This approach is taken by authors such as of [35], [36], etc.

An extension of the model is to concatenate a packet erasure channel with a bit error channel such as one discussed in Sections 2.3.2 and 2.3.2. This channel model can be used to simulate the end-to-end data transmission from a server through a wireline network to a radio base station which broadcasts the data to mobile receivers. The wireline part of the channel suffers packet losses. Each packet is assumed to be either correctly transmitted or dropped at random. The wireless part is assumed

to have no packet losses, but may have considerable bit errors (if the wireless part does have packet losses, the packet erasure probability of the wireline part can be appropriately increased to account for it).

CHAPTER 3

JOINT SOURCE/CHANNEL CODING FOR SINGLE IMAGE TRANSMISSION OVER MEMORYLESS CHANNELS

3.1 Introduction

Today's multimedia applications generate large volumes of source information, including speech, audio, images and video. To transmit this information efficiently, source compression is needed to remove as much redundancy as possible. On the other hand, in order to combat errors introduced by noisy channels, channel coding techniques are often employed to add controlled redundancy. As introduced in Chapter 2, Shannon's separation theorem may not always apply in practice or certain types of applications. Therefore, joint source/channel coding (JSCC) techniques may be of interest for practical systems.

The new generation of wavelet-based image coders, such as SPIHT [37] and JPEG2000 [15], can provide efficient compression and highly scalable bitstreams, which make them very suitable for image transmission purposes. However, through the extensive use of context-based entropy coding, their bitstreams are very sensitive to bit errors. A single error could render a whole bitstream useless. The application

of JSCC to image transmission with these image coders has attracted a significant amount of research effort recently.

In applications employing JSCC, joint allocation is usually performed to determine the optimal distribution of limited resources (such as total transmission rate) between source and channel coders, with the goal to optimize some desired measure(s) (such as delivered image quality). Generally, there are two ways to perform the rate allocation: equal error protection (EEP) and unequal error protection (UEP). In EEP, an entire bitstream generated by an image coder receives the same amount of protection from a channel coder. The UEP schemes, however, apply different amounts of protection to different sections of a bitstream. Since bitstreams from progressive image coders are generally scalable, the importance of different bits may not be equal. UEP schemes can effectively shift uncorrectable bit errors toward the less importance sections. Therefore, it is possible to achieve better performance with UEP than with EEP.

A practical system with rate-compatible punctured convolutional (RCPC) codes was presented in [38] to protect SPIHT bitstreams for a set of binary symmetric channels. A channel code rate is carefully chosen for each channel such that it has an extremely low probability of error after channel decoding. Then, an entire SPIHT bitstream is equally protected with the selected code. In [39], a UEP scheme using RCPC was proposed for SPIHT. An iterative method is used to partition a bitstream into different sections and to select a channel code rate for each section. This UEP scheme provides an improvement of around 0.3 dB over the EEP employed

in [38]. A dynamic programming method is used in [40] to perform rate allocation using optimization criteria based on MSE, PSNR and available source rate. The last criterion is suggested as the best approach, since it reduces the complexity, eliminates the need to transmit the rate schedule, and allows optimal transmission at many rates lower than the optimized target rate. In [41], the authors proposed a UEP scheme based on RCPC and an error resilient wavelet coder. The source coder generates a number of layers based on bitplanes of blocks in the wavelet domain. These layers are independently decodable and optimal RCPC codes are applied to the segments of each layer. The whole bitstream consists of the concatenation of all these individually protected layers, and it is possible for its source decoder to decode portions of the bitstream after the occurrence of uncorrectable errors from the channel decoder. A new measure was proposed in [42] which considers the performance not only at the target transmission rate but also several intermediate rates for progressive transmission with SPIHT. Using dynamic programming, the optimal block lengths and the associated channel rates are obtained for Reed-Solomon (RS) and turbo codes.

In [43], an efficient rate allocation was proposed based on the available source rate criterion from [40]. It achieves optimal protection with reduced complexity for the SPIHT image coder. RS codes were used for JPEG2000 image transmission in [44]. An iterative optimization scheme is employed to find the optimal rate allocation. In [45], a Viterbi-based rate allocation method was presented for protecting

JPEG2000 bitstreams with turbo codes. It has complexity growing as $O(N^2)$ with N the number of transmitted packets, and results in a good approximation to the optimal performance obtained from brute force search. In [46], the authors use a local search algorithm for rate allocation. It starts from a rate-optimal solution and converges to a locally distortion-optimal solution, which gives comparable performance to that obtained by [45], but with reduced complexity. A scheme employing the same rate allocation scheme but more powerful irregular repeat-accumulate (IRA) channel codes was presented in [47].

For many applications, the channel condition can not be predicted precisely before transmission, or it may be location- or time-varying (such as wireless channels). As JSCC techniques usually require adequate knowledge about the transmission channel, a channel condition mismatch could cause devastating decoding effects. Therefore it is highly desirable for those applications to be able to tolerate a certain amount of residual errors caused by the nonergodic channel behavior or from some other error sources [48].

In some UEP schemes (for example, [38] - [42]), a fixed number of source bits are organized into each packet and then the packets are protected by channel codes with different rates by adding a variable number of parity bits. The packets therefore may have different total lengths, but each packet contains a fixed number of source bits. On the other hand, other schemes (such as [43] - [47]) use fixed-length channel packets, i.e., all packets have a fixed length after channel coding. In this case, packets

using different code rates have a different number of source bits. In practice, either scheme may be favored depending on the situation. However, most UEP schemes address only one of the two cases.

In this chapter, we propose a joint source/channel coding scheme with unequal error protection for transmission of a JPEG2000 codestream over memoryless channels within a given total bit rate. The proposed scheme combines the forward error correction (FEC) capability provided by channel codes, together with the error detection and localization functionality provided by JPEG2000 error resilience (ER) tools [18] in an effective way. That is, FEC is used to reduce the bit error rate introduced by the channel, and ER tools are used to reduce the impact of any residual errors in the codestreams. During rate allocation, the proposed scheme utilizes the source distortion-rate characteristics and the error rate statistics of the available channel codes, with the goal to minimize the delivered image distortion subject to a total transmission rate. It forms quality scalable, error-resilient codestreams. When a priori knowledge about the channel is available, the proposed scheme provides performance competitive with that of other schemes known for JPEG2000 using similar channel codes. When the channel deviates from the assumed condition, the proposed scheme can provide more graceful quality degradation through robust source decoding. Furthermore, the proposed scheme can efficiently form either fixed-length source or channel packets with the same algorithm.

This chapter is organized as the following: Section 3.2 gives a general description of the error resilience tools provided by JPEG2000. Our joint source/channel coding algorithm is developed in Section 3.3. Section 3.4 provides further explanation and discussion of the algorithm. Section 3.5 presents the experimental results and Section 3.6 contains some concluding remarks.

3.2 JPEG2000 Error Resilience Mechanisms

A JPEG2000 codestream can be partitioned into a series of hierarchical structures, as shown in Fig. 3.1. Each codestream starts with a main header. It contains critical information for a decoder to decode the entire codestream correctly. The codestream following the main header can be partitioned into a single or multiple tile-streams, depending on whether the input image is divided into multiple tiles before the wavelet transform. Inside each tile-stream, there are one or several layers and each layer contains a certain number of packets. Each packet collects the compressed data from a precinct at a resolution level. Each precinct consists of some codeblocks from all subbands at a single resolution level and finally each codeblock contributes a certain number of coding passes to each packet.

During its encoding procedure, a JPEG2000 encoder typically computes a distortion-rate slope for every coding pass from each codeblock. This information may be used for rate allocation to control the creation of packets and layers. That being said, the standard does not specify any required rate allocation criterion or algorithm to be

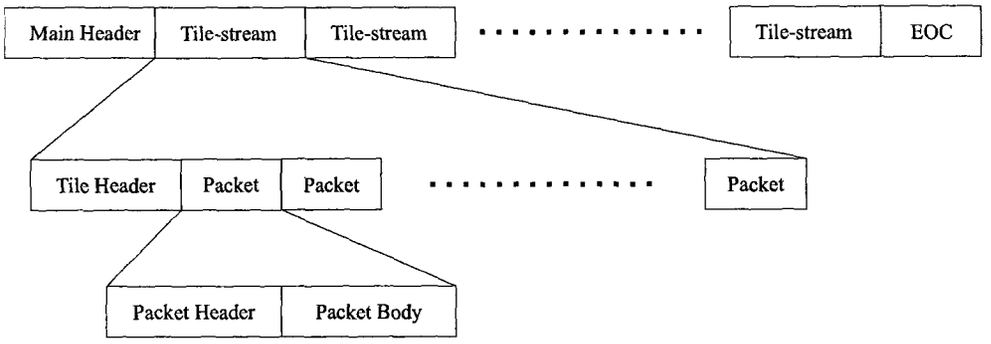


Figure 3.1: A simple JPEG2000 codestream.

used. The only requirement is that a compliant decoder should be able to decode any resulting codestream. This leaves tremendous flexibility for application-specific rate allocation systems.

JPEG2000 entropy coding is based on context-based arithmetic coding. The particular arithmetic coder employed is known as the MQ coder. It is crucial for the arithmetic encoder and decoder to maintain synchronization in order to correctly decode a codestream. A bit error in a codestream can easily destroy this synchronization, and lead to disastrous decompression effects if no special measures are taken. To combat this problem, JPEG2000 provides some useful mechanisms for error resilience. They generally serve two purposes: (1) hierarchical data partitioning and resynchronization, (2) error detection and isolation [18], [49].

The JPEG2000 error resilience tools are built upon the structures introduced above. The smallest independent coding unit is the codeblock. Bit errors will not propagate from one codeblock to another as long as codeblock synchronization is maintained. As mentioned above, codeblock data are collected into packets. A

packet consists of a packet header and a varying number of bytes from each appropriate codeblock. With correct packet header information, a JPEG2000 decoder can determine the number of bytes corresponding to each codeblock. Thus, even though the data for a codeblock may be damaged (within a packet body), the decoder can reestablish synchronization with any undamaged codeblocks.

Even when an error is confined to a single codeblock, it is desirable to maximize the decoded quality within that codeblock. To this end, JPEG2000 provides a set of mechanisms for detecting and isolating errors to a single coding pass. All previous coding passes within the codeblock can then be decoded correctly.

In its "default mode," a JPEG2000 bitplane coder produces one contiguous arithmetic codeword for each codeblock. In this mode, even if a JPEG2000 decoder can detect that an error is present in a codeword, there is little hope for the decoder to determine the location of this error. In general then, the decoder has to discard the whole codeword for that codeblock. JPEG2000 provides some mode variations to improve on this situation. These alternative modes allow additional error resilience capabilities in exchange of a compression efficiency loss. Mode variations are controlled by the flags signaled in the headers. Table 3.1 lists the relevant error resilience modes provided by JPEG2000.

When the *RESET* mode is used, the context states (i.e., the probabilities used in arithmetic coding) are reset to their initial values at the end of each coding pass.

Table 3.1: Error resilience mode variations in JPEG2000.

Switch	Description
RESET	reset context states
RESTART	terminate and restart MQ coder
ERTERM	predictable termination
SEGMARK	segmentation marker

If the RESET switch is not specified, this initialization occurs only prior to the first coding pass of a codeblock.

When the *RESTART* mode is used, the MQ coder is restarted at the end of each coding pass. Specifically, the MQ codeword is appropriately terminated and the MQ coder is re-initialized (excluding its probability models). Each coding pass is then represented by a separate MQ codeword segment. The length of each such segment is explicitly signaled in the relevant packet header.

When the *ERTERM* mode is used, a predictable termination policy is employed by the MQ coder for each codeword segment. A decoder can exploit the property of this termination policy to detect errors which may exist in an MQ codeword segment.

When the *SEGMARK* mode is used, a string of four binary symbols are encoded at the end of each bit-plane. A decoder must be careful to consume these four symbols before proceeding to the next bit-plane. An error resilience implementation of the decoder may use *SEGMARK* symbols to detect the presence of errors and take measures to conceal the error effects.

The modes *RESTART* and *ERTERM* are employed in the proposed algorithm. When they are used in concert, an encoder creates a separate, predictably terminated codeword segment for each coding pass. If an error occurs in the codestream, it is very likely that a decoder will terminate in a state that is inconsistent with the predictable termination policy. Therefore, a decoder can detect the presence of the error at the end of the coding pass in which it occurred. With the length information signaled in the packet headers, the decoder can discard only the corrupt coding pass (and all subsequent coding passes from the same codeblock), thereby concealing the possible visual artifacts which would otherwise result from such corruption, at the same time maximizing the decoded quality within that codeblock.

In addition to the mechanisms discussed above, JPEG2000 provides a useful functionality to relocate packet headers from their respective packet bitstreams to the main (or tile) header. This is referred to as *packed packet headers*. Correct decoding of a packet header is not only crucial to decoding its associated packet, but also for all subsequent packets within the same precinct. For this reason, it is important to receive these packet headers correctly. Collecting all packet headers into the main header facilitates special protection for them.

3.3 Joint Source/Channel Coding Algorithm

The general problem of joint source/channel coding can be described as follows: given source and channel coders, channel conditions, and resource constraints (such

as total transmission rate, energy, buffer size, etc.), how can we optimally distribute the available resource(s) between the source and channel coders? In this chapter, we treat the situation in which a JPEG2000 coded image is transmitted through a discrete memoryless channel within a given total bit rate, and the minimum delivered image distortion is desired.

3.3.1 Problem Formulation

Without considering channel noise, only distortion due to image quantization is of issue. For JPEG2000, the problem is to minimize the sum of distortions contributed from all codeblocks, while keeping the sum of their codestream lengths within the designated total length:

$$\min \sum_b D_b \qquad \text{s.t.} \qquad \sum_b L_b \leq L_T \tag{3.1}$$

where D_b and L_b are the distortion and codestream length of codeblock b, respectively, and L_T is the codestream length corresponding to a designated total bit rate.

When channel noise is taken into account, expected end-to-end distortion is the relevant metric, since distortion becomes a function of both quantization and channel noise. Let V be a vector with components V_b specifying the rate allocation for codeblock b. That is, V_b specifies the number of source and parity bytes allocated to codeblock b. More specifically, it specifies the number of coding passes included, and the channel code rate employed for each such coding pass. Codeblock b then has

expected distortion given by:

$$E[D_b(V_b)] = D_{b,0} - E[\Delta D_b(V_b)] \tag{3.2}$$

where $D_{b,0}$ is the zero rate distortion, $E[\Delta D_b(V_b)]$ is the expected distortion reduction when V_b is employed, resulting in a length L_b bitstream (including parity bits) for codeblock b.

In the noisy channel case, the problem then becomes finding V to solve:

$$\min\{\sum_b (D_{b,0} - E[\Delta D_b(V_b)])\} \qquad \text{s.t.} \qquad \sum_b L_b(V_b) \leq L_T$$

or equivalently,

$$\min\{\sum_b -E[\Delta D_b(V_b)]\} \qquad \text{s.t.} \qquad \sum_b L_b(V_b) \leq L_T \tag{3.3}$$

Similar to [50], we may employ the Lagrange multiplier method:

$$\min\{\sum_b -E[\Delta D_b(V_b)] + \lambda \sum_b L_b(V_b)\} \tag{3.4}$$

For a given $\lambda \geq 0$, the solution to (3.4) can be obtained by solving each term (corresponding to b) independently. By sweeping λ over the range of zero to infinity, sets of $\{V(\lambda)\}$ and $\{\sum_b L_b(\lambda)\}$ can be created. If a $(\sum_b L_b)$ happens to equal L_T, then a desired rate allocation scheme V has been found. As discussed below, bisection can be employed to determine the desired λ in an efficient manner.

Minimizing each term in (3.4) corresponds to an optimization task at the coding pass level within a single codeblock. For codeblock b, denote N_c as the maximum

number of coding passes that may be included. For each coding pass i, there is an associated distortion reduction d_i, with length l_i bytes (not including parity), where $i \in \{1, 2, \ldots, N_c\}$. From (3.4), for a given λ, it is desired to find a rate allocation scheme V_b which minimizes:

$$-E[\Delta D_b(V_b)] + \lambda L_b(V_b) \tag{3.5}$$

Suppose a given channel codec provides a family of codes with different error correction capabilities and code rates. Denote r_i as the channel code rate assigned for coding pass i ($0 \leq r_i \leq 1$) and denote $P(r_i, \frac{l_i}{r_i})$ as the probability that there are one or more uncorrected errors after channel decoding for coding pass i. Note that $\frac{l_i}{r_i}$ is then the length of coding pass i including parity. Finally, let $N_c^{'}$ ($N_c^{'} \leq N_c$) be the number of coding passes included by V_b. As mentioned above, we employ JPEG2000 error resilience mode switches *RESTART* and *ERTERM*. In this way, a JPEG2000 decoder can decode all correct coding passes within any given codeblock prior to the one containing the first detected bit error. Decoding of other codeblocks is unaffected. With this condition, the expected distortion reduction for codeblock b including $N_c^{'}$ coding passes using channel code rates $r_1, r_2, \ldots, r_{N_c^{'}}$ is:

$$\begin{aligned} E[\Delta D_b(V_b)] \;=\; & \sum_{j=1}^{N_c^{'}-1} \Big(\sum_{k=1}^{j} d_k\Big)\Big(\prod_{k=1}^{j}[1 - P(r_k, \frac{l_k}{r_k})]\Big)P(r_{j+1}, \frac{l_{j+1}}{r_{j+1}}) \\ & + \Big(\sum_{k=1}^{N_c^{'}} d_k\Big)\Big(\prod_{k=1}^{N_c^{'}}[1 - P(r_k, \frac{l_k}{r_k})]\Big) \end{aligned} \tag{3.6}$$

With (3.5) and (3.6), the objective function to be minimized for each codeblock b becomes:

$$\begin{aligned} f(V_b) &= -\sum_{j=1}^{N_c'-1}(\sum_{k=1}^{j} d_k)(\prod_{k=1}^{j}[1-P(r_k,\frac{l_k}{r_k})])P(r_{j+1},\frac{l_{j+1}}{r_{j+1}}) \\ &\quad -(\sum_{j=1}^{N_c'} d_j)(\prod_{j=1}^{N_c'}[1-P(r_j,\frac{l_j}{r_j})]) + \lambda\sum_{j=1}^{N_c'}\frac{l_j}{r_j} \end{aligned} \tag{3.7}$$

To minimize this expression, we must jointly optimize the number of included coding passes N_c' and the channel code rates $r_1, r_2, \ldots, r_{N_c'}$. The next two subsections discuss procedures to accomplish this goal.

3.3.2 A Constrained Exhaustive Search Based Approach

Since (3.7) can be evaluated for any rate allocation scheme $V_b = (r_1, r_2, \ldots, r_{N_c'})$, one possible way to find an optimal V_b is by exhaustive search. By enumerating all possible V_b and substituting into (3.7), an optimal scheme V_b^* can be determined. That being said, suppose there are R channel coding rates available. If N_c' coding passes are included for a given codeblock, there are $R^{N_c'}$ possible V_b for that codeblock. So, for an N_c-coding pass codeblock b, there are $\sum_{i=1}^{N_c} R^i$ rate allocation schemes constituting the search space. Generally, without imposing any constraints, this space is too large to search directly.

In [39], it was shown that the optimal protection level decreases along an embedded bitstream. This can be applied to codestreams from codeblocks in our case and it implies $r_i \leq r_{i+1} (1 \leq i \leq N_c')$. Also, from [39] (and other results from the

literature), at most three protection levels are usually enough to obtain most UEP gains for binary symmetric channels with error probability $\epsilon \leq 10^{-1}$. With these two conditions, the search space is significantly reduced, often admitting an exhaustive search to find an optimal rate allocation scheme V_b^*.

3.3.3 A Dynamic Programming Based Approach

In the subsection above, a constrained exhaustive search based approach is proposed to find V_b^* over a reduced search space. Even though feasible, the resulting search is still computationally intense. Also, when the number of available channel code rates are large or the channel condition assumption is not satisfied, the exhaustive search approach is prohibitive. In this section, a dynamic programming optimization is proposed, which aims to reduce the computational complexity in solving (3.7). The notation is based on that in [51].

By proper rearrangement, (3.7) can be written as:

$$\begin{aligned} f(V_b) &= \sum_{i=1}^{N_c'} \left\{ (\prod_{j=1}^{i} [1 - P(r_j, \frac{l_j}{r_j})])(-d_i) \right\} + \lambda \sum_{i=1}^{N_c'} \frac{l_i}{r_i} \\ &= \sum_{i=1}^{N_c'} \left\{ (\prod_{j=1}^{i} [1 - P(r_j, \frac{l_j}{r_j})])(-d_i) + \lambda \frac{l_i}{r_i} \right\} \end{aligned} \tag{3.8}$$

In light of (3.8), for the purpose of backward dynamic programming, the cost function at stage k, state r_k, is defined as:

$$g_k(r_k) = [1 - P(r_k, \frac{l_k}{r_k})](-d_k) + \lambda \frac{l_k}{r_k} \tag{3.9}$$

where stage k corresponds to the coding pass k ($1 \leq k \leq N_c$) and state r_k belongs to the set of available channel code rates at stage k denoted as $\mathbb{R}(k)$.

The cost-to-go function at stage k, state r_k is defined as:

$$J_{N_c}(r_{N_c}) = g_{N_c}(r_{N_c}) \qquad (\text{when } k = N_c)$$

and

$$\begin{aligned} J_k(r_k) = g_k(r_k) + \min_{r_{k+1} \in \mathbb{R}(k+1)} \{ & [1 - P(r_k, \frac{l_k}{r_k})][J_{k+1}(r_{k+1}) - \lambda(\frac{l_{k+1}}{r_{k+1}} + \sum_{p=k+2}^{N_c} \frac{l_p}{r_p^*})] \\ & + \lambda(\frac{l_{k+1}}{r_{k+1}} + \sum_{p=k+2}^{N_c} \frac{l_p}{r_p^*}) \} \\ & (\text{when } 1 \leq k \leq N_c - 1) \end{aligned} \tag{3.10}$$

where

$$r_p^* = \begin{cases} \underset{r_{k+2} \in \mathbb{R}(k+2)}{\text{argmin}} \; J_{k+1}(r_{k+1}) & \text{when } p = k+2 \\ \underset{r_p \in \mathbb{R}(p)}{\text{argmin}} J_{p-1}(r_{p-1}^*) & \text{when } k+3 \leq p \leq N_c. \end{cases} \tag{3.11}$$

Note that for the cost-to-go function J_k at each stage k, in addition to the states corresponding to the available channel codes, we include a value-0 entry indicating the possible choice that only the first $k-1$ coding passes are included by V_b. So $V_b = (r_1, r_2, \ldots, r_{N_c'})$ where N_c' is the last coding pass index with non-zero channel code rate.

For each codeblock b and with any given λ, the proposed approach starts from the last possible coding pass that can be included in the codestream and proceeds backwards to the first coding pass. For each coding pass (or stage), it computes the

optimal cost-to-go using (3.10) for each available channel code rate (or state). The minimal cost associated with a state in the first stage becomes the optimal cost, and the path which leads to this state from the last stage given by (3.11) determines the optimal source and channel rate allocation for this codeblock.

Without any code rate constraint, each state can choose any state from the previous stage, with complexity of $O(R^2)$. The non-decreasing channel code rate constraint discussed previously. can be imposed as $r_i \leq r_{i+1}$ $(1 \leq i \leq N_c)$, yielding even lower complexity.

3.3.4 Algorithm Summary

Based upon the discussion above, the proposed rate allocation scheme can be divided into two levels. At the lower level, it deals with coding passes in each codeblock. Given a $\lambda \geq 0$, it finds an optimal code rate vector V_b^* which minimizes (3.7), using either exhaustive search (Section 3.3.2) or dynamic programming (Section 3.3.3). This step will result in a codestream with length L_b for each codeblock b. At the higher level, a search over λ is performed to find λ^* such that $\sum L_b(\lambda^*)$ equals the desired total length. Because of the monotonic relationship between λ and the resulting codestream length, a bisection method can be utilized to find λ^* for a particular designated total bit rate.

Fig. 3.2 shows a block diagram of a JPEG2000 encoder containing the proposed joint source/channel rate allocation. The algorithm discussed in this section is embedded as a source rate allocator in the system. It utilizes the necessary source, channel and FEC statistic information to control the formation of a codestream. Just as PCRD [19] in the noiseless case, the proposed scheme can serve as an optional rate allocation algorithm for any JPEG2000-compliant encoder.

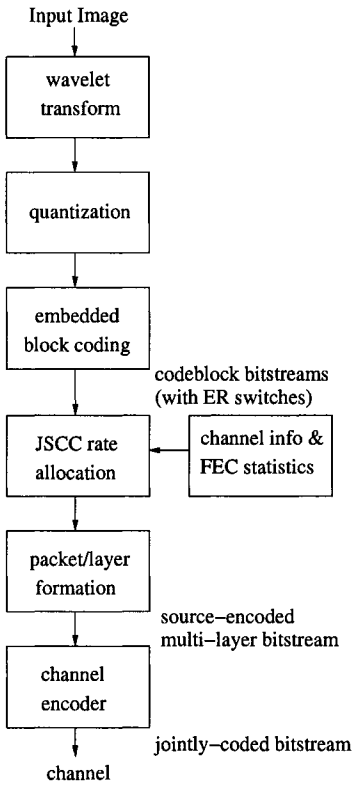


Figure 3.2: Joint source/channel coding (JSCC) system encoder diagram.

Each V_b^* determines, for a codeblock b, how many coding passes should be included in the final codestream, and which protection level should be applied to each of them. We can collect those coding passes which need the same protection level from their

individual codeblock codestreams and form them into one JPEG2000 quality layer. In addition, main headers and packet headers are crucial in correct decoding of the codestream. As discussed in Section 3.2, packed packet header functionality can be invoked to consolidate packet headers, which need to be strongly protected. In this way, a proper strength channel code can be easily applied to these critical data. In our simulation results below, these data were treated using the same channel code as the first (most important) quality layer.

Conceptually, a "usual" JPEG2000 quality layer contains the bits of the same (deterministic) importance in reconstructing the image, which is solely determined by the image source. By contrast, in our joint source/channel coding algorithm, the quality layers contain bits of the same importance in a probabilistic sense. The content of these quality layers is jointly determined by the image, channel condition and available channel codes.

3.4 Algorithm Discussion

With the use of the Lagrangian method in the higher level of the proposed scheme, there is the potential that we may not be able to meet a desired target rate requirement. This is due to the fact that we can only select from a finite number of points on the convex hull of the operational distortion-rate function of a particular image source. However, this problem is mostly remedied by the fine embedding capability provided by the entropy coding of JPEG2000 [19]. With this algorithm, it is possible

to efficiently compress relatively small codeblocks (say 64×64 or 32×32 samples each) independently, yielding an embedded codestream consisting of a large number of operational points (corresponding to coding passes), such that most of these operational points lie on the convex hull of the corresponding distortion-rate curve. With a densely populated convex hull, it becomes very unlikely that there is a large gap between the designated target rate and optimal solution rate that the proposed scheme can provide. This is born out by our experimental results in Section 3.5.

When the dynamic programming approach is employed, a term $[1 - P(r_k, \frac{l_k}{r_k})]$ of stage k is multiplied with the adjusted cost-to-go function J_{k+1} of stage $k+1$ (see (3.10)). This is because a JPEG2000 decoder must decode a codeblock's bitstream in a sequential and uninterrupted fashion. This is referred to as a "sequential decoding requirement" in [42]. With the presence of this multiplicative term, the dynamic programming based scheme cannot be guaranteed to provide a global minima in some cases. However, this can be alleviated by carefully choosing the channel codes, as mentioned in [45]. Furthermore, the possible suboptimality is localized to the codeblock level in our scheme. In Section 3.5, we will show that in most cases, dynamic programming and exhaustive search yield identical solutions. Even in the cases when they differ, the suboptimality incurred by dynamic programming is negligible. Thus, dynamic programming should be favored in most situations.

A practical issue arising in applications using unequal error protection is that it is often desirable to form transmission packets with either a fixed number of source

bits (fixed-k case), or a fixed number of channel-coded bits (fixed-n case) depending on the situation, as discussed in Section 3.1. Our proposed scheme provides the flexibility to handle these two cases with the same algorithm. This is because our algorithm depends only on the available channel code rates and the residual error properties ($P(r_j, \frac{l_j}{r_j})$), and not on whether the codewords have fixed k or fixed n. As shown in Fig. 3.2, an optimal multi-layer codestream will be generated by the source encoder. In either case, we simply divide the length of each layer by its k to get the number of channel codewords needed for this layer. In the fixed-k case, the value of k_i is constant for each layer i. In the other case, n is fixed and $k_i = n \times r_i$.

As introduced in Section 3.1, [45] proposed a rate allocation scheme based on the Viterbi algorithm (VA) for JPEG2000 codestreams with turbo codes. The optimization in that paper is done for an existing JPEG2000 codestream having many quality layers. It has a fixed-n codeword constraint and this constraint is translated to a fixed number of channel packets on which the VA is used to determine the optimal channel rate applied to each packet. The source decoding strategy employed there is to halt all decoding (source and channel) when the channel decoder detects the first uncorrectable codeword, and the remainder of the codestream is discarded. This procedure is based on the assumption that further decoding will cause error propagation and thus degrade the reconstructed image quality.

In contrast with [45], our rate allocation for unequal error protection is based on an error resilient codestream. The proposed scheme performs the optimization at the

level of coding passes within individual codeblocks and uses the rate allocation results to generate a codestream with multiple layers corresponding to different protection levels. In a jointly coded bitstream, each layer contains the bitstreams contributed from all the codeblocks to this layer and they are coded into a certain number of channel codewords. With the use of JPEG2000 error resilience modes in the codestream, a JPEG2000 decoder can detect and localize errors within a codeblock, and terminate further decoding only for that codeblock. Therefore, any uncorrectable channel codeword within a layer only affects some of the codeblocks, and the decoding of others can continue to the end of the codestream. As shown in the next section, this property is particularly useful when the actual channel condition mismatches that assumed when the rate allocation is performed. However, because of the use of JPEG2000 error resilience features, our scheme generally has fewer quality layers (less scalable). At the same time, it comes with a small cost in source coding efficiency due to the use of the error resilience modes.

3.5 Experimental Results

In the following experiments, we explore the transmission of JPEG2000 codestreams through noisy memoryless channels with different channel conditions and total bit rates. Rate allocation is performed with a priori knowledge about the channel. Further experiments are conducted for cases when the actual channel condition mismatches the one assumed during the encoding process. Kakadu is used as our

source coder in the experiments. All test images for the proposed scheme are generated with a 5-level (9,7) wavelet transform, a codeblock size of 64×64, ERTERM + RESTART modes turned on, and PPM marker segments employed.

In the first experiment, the effectiveness of the proposed rate allocation schemes based on exhaustive search (ES) and dynamic programming (DP) is compared. Rate compatible punctured convolutional (RCPC) codes are chosen as the channel codes for binary symmetric channel (BSC) with $\epsilon = 10^{-2}$. From [25], we selected the 4 code rates $\{1, \frac{8}{9}, \frac{4}{5}, \frac{2}{3}\}$, where rate 1 corresponds to the case of no channel coding. Since the channel condition satisfies the assumption required by ES and there are only 4 code rates to choose from, the comparison between ES and DP is possible.

We employ the same encoding/decoding structure as in [38]. Specifically, every 200 source bits are augmented by 16 CRC bits and passed through a rate r RCPC coder to form a channel codeword. At the decoder, list Viterbi decoding is performed with a depth of 100. Error statistics for these codes were simulated and tabulated for use in our rate allocation algorithm. Consistent with [38], we found that the rate $\frac{2}{3}$ RCPC code punctured from a memory 6, rate $\frac{1}{3}$ code yields an extremely small probability of uncorrected error for the channel.

Goldhill (512×512) and Lenna (512×512) are used as our test images in this experiment. Optimal rate allocation is performed at total bit rates of 0.10, 0.25, 0.50, 0.75 and 1.00 bpp, respectively. Each bit rate was simulated with 1000 trials. The reported mean PSNR values were calculated by averaging the decoded MSE values

and then converting the mean MSE to the corresponding PSNR values. During the rate optimization, the non-decreasing code rate constraint ($r_i \leq r_{i+1}$) is imposed to simplify the final codestream structure with negligible loss. The results from both ES and DP based approaches are compared in Tables 3.2 and 3.3. In both cases, rate $\frac{2}{3}$ and $\frac{4}{5}$ codes are always preferred over the other two weaker codes. In other words, in each codeblock a certain number of coding passes are selected to be protected using the rate $\frac{2}{3}$ code, with some remaining coding passes protected by the rate $\frac{4}{5}$ code. As discussed in Section 3.4, a 2-layer codestream is then formed corresponding to these two different protection levels. Also the main header is well protected by the rate $\frac{2}{3}$ code together with the first layer. The fraction of data belonging to each layer, the source rates, the noise-free and noisy PSNR performance are all compared between the ES and DP approaches. In all these cases, DP yields identical results to ES, with exceptions at 1.00 bpp for both images, incurring negligible difference in these cases. At the same time, the computation time required with 4 available RCPC rates is reduced by a factor of one hundred from ES to DP. For more than 4 code rates, ES is essentially unmanageable. Thus it is reasonable to prefer the DP rate allocation for a good tradeoff between performance and complexity. So, in the following experiments with turbo codes, which provide more code rates to choose from, only the DP approach is employed.

In the following experiments, we use stronger rate compatible punctured turbo codes (RCPT) [25] as the channel codes and only DP rate allocation is considered.

Table 3.2: ES vs. DP rate allocation results for Goldhill (512 × 512).

Bit Rate (bpp)	0.1	0.25	0.50	0.75	1.00
ES RCPC: $\frac{2}{3}$, $\frac{4}{5}$ rates layer length ratio	1:5.27	1:5.39	1:5.46	1:4.05	1:3.93
DP RCPC: $\frac{2}{3}$, $\frac{4}{5}$ rates layer length ratio	1:5.27	1:5.39	1:5.46	1:4.05	1:3.93
ES RCPC: Source Rate(bpp)	0.08	0.19	0.36	0.53	0.69
noise-free PSNR(dB)	26.78	29.30	31.63	33.26	34.42
DP RCPC: Source Rate(bpp)	0.08	0.19	0.36	0.53	0.70
noise-free PSNR(dB)	26.78	29.30	31.63	33.26	34.45
UEP ES Noisy PSNR(dB)	26.73	29.19	31.46	33.08	34.21
UEP DP Noisy PSNR(dB)	26.73	29.19	31.46	33.08	34.21

Table 3.3: ES vs. DP rate allocation results for Lenna (512 × 512).

Bit Rate (bpp)	0.1	0.25	0.50	0.75	1.00
ES RCPC: $\frac{2}{3}$, $\frac{4}{5}$ rates bitstream length ratio	1:4.06	1:4.31	1:2.95	1:3.31	1:2.82
DP RCPC: $\frac{2}{3}$, $\frac{4}{5}$ rates bitstream length ratio	1:4.06	1:4.31	1:2.95	1:3.31	1:2.81
ES RCPC: Source Rate(bpp)	0.08	0.18	0.35	0.52	0.70
noise-free PSNR(dB)	28.16	32.06	35.30	37.21	38.46
DP RCPC: Source Rate(bpp)	0.08	0.18	0.35	0.52	0.70
noise-free PSNR(dB)	28.16	32.06	35.30	37.21	38.45
UEP ES Noisy PSNR(dB)	28.06	31.90	35.13	37.01	38.27
UEP DP Noisy PSNR(dB)	28.06	31.90	35.13	37.01	38.25

Lenna (512×512) is chosen as our test image. Simulations are conducted for the BSC with bit error rates of $\epsilon = 0.01$, 0.08 and 0.1, at total bit rates of 0.25, 0.50 and 1.00 bpp respectively. Both fixed-k and fixed-n cases are considered.

We employ a rate $\frac{1}{3}$ parallel concatenated convolutional coder with generator polynomial $(33/31)_{oct}$ as our channel encoder, and a BCJR-MAP decoder using a maximum of 20 iterations. An S-random interleaver [52] is used with $s = \lfloor \frac{1}{2}\sqrt{N/2} \rfloor$, where N is the number of message bits plus memory flush bits in a codeword. Our puncturing table is chosen from [28] with puncturing period 8. This provides a set of code rates as follows: $\{\frac{8}{9}, \frac{4}{5}, \frac{8}{11}, \frac{2}{3}, \frac{8}{13}, \frac{4}{7}, \frac{8}{15}, \frac{1}{2}, \frac{8}{17}, \frac{4}{9}, \frac{8}{19}, \frac{2}{5}, \frac{8}{21}, \frac{4}{11}, \frac{8}{23}, \frac{1}{3}\}$. In the fixed-$k$ case, each channel codeword contains 500 source bytes, followed by 4 extra bytes. These 504 bytes are then encoded by a rate r RCPT channel coder to form a channel codeword with size $\frac{504}{r}$ bytes. The 4 extra bytes can be used for CRC, and/or to transmit side information such as channel code rates, packet lengths, etc. In our experiments, these bytes are used for CRC-32 with generator polynomial $(40460216667)_{oct}$. The error detection capability thus provided is used to accelerate decoding, i.e., whenever the CRC succeeds in an iteration, the turbo decoder halts processing for that codeword.

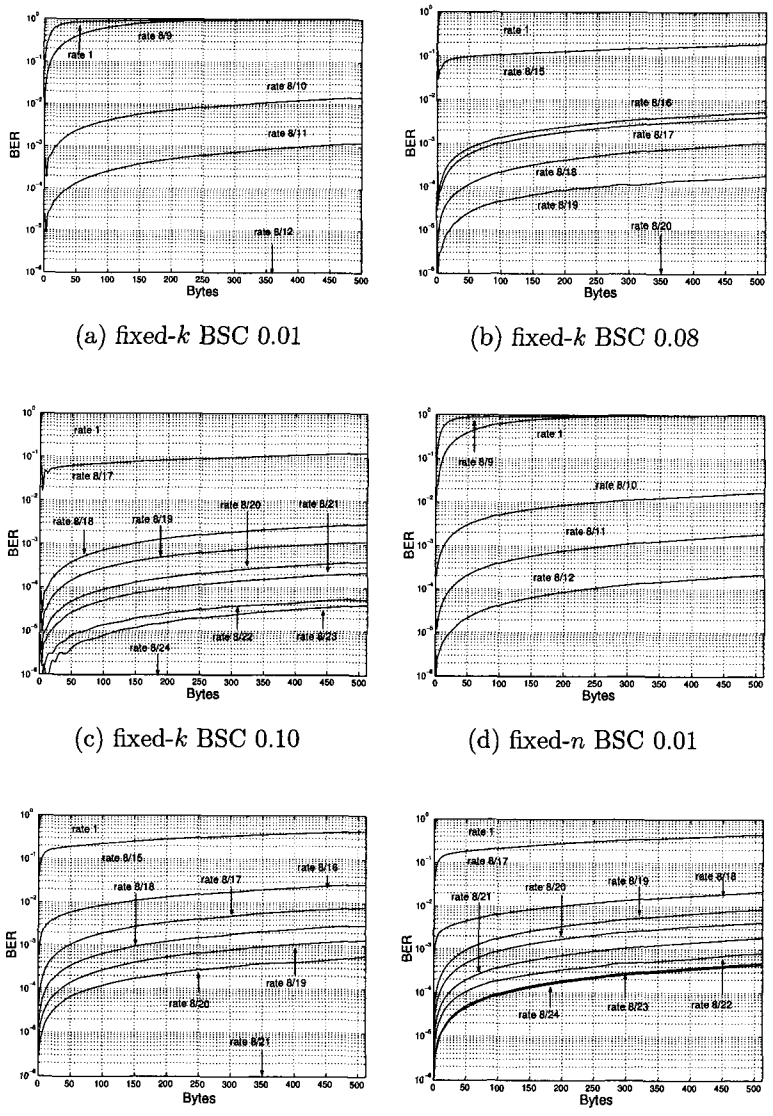
In the fixed-n case, each channel codeword has a fixed length of 500 bytes. The number of source bytes inside each codeword is calculated such that after appending 4 CRC bytes and passing through a rate r RCPT channel encoder, a 500-byte channel codeword will result. The number of source bytes inside each codeword according to

different code rates are then:

{440,395,359,328,303,281,262,245,230,217,206,195,186,177,169,162}. The final code rates (including the extra 4 bytes and 4 memory flushing bits) are calculated for both the fixed-k and fixed-n cases.

As in the RCPC case, the error statistics of our codes were simulated and tabulated. Fig. 3.3 shows the coding pass error rates $P(r_i, \frac{l_i}{r_i})$, which are used in our rate allocation scheme. These values are obtained by examining the residual error statistics, i.e., the probability of one or more residual bit errors in l_i consecutive bytes of decoded source data when they are protected with channel code rate r_i. Because turbo code performance improves as codeword length increases, with the same code rate, the channel code strengths are different for the fixed-k and fixed-n cases (fixed-k is stronger than fixed-n, as can be seen from Fig. 3.3 with the difference becoming more noticeable as code rate decreases.

Different codestreams are generated using the DP approach for different target bit rates, channel conditions, and both fixed-k and fixed-n cases. For the fixed-n case, we have channel packets with a fixed length of 500 bytes. At target bit rates {1.00, 0.50, 0.25}(bpp) for 512 $\times$ 512 8-bit images, {66, 33, 17} channel codewords are formed for each image, corresponding to total bit rates of {1.0070, 0.5035, 0.2594}(bpp) respectively. In the fixed-k case, the final total bit rate can vary somewhat from image to image. However, the final total bit rate is always very close to the designated total rate.



(a) fixed-k BSC 0.01

(b) fixed-k BSC 0.08

(c) fixed-k BSC 0.10

(d) fixed-n BSC 0.01

(e) fixed-n BSC 0.08

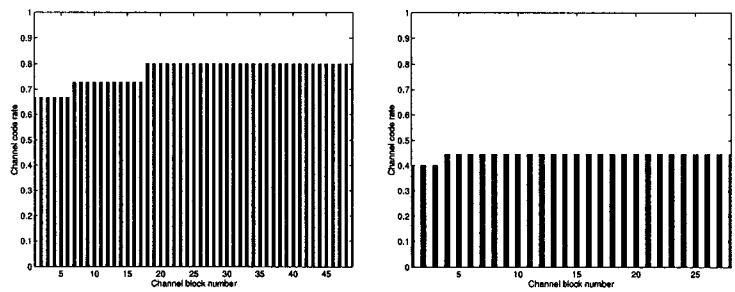
(f) fixed-n BSC 0.10

Figure 3.3: Coding pass error rate ($P(r_i, \frac{l_i}{r_i})$) vs. coding pass length.

The joint rate allocation for each case results mostly in only a few channel code rates being chosen from the many possibilities. In order to form an integer number of channel codewords for each chosen code rate, the following rules are applied. In the case when some code rate is selected so that the coding passes it protects do not fill even a single channel codeword, this code rate is de-selected and the coding passes are assigned to neighboring selected rates. In the case when the coding passes chosen with a code rate do not form an integer number of channel codewords, simple rounding off is applied. This either leaves some bytes to the next layer, or includes some bytes from the next layer. These procedures incur negligible loss in optimality.

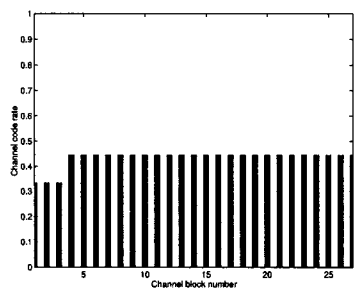
With the use of dynamic programming, the computational complexity is greatly reduced compared to exhaustive search. With the source and channel coding parameters used in the paper, the computational time for the entire image with each λ is about 1.2 seconds on a Pentium III 1GHz PC. In practice, one to several λ values need to be tried to obtain the optimal solution. Since the optimization for each codeblock (also for λ) is totally independent, they can be processed in parallel if desired.

The rate allocation results from the proposed DP-based scheme are as follows. In the fixed-k case, the code rate sets $\{\frac{2}{3}, \frac{8}{11}, \frac{4}{5}\}$, $\{\frac{2}{5}, \frac{4}{9}, \frac{1}{2}\}$ and $\{\frac{1}{3}, \frac{4}{9}\}$ are chosen for BSC $\epsilon = 0.01$, 0.08 and 0.1, respectively (except for the $\epsilon = 0.08$ at 1.00 bpp case, where $\{\frac{2}{5}, \frac{4}{9}\}$ is chosen instead.) In the fixed-n case, the code rate sets $\{\frac{2}{3}, \frac{8}{11}, \frac{4}{5}\}$, $\{\frac{8}{21}, \frac{8}{17}, \frac{1}{2}\}$ and $\{\frac{1}{3}, \frac{8}{19}\}$ are chosen for BSC $\epsilon = 0.01$, 0.08 and 0.1, respectively (except for the $\epsilon = 0.08$ at 1.00 bpp case, where $\{\frac{8}{21}, \frac{8}{17}\}$ is chosen instead.) Corresponding numbers

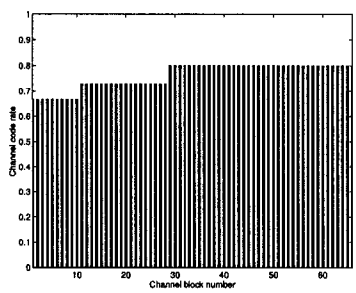


(a) BSC $\epsilon = 0.01$, fixed-k

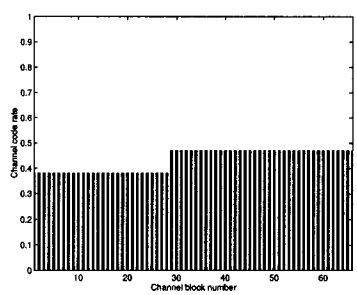
(b) BSC $\epsilon = 0.08$, fixed-k



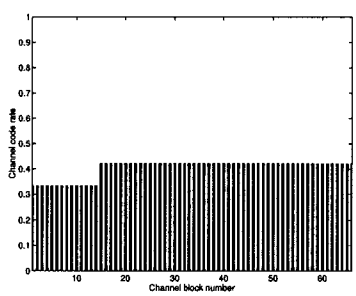
(c) BSC $\epsilon = 0.1$, fixed-k



(d) BSC $\epsilon = 0.01$, fixed-n



(e) BSC $\epsilon = 0.08$, fixed-n



(f) BSC $\epsilon = 0.1$, fixed-n

Figure 3.4: Lenna (512×512) channel code rate assignments at 1.00 bpp.

of quality layers are generated according to the number of channel code rates selected for each codestream. Similar to the RCPC case, the main header is protected with the same channel code rate as the first layer, which is always the strongest code in the chosen set and yields very low probability of error. As examples, Fig. 3.4 shows the optimal rates assigned to the different channel codewords of the generated codestreams for a BSC with $\epsilon = 0.01, 0.08, 0.1$ for both the fixed-k and fixed-n cases at 1.00 bpp, respectively.

Simulation results for the proposed rate allocation scheme are presented in Tables 3.4 to 3.6. As in the RCPC case, the reported values were calculated by converting the average decoded MSE values to PSNR, which is consistent with [53] (but not [45]). For each case, 10000 trials were conducted. The source coder and channel codes used in [45], [53] are both different from ours. For a fair comparison, we have implemented their algorithm using our source and channel codes. These results are listed in the same tables with our results, and the original results from [53] are also listed for reference [1] .

In these tables, the results from our rate allocation are termed as "ER-UEP" (error-resilient unequal error protection). Two results are reported for each ER-UEP case: one is to allow the decoder to decode the entire corrupted codestream, which corresponds to the "Continue" case; the other is to stop the decoding whenever the

[1]The column in the tables labeled "UEP-based on [45], [53]" is our simulation of their system, but using Kakadu and the channel codes employed in our system. The column labeled "UEP from [53]" contains the results as copied from [53].

Table 3.4: End-to-end PSNR performance: Lenna 512×512 at BSC $\epsilon = 0.01$.

BSC 0.01	Fixed-k		Fixed-n		UEP based on [45], [53] (Fixed-n)	UEP from [53] (Fixed-n)
	EEP	ER-UEP	EEP	ER-UEP		
	Default/ER	Continue (Halt)	Default/ER	Continue (Halt)		
total bit rate (bpp)	0.994	0.994	1.007	1.007	1.007	0.994
noisy PSNR (dB)	38.43/38.27	38.76 (36.25)	38.47/38.33	38.69 (35.96)	38.61	38.62
total bit rate (bpp)	0.496	0.496	0.504	0.504	0.504	0.505
noisy PSNR (dB)	35.30/35.11	35.63 (33.95)	35.33/35.15	35.57 (33.89)	35.55	35.61
total bit rate (bpp)	0.249	0.249	0.259	0.259	0.259	0.252
noisy PSNR (dB)	32.12/31.94	32.47 (31.51)	32.30/32.10	32.47 (31.97)	32.57	32.47

Table 3.5: End-to-end PSNR performance: Lenna 512×512 at BSC $\epsilon = 0.08$.

BSC 0.08	Fixed-k		Fixed-n		UEP based on [45], [53] (Fixed-n)	UEP from [53] (Fixed-n)
	EEP	ER-UEP	EEP	ER-UEP		
	Default/ER	Continue (Halt)	Default/ER	Continue (Halt)		
total bit rate (bpp)	1.000	1.000	1.007	1.007	1.007	0.994
noisy PSNR (dB)	36.09/35.98	36.31 (35.93)	35.89/35.75	36.13 (35.10)	35.96	35.79
total bit rate (bpp)	0.499	0.499	0.504	0.504	0.504	0.505
noisy PSNR (dB)	32.83/32.73	33.59 (32.56)	32.64/32.55	33.25 (32.13)	33.20	32.96
total bit rate (bpp)	0.249	0.249	0.259	0.259	0.259	0.252
noisy PSNR (dB)	29.82/29.59	30.46 (30.05)	29.59/29.34	30.25 (29.89)	30.12	29.88

Table 3.6: End-to-end PSNR performance: Lenna 512×512 at BSC $\epsilon = 0.1$.

BSC 0.1	Fixed-k		Fixed-n		UEP based on [45], [53] (Fixed-n)	UEP from [53] (Fixed-n)
	EEP	ER-UEP	EEP	ER-UEP		
	Default/ER	Continue (Halt)	Default/ER	Continue (Halt)		
total bit rate (bpp)	0.994	0.994	1.007	1.007	1.007	0.994
noisy PSNR (dB)	35.30/35.11	36.14 (34.85)	35.25/35.10	35.74 (33.58)	35.57	35.76
total bit rate (bpp)	0.503	0.503	0.504	0.504	0.504	0.504
noisy PSNR (dB)	32.16/31.97	33.02 (32.77)	32.08/31.82	32.63 (31.40)	32.54	32.72
total bit rate (bpp)	0.252	0.252	0.259	0.259	0.259	0.252
noisy PSNR (dB)	29.03/28.80	29.81 (29.52)	29.21/29.01	29.64 (29.17)	29.69	29.39

channel decoder detects a channel decoding failure, which corresponds to the "Halt" case in the parentheses. From the tables it can be seen that significant gains are achieved by continuing the decoding.

In general, the length of the source codestream generated from the proposed algorithm affects the gain between "Halt" and "Continue." With a longer source codestream, there is a higher probability of more decodable source bits that follow any corrupted source segment. Additionally, the relative length between a quality layer and the message bits of the channel codewords (selected to protect this layer) affects this gain. Suppose after the rate allocation, each layer is protected by a number of channel codewords (as shown in Fig. 3.4, each layer is protected by several to several tens of channel codewords.) Thus, each corrupted codeword only affects part of the codeblocks in its layer. The decoding of those correct codeblocks in the same and the following layers is unaffected. On the other hand, suppose a layer is protected by long codewords such that the length of their message bits is close to that of the layer. Then a channel decoding failure will likely corrupt most of the codeblocks in the layer and also result in most of the following layer(s) being discarded. Hence there is little gain obtainable by continuing decoding in this case.

The factors discussed above explain the shrinkage in gain between the "Continue" and "Halt" results for lower bit rates, as well as for channels with higher BERs in the tables. With a fixed total transmission rate, more bits are allocated to parity for a channel with a higher error rate and thus the corresponding source bit rate is

effectively reduced. At the same time, with a reduced source bit rate, all quality layers in a source codestream generally get shorter so that a layer contains a fewer number of channel codewords.

Based on the results from Tables 3.4 through 3.6, we note that our scheme ("Continue") is comparable to that based on [45] (in the fixed-n case). In the fixed-k case, our results are also comparable to the results based on [45] for $\epsilon = 0.01$. For the other two channel conditions, our results are better by about 0.35 dB on average (which is mainly due to the fact that the channel codes become stronger.) Our ER-UEP "Halt" results are worse than those based on [45]. This is expected because given any two prefixes of the same length from codestreams generated by our algorithm and [45] respectively, ours is less quality scalable and has a larger main header that needs to be well protected.

Tables 3.4 through 3.6 also include the performance for EEP with our turbo codes, to demonstrate our UEP gain. For this experiment, two different source codestreams are generated such that they have the same total bit rate after channel coding. The first codestream uses default parameter settings (termed "Default"), while the second codestream employs ERTERM+RESTART (termed "ER") as with the proposed scheme. In both codestreams, only one layer is used. The codestreams are then protected with the strongest codes chosen in the corresponding UEP cases. Since the probability of decoding failure is extremely low for the chosen channel code, the small difference in performance between the two cases is due to the small decrease in source

coding efficiency caused by the ER modes. As shown in the tables, our UEP scheme can provide 0.33 to 1.05 dB gain over the two EEP schemes.

In Fig. 3.5 the scaled minimum values of (3.8) for Lenna are shown for both RCPC and turbo codes at BSC $\epsilon = 0.01$. Turbo codes yield improvement over the entire λ range as a result of stronger protection. Therefore it can be expected that with the proposed scheme, better performance can be achieved with even stronger channel codes.

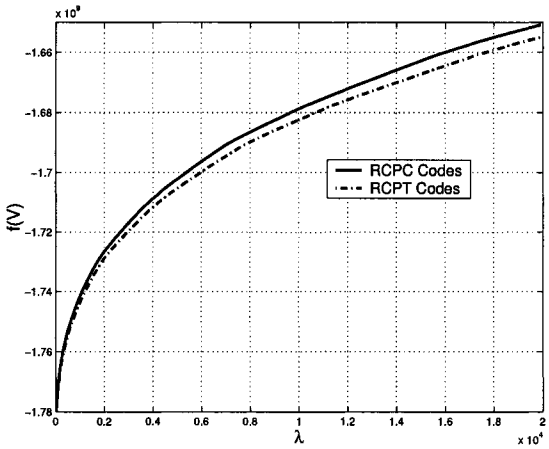


Figure 3.5: Optimal minimum values of RCPC and turbo codes for Lenna (512×512) at BSC $\epsilon = 0.01$.

The advantage brought by the use of turbo codes comes with a cost that the codestream can only be decoded at an integer number of source bytes as contained in each channel codeword. This is a general problem when long channel codes are used, and has been studied further in [42]. Additionally, the upper bound of the gap between an arbitrary designated total bit rate and the closest rate achievable by the turbo codes employed is determined by half of the shortest codeword length. In this

chapter, for the 3 different binary symmetric channels considered, with fixed-n, the upper bounds are all 7.63×10^{-3} bpp. For fixed-k, they are 9.63×10^{-3}, 0.017 and 0.018 bpp respectively. We also note that our scheme includes only a small number of layers in the codestream. The resulting codestream is then not as scalable as that in [45]. It is still possible to decode at any desired rate (within the accuracies given above). This can be accomplished by partially decoding a layer. However, the resulting image quality will not be as good (at least in the noise free case) as that obtained if more layers had been used.

In the experiments discussed so far, we assume that we know the correct channel state information when we perform the optimal rate allocation. From the discussion above, compared to the highly progressive approach employed by [45], our approach produces less scalable, but more error resilient JPEG2000 codestreams for the purpose of JSCC. Although the JPEG2000 error resilience switches come with a cost in source efficiency, this is mostly offset by the more robust source decoding, as demonstrated by the experimental results. At the same time, the resulting codestreams from our approach have the advantage of being more error resilient in the case when channel state information may not be accurately known.

In the next experiment, we consider the following situation: in addition to bit errors introduced by BSC, the codestreams are subjected to random channel codeword losses, which are not taken into consideration when the rate allocation is performed. This situation can rise when some bursty errors randomly occur within a channel

codeword transmission period, causing severe channel codeword decoding failure. It may also happen when the codestream is sent through a memoryless packet erasure channel concatenated with the BSC, where each channel codeword is sent as a packet.

Comparison is made between our ER-UEP based scheme and the highly scalable based scheme at a total bit rate of 1.00 bpp, BSC with $\epsilon = 0.01, 0.08, 0.1$ and codeword loss rates from 1% to 5% respectively. In all the following experiments, 5000 trials were conducted for each case, and the results were reported in PSNR converted from mean MSE.

To be overly fair to the highly scalable scheme of [45], we generated source codestreams with the default settings and 50 quality layers. The source rates are chosen such that these source codestreams are assumed to be protected by some ideal channel codes which achieve the capacity of the given binary symmetric channels. Each channel codeword has 500 bytes and the source codestreams are loaded into these codewords having the code rate equal to the BSC capacity. The resulting data are then exempted from the BSC and subjected only to complete codeword (packet) loss. The codestreams from our scheme, also designed only for BSC, are subject to both BSC and codeword loss. The number of channel codewords is the same in both systems (for the fixed-n case). Results are shown in Fig. 3.6. It is clear that our ER-UEP scheme provides significant gains in this situation. Also shown in the figures

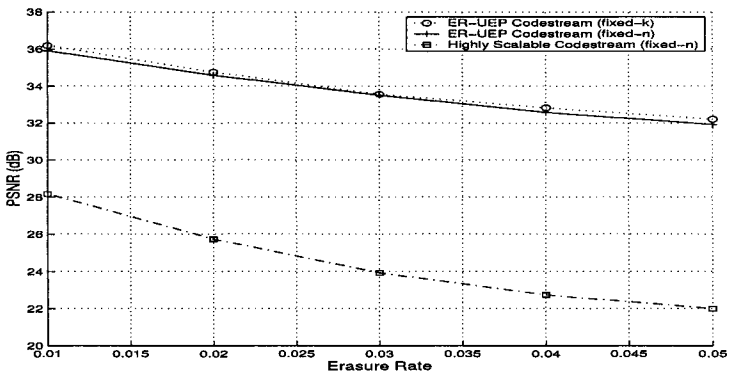
are the results for our fixed-k case. Since in this case, there are fewer channel codewords than the fixed-n case, the comparison may not be fair. Nevertheless, results are included for completeness.

Finally, we consider the mismatched channel case in which the actual channel is BSC, but the bit error rate is greater than what was assumed during the rate allocation process. Bit error rate degradation from 0.01 to 0.03 and 0.08 to 0.1 were simulated for a total bit rate of 1.00 bpp for both schemes, as shown in Table 3.7. The results show significant gains in our scheme from allowing decoding to continue to the end of codestreams, rather than halting before the first detected error. Compared with the results obtained from the rate allocation based on [45], [53] using our source and channel codes (and the original data from [53]) as listed in the table, we can also provide more graceful performance degradation in this mismatched case.

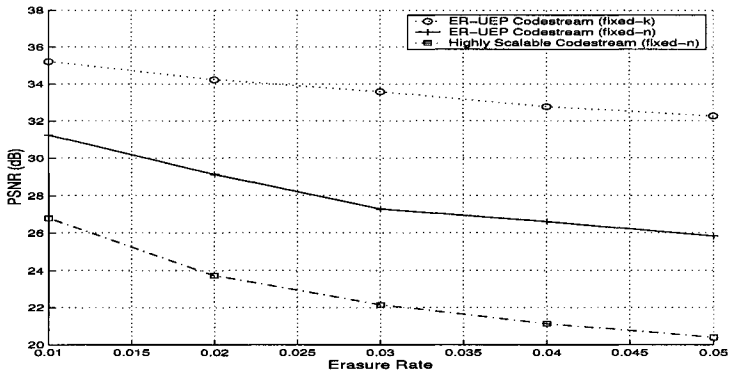
Table 3.7: PSNR performance when channel condition degrades: Lenna (512×512) at 1.00 bpp.

Designed/ Actual BER	ER-UEP		UEP based on [45], [53] (Fixed-n)	UEP from [53] (Fixed-n)
	Fixed-k Continue (Halt)	Fixed-n Continue (Halt)		
0.01/0.03	30.66 (28.11)	31.10 (27.78)	29.41	30.10
0.08/0.10	36.18 (33.83)	34.13 (30.30)	33.68	32.83

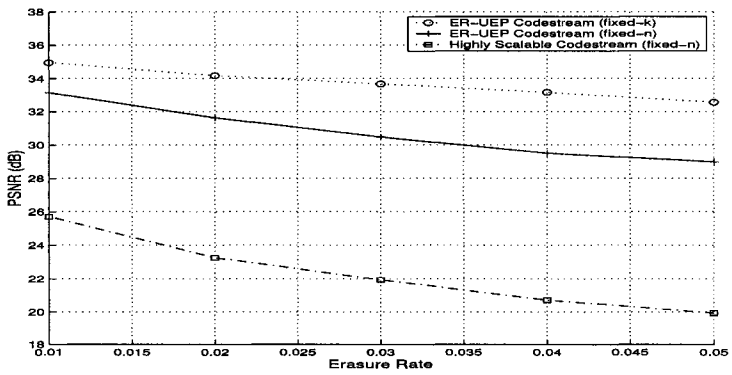
As in the matched case, the gain provided by our error resilience diminishes as the total bit rate decreases. For example, at a total bit rate of 0.25 bpp, in the first mismatched experiment, the gains decrease to as little as half of those in the corresponding 1.00 bpp case. In the second mismatched experiment, the gains between



(a) BSC 0.01 + Codeword Erasures (1% to 5%)



(b) BSC 0.08 + Codeword Erasure (1% to 5%)



(c) BSC 0.1 + Codeword Erasure (1% to 5%)

Figure 3.6: PSNR performance for mismatched channel conditions: Lenna (512×512) at 1.00 bpp.

"Continue" and "Halt" are reduced to within about 1 dB in both the "0.01/0.03" and "0.08/0.10" cases.

3.6 Conclusion

In this chapter, a joint source/channel coding scheme for single image transmission over memoryless channels tailored for JPEG2000 is proposed. It combines the forward error correction capability provided by channel codes with the error detection/localization capability provided by JPEG2000 error resilience mechanisms in an effective way. Based on source and channel code statistics, the proposed rate allocation scheme determines the optimal rate distribution between the source and channel coders. It can form quality scalable, error resilient codestreams for both fixed length source packets or fixed length channel packets with the same algorithm. It has competitive performance with existing schemes for the matched channel condition case. When mismatched channel conditions occur, it can provide more graceful quality degradation through robust source decoding.

At the same time, the proposed algorithm explores the use of some error resilience mechanisms of JPEG2000 for joint source/channel coding applications. In particular, the property of independent coding of codeblocks, the quality progressive layering structure, as well as the error resilience functionalities of PPM, ERTERM and RESTART are employed in the proposed algorithm.

CHAPTER 4

JOINT SOURCE/CHANNEL CODING FOR SINGLE IMAGE TRANSMISSION OVER CHANNELS WITH MEMORY

4.1 Introduction

The rapid growth of wireless communication has created tremendous demands for visual content transmission, including images and video. These demands come from a wide range of application areas, including 2.5G/3G mobile communication, WLAN (Wireless Local Area Network) and broadcasting/proprietary radio networks. They provide numerous applications, such as MMS (Multimedia Message Service), video telephony, wireless digital still camera, UWB terminals, wireless sensor networks, etc. However, wireless communications are characterized by high transmission error rates, low bandwidth, power limitations and so forth. These constraints pose great challenges in system design.

Unlike the channels discussed in Chapter 3, the channels experienced in wireless communications usually exhibit memory. The effect of channel memory tends to make channel errors come in bursts during transmission. The channel error characteristics

have a fundamental impact on the strategy taken when designing effective source and channel coding schemes.

Sherwood et al. proposed a scheme using product channel codes for SPIHT coded images over fading channels [54]. Their product codes consist of RCPC/CRC codes along the channel packets and RS codes across channel packets. A decoding failure from the RCPC/CRC codes results in a packet erasure which can be recovered by the following RS decoding. Without bit interleaving, this scheme can start decoding source bits from a received packet if no decoding failure is detected. Otherwise, correct decoding can still occur with some delay when RS codes are decoded after all necessary packets have arrived. A similar approach was taken in [55], but with the employment of recursive systematic convolutional (RSC) codes in both directions of the product codes. With that structure, the channel decoding information from one direction can be utilized for the decoding of the other direction in an iterative fashion. This practice leads to stronger channel codes than those in [54] at the same rate, and thus results in performance improvement. In [56], a scheme employing the same channel coding structure as [54] was proposed. The concept of multiple description was utilized within its rate allocation for progressive image coders like SPIHT. Proper amounts of protection from RS codes are applied to the symbols across the channel packets with consideration of the rate-distortion function of an input image. This scheme provides performance improvement over [54] at the expense of extra delay. In [57], an empirical model for the decoded block-error rate probability of the RCPC

channel codes was developed and incorporated into its rate allocation for progressive source-channel coding with several channel models, including Gilbert-Elliot channels (GECs). The authors in [58] took a different approach than that above. Their scheme is based on principal component partitioning-based vector quantization source coding and maximum *a posteriori* (MAP) channel decoding. The residual redundancy is exploited to improve the performance of image transmission over a GEC and no explicit channel encoder is used. In [59], a hybrid scheme was presented for channel models having both packet erasures and bit errors. The packet erasures come from the wireline portion of the channels due to packet losses. The bit errors come from the wireless portion of the channel with a flat-fading Rayleigh channel model. The scheme concatenates an error resilient source coder (PZW [33]) with an RCPC/CRC channel coder to combat noises with varying characteristics. Bit interleaving is employed to disperse burst errors. In that scheme, FEC is used to correct errors throughout a bitstream. When bursty errors occur and overwhelm the FEC, the PZW source decoder can localize the distortion impacts from the residual errors by decoding those wavelet trees in a bitstream that are not corrupted by the errors.

Recent research in error correction codes has seen a great deal of interest in low-density parity-check (LDPC) codes. Irregular LDPC codes that can achieve reliable transmission, close to the Shannon limit on AWGN channels, have been constructed [14]. Well constructed LDPC codes outperform the best known Turbo codes when the codeword length is large. Furthermore, LDPC codes possess several other distinct

advantages, such as low complexity decoding and fully parallelizable message-passing algorithm. Therefore, it is attractive to apply LDPC codes to image and video transmission.

In [47], punctured irregular repeat-accumulate (IRA) codes were applied for scalable image and video transmission over binary symmetric channels. A procedure was developed for designing a set of IRA codes obtained by puncturing from a mother IRA code. The mother code degree profiles are optimized to ensure that the entire set of codes thus obtained perform well. The resulting channel codes are applied to scalable image coders (SPIHT and JPEG2000) and video coders (3D-SPIHT and H.26L-based PFGS) with rate allocation following that in [46]. It adopts the same channel packet setting as [45] and provides consistently better performance than [45]. This improvement comes primarily from the stronger protection of IRA codes as compared to the Turbo codes employed in [45]. In [60], a procedure for source-optimized IRA code design was proposed. It utilizes the inherent unequal error protection capabilities of IRA codes due to node degree irregularities. The rate-distortion function of a source is incorporated into the density evolution of an IRA code design in order to obtain the optimal code degree profile, which minimizes the expected source distortion. A single channel codeword is encoded for an entire source bitstream of an input image coded by SPIHT. CRC bits are embedded periodically in a source bitstream to detect the first error after channel decoding, after which its source decoding stops. This scheme can achieve performance close to theoretical limit.

As discussed in Chapter 2 and Chapter 3, JPEG2000 can provide state-of-the-art compression efficiency as well as a rich set of functionalities, such as scalability and error resilience. All these advantages make it a good candidate for wireless multimedia applications. For example, one of its extensions, wireless JPEG2000 is currently under development [61].

In this chapter, a joint source/channel coding scheme for image transmission over channels with memory is proposed. It is based on desired LDPC code properties and JPEG2000 functionalities. In Section 4.2, a novel irregular LDPC code construction is proposed. It has the advantage of efficient representation of the parity check matrix of an irregular LDPC code. In Section 4.3, the underlying rationale of the proposed scheme is described. Experimental results are presented in Section 4.4 and Section 4.5 draws some concluding remarks.

4.2 Lattice-Based Irregular LDPC Code Construction

LDPC codes can be well represented by bipartite graphs in which one set of nodes, the variable nodes, corresponds to elements of the codeword and the other set of nodes, the check nodes, corresponds to the set of parity-check constraints which define the code. The number of edges connected to each node is called the degree. Regular LDPC codes are those for which all nodes of the same type have the same degree; while for irregular LDPC codes, the degree of each node is allowed to vary, according to some distribution.

An irregular LDPC code ensemble can be specified by a degree distribution pair (λ, ρ)

$$\lambda(x) = \sum_{i=2}^{d_v} \lambda_i x^{i-1} \quad \text{and} \quad \rho(x) = \sum_{i=2}^{d_c} \rho_i x^{i-1} \tag{4.1}$$

where λ_i and ρ_i are the fractions of edges belonging to degree-i variable and check nodes, d_v and d_c are maximum variable degree and check degree, respectively. The code rate r is given by

$$r = 1 - \frac{\int_0^1 \rho(x)dx}{\int_0^1 \lambda(x)dx} \tag{4.2}$$

For LDPC codes, there exists a noise threshold effect [62]: if the noise level of a channel is below a certain noise threshold, as the LDPC code block length tends to infinity, it can achieve bit error probability arbitrarily small. Conversely, if the noise level is above the noise threshold, its bit error probability is always bounded away from zero. Given a degree distribution pair, density evolution [62] is an algorithm that tracks the evolution of message distributions during each message-passing decoding iteration and it can lead to the determination of the noise threshold. It has been successfully applied for LDPC code design, such as in [14], [63].

Short cycles in bipartite graphs make the extrinsic information from different nodes correlated in fewer iterations, hence they compromise the optimality of the density evolution as well as message-passing decoding. Therefore, removing short cycles (for example, length-four cycles) in a finite-length LDPC code is often important.

Several explicit methods of constructing LDPC codes free of short cycles are known, such as finite geometries in [64] and balanced incomplete block design (BIBD) in [65]. We consider a construction based on an integer lattice [66], referred to as a Lattice code. The parity check matrix of a Lattice code is obtained as a point-line incidence matrix of a structure comprising a set of points on a $p \times q$ integer lattice and lines with slopes ranging from zero to $p-1$ such that each line has exactly p points on it. The construction described in [66] requires that p and q be coprime, and it gives a parity check matrix in the form of a $q \times p$ block matrix composed of circulant blocks of dimension $p \times p$. It can be shown that the blocks can be written as powers of circulant permutation matrices. For example, for $q = 2$ and $p = 3$, we have

$$H = \left[\begin{array}{ccc|ccc|ccc} 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ \hline 1 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 0 & 0 \end{array}\right]$$

As can be seen, the Lattice codes are (q, p) regular LDPC codes. The absence of length-four cycles is guaranteed by the construction, i.e., by the fact that no more than one line passes through any pair of points.

In this section, a new method for constructing irregular LDPC codes is proposed based on the Lattice codes and density evolution. The LDPC codes thus constructed are applied for transmitting images later in this chapter.

Given a desired codeword length, channel parameter and maximum variable and check node edge degrees d_v and d_c, our construction goal is to find an irregular LDPC code with highest code rate which has threshold "worse" than the channel parameter, with its codeword length closest to the desired one.

Suppose H_{reg} is a matrix with $p \times p$ columns constructed based on the Lattice codes, where p is a prime such that it has p^2 closest to the desired codeword length among all the primes. If H_{reg} has q blocks in its rows and p blocks in its columns, then as a binary matrix it has dimension $(qp) \times (p^2)$ and can be uniquely determined by the Lattice codes construction. As discussed before, H_{reg} is a parity-check matrix specifying a (q, p) regular code and is free of length-four cycles as guaranteed by the Lattice codes.

We denote H_p as a $q \times p$ matrix with binary elements such that each entry of H_p corresponds to a block in H_{reg}. Let H_{irr} have the same dimension as H_{reg} and define it as

$$H_{irr}(i,j) = \begin{cases} H_{reg}(i,j) & \text{if} \quad H_p(x,y) = 1 \\ 0 & \text{if} \quad H_p(x,y) = 0 \end{cases} \tag{4.3}$$

where

$$i \in [(x-1)p+1, xp],\ j \in [(y-1)p+1, yp] \quad \text{and} \quad x \in [1,q],\ y \in [1,p]$$

From the above definition, we can see that when $H_p(x, y) = 1$, the corresponding block in H_{reg} is retained in H_{irr}; when $H_p(x, y) = 0$, the corresponding block in H_{irr} is set to $\mathbf{0}$. Because removing 1s from a matrix does not decrease its girth, H_{irr} is still free of length-four cycles. However, by being able to create all-0 sub-matrices in H_{irr}, we can have different weights in the columns and rows of H_{irr}. Therefore, H_{irr} can represent the parity-check matrix of an irregular LDPC code. Clearly its structure is uniquely determined by q, p and the much smaller matrix H_p.

Suppose LDPC codes are to be designed for AWGN channels, and we follow the design procedure of density evolution. Calculating thresholds and optimizing degree distributions based on density evolution are computationally intensive tasks [62]. They are usually difficult for most channels except binary erasure channels (BECs). To simplify the analysis of the LDPC decoding algorithm for the memoryless binary-input continuous-output AWGN channel, an algorithm called Gaussian Approximation (GA) was proposed in [67]. It converts the infinite-dimensional problem of iteratively calculating message densities to a one-dimensional problem of updating means of Gaussian densities, in order to find the exact threshold. In particular, for AWGN channels with sum-product decoding, the algorithm approximates message densities as Gaussian (for regular LDPC codes) or Gaussian mixtures (for irregular LDPC codes). Therefore, the mean of a Gaussian density can be used as a faithful surrogate for the message density, hence simplifying an infinite-dimensional vector into a one-dimensional quantity.

We consider an ensemble of irregular LDPC codes with degree distributions $\lambda(x)$ and $\rho(x)$, with maximum check node and variable node degrees denoted by d_v and d_c, respectively. A node gets i.i.d. messages from its neighbors, where each message is a random mixture of different densities from neighbors. Let v and u denote these message mixtures from variable nodes and from check nodes, respectively.

According to GA assumptions, the individual output of a variable or a check node is Gaussian. Therefore, the mean $m_{v,i}^{(l)}$ of the output message of a degree-i variable node at the lth iteration is given by

$$m_{v,i}^{(l)} = m_{u_0} + (i-1)m_u^{(l-1)} \tag{4.4}$$

In this equation, u_0 is the observed log-likelihood ratio (LLR) of the output bit associated with the variable node, and m_{u_0} is the mean of u_0. $m_u^{(l-1)}$ is the mean of u at the $(l-1)$st iteration, where u is a Gaussian mixture in general. Also, the mean $m_{u,j}^{(l)}$ of the Gaussian output message $u_j^{(l)}$ of a check node with degree j at the lth iteration is given by

$$m_{u,j}^{(l)} = \phi^{-1}\left(1 - \left[1 - \sum_{i=2}^{d_c} \lambda_i \phi(m_{v,i}^{(l)})\right]^{j-1}\right) \tag{4.5}$$

where the function $\phi(x)$ for $x \in [0, \infty)$ is defined as

$$\phi(x) = \begin{cases} 1 - \frac{1}{\sqrt{4\pi x}} \int_{\mathbb{R}} \tanh \frac{u}{2} e^{-\frac{(u-x)^2}{4x}} du \text{ , if x} > 0 \\ 1 \qquad\qquad\qquad\qquad\qquad\qquad \text{, if x} = 0 \end{cases} \tag{4.6}$$

By linearly combining these means with their corresponding weights ρ_i, the mean of u at lth iteration is

$$m_u^{(l)} = \sum_{j=2}^{d_c} \rho_j \phi^{-1} \left(1 - \left[1 - \sum_{i=2}^{d_v} \lambda_i \phi \left(m_{u_0} + (i-1) m_u^{(l-1)} \right) \right]^{j-1} \right) \tag{4.7}$$

Based on (4.7), for $0 < s < \infty$ and $0 \leq t < \infty$, $f_j(s,t)$ and $f(s,t)$ are defined as

$$\begin{aligned} f_j(s,t) &= \phi^{-1} \left(1 - \left[1 - \sum_{i=2}^{d_c} \lambda_i \phi(s + (i-1)t) \right]^{j-1} \right) \\ f(s,t) &= \sum_{j=2}^{d_c} \rho_j f_j(s,t) \end{aligned} \tag{4.8}$$

With (4.8), (4.7) can be re-written as

$$t_l = f(s, t_{l-1}) \tag{4.9}$$

where $s = m_{u_0}$ and $t_l = m_u^{(l)}$, and the initial value t_0 is 0. The threshold s^* is defined as the infimum of all s in $\mathbb{R}^+$ such that $t_l(s)$ converges to ∞ as $l \to \infty$. And from Lemma 2 in [67], $t_l(s)$ will converge to ∞ iff

$$t < f(s,t) \tag{4.10}$$

for all $t \in \mathbb{R}^+$.

In general, we can use density evolution based on the GA algorithm to obtain an optimal degree distribution pair (λ, ρ) for a desired channel parameter, and then search for a random matrix of specific dimension which can represent the code with the optimal distribution, and during the search remove all short cycles in the matrix.

But if we start the search with the matrix H_{reg}, we can save the task of removing all the length-four cycles to reduce construction complexity.

One practical problem with an LDPC code generated based on density evolution and a random matrix is that we have to present the entire matrix. When the matrix is large, this may impose a serious overhead burden to store the matrix. From above, our construction circumvents this problem, as it is possible to use q, p and a small matrix H_p $(q \times p)$ to describe the large matrix H_{irr} of dimension $(qp) \times (p^2)$, thus giving a factor of p^2 in savings. However, because each entry in H_p corresponds to a base block in H_{irr}, in order to specify H_{irr} by H_p, we can only assign one degree to all columns (or rows) within the blocks that correspond to a column (or a row) in H_p. Therefore a new restriction is imposed for an LDPC code that is representable by H_p.

Constraint 1 *The fraction of variable nodes (or check nodes) of any degree must be an integer multiple of* $1/p$ *(or* $1/q$*), where there are* p^2 *variable nodes and* qp *check nodes in the graph, with* p*,* q *coprime.*

A straightforward way to design irregular LDPC codes using this construction is to first obtain an optimal degree distribution for a specific channel parameter from density evolution, then approximate this distribution to the nearest multiple of $1/p$ (or $1/q$) for each parity node degree (or check node degree). The problem with this method is that due to the coarse granularity of $1/p$ or $1/q$, the resulting distribution

is usually different from the optimal (λ, ρ) distribution (especially for small p or q values) and therefore may have a much higher threshold.

To circumvent this problem, we propose to combine the two design steps into one. During the optimization procedure, to find the best degree distribution with density evolution, we impose the new constraint that every possible distribution candidate must have each of its variable (or check node) fractions equal to a multiple of $1/p$ (or $1/q$). The matrix thus generated represents an optimal irregular LDPC code subject to this constraint. Compared to the unconstrained case, it is suboptimal. However, our benefit is that, by trading off some optimality (in the form of decreased code rate), we are able to represent a large irregular matrix of size $(qp) \times (p^2)$ with two natural numbers p, q and a small matrix of size $q \times p$ in GF(2). This advantage is more pronounced as p or q becomes large, while at the same time, the sub-optimality decreases because of the finer approximation granularity of $1/p$ and $1/q$.

Differential Evolution (DE) [68] is employed as our optimization algorithm to find the best LDPC code with density evolution. It is a robust optimizer for nonlinear constraint satisfaction with continuous space parameters, and it has been successfully applied to the design of good LDPC codes [69].

The major steps for the proposed irregular LDPC code design are shown in Table 4.1.

Table 4.1: Lattice code based irregular LDPC code construction procedure.

Step 1 Determine the block dimension $(p \times p)$ from the codeword length requirement. Set the iteration number IT to 1 and the optimal code rate R to 0.

Step 2 According to DE specifications, randomly generate a population of edge degree distribution pairs (λ, ρ) for both variable and check nodes with some pre-specified size and input degrees, such that for each (λ, ρ), $\sum_i \lambda_i = \sum_i \rho_i = 1$, and their corresponding node degree distribution satisfies Constraint 1.

Step 3 Check each (λ, ρ) member of the population. If it satisfies (4.10), calculate its rate r according to (4.2). If $r > R$, set $R = r$ and record this distribution. After all the members are checked, increment IT by 1.

Step 4 If R is less than a pre-specified rate, or the iteration number IT is less than a pre-specified maximum number, go to Step 2; otherwise continue to Step 5.

Step 5 Set $q = p \cdot R$ and randomly generate a matrix H_p of dimension $q \times p$ according to the recorded distribution.

Step 6 Create a Lattice code matrix H_{reg} of dimension $(qp) \times (p^2)$. Construct H_{irr} according to (4.3) with H_{reg} and H_p. H_{irr} is the parity-check matrix for the desired irregular LDPC code.

4.3 Joint Source/Channel Coding Algorithm

In this section, we consider the transmission of a single image over a channel with memory within a specified total bit rate.

As discussed in Section 4.1, most schemes designed for channels with memory employ some product codes with or without bit interleavers, in order to combat bursty errors introduced by the channels. Generally, for schemes employing interleaving, better performance can be achieved but at the cost of extra delay. This is because with interleaving, a source decoder has to wait until all the packets have arrived to begin decoding. This may not be tolerable for applications with stringent scalability requirement.

By contrast, LDPC codes are proposed in this section to replace product codes for channels with memory. It has been observed that LDPC codes are effective for combating burst errors [63] [70] (such as those introduced by GECs.) This is due to the so-called "built-in interleaver" property in the parity-check matrix H of an LDPC code [63]. Heuristically, for a randomly constructed LDPC code, permuting its columns results in a different LDPC code. However, both codes have the same degree distribution and thus belong to the same code ensemble. By taking this permutation into the bursty channel models, it can be shown that bursts are effectively scattered, which gives the same effect as an interleaver [70]. Because of this property, LDPC codes can be very effective for channels with memory.

Because well designed LDPC codes are capable of approaching channel capacity given sufficiently long codewords [14], it can be expected that with judicious design and long enough codewords, LDPC codes can surpass product codes in performance. Furthermore, since no interleaving is necessary by using LDPC codes, source decoding scalability can be maintained [1] .

A straightforward way to perform the joint rate allocation with LDPC codes is to carefully design an LDPC code such that it has very low probability of decoding failure for a specific channel. An entire bitstream from an image coder is then protected equally by the code according to a total bit rate constraint. This is the strategy taken by [38], which provides equal error protection (EEP) for a source bitstream.

However, for the bitstream from a scalable image coder, different segments of the bitstream usually have different importance in reconstructing an original image. Therefore, unequal error protection (UEP) can be applied to improve performance, as shown in Chapter 3. There is a practical problem of designing effective UEP schemes based on LDPC codes. In order to achieve superior performance with LDPC codes, their codeword sizes usually are chosen to be relatively large. This may result in a few or even one encoded LDPC codeword for an image to be transmitted. When channel codeword sizes are small, a jointly coded bitstream can consist of many channel codewords, such as the RCPC and RCPT cases in Chapter 3. In these cases, a channel decoding failure in those codewords will very likely leave most of the bits

[1]This is true for systematic LDPC codes or if the LDPC codeword is short compared to the "effective" codeword length of a product code.

in the codeword corrupted. Hence, a source decoder can stop decoding and discard an entire corrupted codeword without losing many correct source bits within the codeword. However, this practice may be costly for an LDPC codeword with large size, especially for bursty channels. An example is shown in Fig. 4.1. In the figure, an LDPC codeword is divided into 5 segments. Suppose before channel decoding (upper part), there are bit errors in segments 1, 2, 4 and 5. After LDPC decoding (lower part), the bit errors in segments 2 and 5 are corrected. However, the bit error in segment 1 and most bit errors in segment 4 still remain. Despite the residual errors, it is desirable that a scheme can utilize the correct bits from segment 2, 3 and 5, and even those from segment 1 and 4. This requires effective error detection and localization with a scheme using LDPC codes. If the correct bits can be utilized by a source decoder, it is beneficial for reconstructing the original source, especially for long LDPC codewords. For this reason, the prevalent practice to stop source decoding before a channel codeword failure, taken by [40], [44], [45], [71] etc., may not be suitable in this situation.

In [60], a UEP scheme based on LDPC codes and the SPIHT image coder was proposed for scalable image transmission over binary symmetric channels. In that scheme, one LDPC codeword of large size (for example, 66060 bits and 132380 bits) is employed to encode an entire image source bitstream. The inherent unequal error protection from an irregular LDPC code is utilized to provide UEP gain. Inside the long LDPC codeword, CRC codes are embedded periodically to detect the first

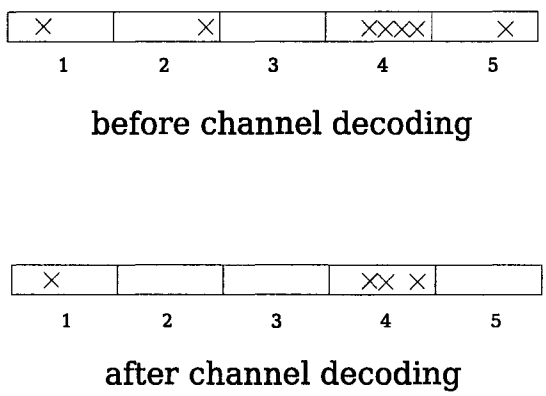


Figure 4.1: Channel bit errors inside an LDPC codeword before and after LDPC decoding.

residual error after channel decoding and its source decoding halts after the error. However, since one channel codeword is coded for an image, the receiver has to receive all the bits in order to start channel and source decoding. This introduces extra delay. Furthermore, if that scheme is applied to bursty channels such as GECs, although after channel decoding, it is possible to utilize the correct bits before the first error

based on the information from the CRC codes, the correct bits after the errors (for example, segments 2, 3 and 5 in the lower part of Fig. 4.1) are not used.

To overcome this limitation, we propose a joint source/channel coding scheme for image transmission over bursty channels based on LDPC codes and JPEG2000. In our scheme, in order to reduce source decoding delay and improve scalability, a certain number of LDPC codewords are formed for an image bitstream, as opposed to only one in [60]. Therefore, a source decoder can utilize those source bits within each received codeword to improve the quality of the reconstructed image. The number of codewords is chosen such that most of the UEP gains from a scalable source can be achieved. And with this constraint, the lengths of the LDPC codewords are chosen as long as possible to have good channel coding performance. To obtain a set of LDPC codes for different error protection levels in exchange of code rates, different thresholds are supplied in (4.10) during code construction based on the procedure proposed in Table 4.1.

The same set of error resilience mechanisms are employed in this chapter as in Chapter 3 for memoryless channels. To be more specific, error resilience modes *RESTART* and *ERTERM* are invoked during the creation of a source bitstream. As discussed in Chapter 3.2, when they are used in concert, a residual error in a source bitstream can be detected by a JPEG2000 decoder with high probability at the end of the coding pass in which it occurred. Then the decoder only discards the corrupt coding pass and all subsequent coding passes from the same codeblock to

prevent otherwise erroneous decoding artifacts caused by loss of synchronization. In addition, *PPM* marker segments are employed to relocate packet headers into the main header where they are protected strongly by LDPC codes.

With the bitstream structure adopted, the same rate allocation algorithm proposed in Chapter 3.3 can be applied directly for the bursty channels and the LDPC codes in this chapter. Thus in this chapter, our scheme generates a scalable and error resilient source bitstream which is channel coded by a few LDPC codewords. With the proposed scheme, the error resilience mechanism from JPEG2000 and the forward error correction capability from LDPC codes are combined in an effective way. The LDPC codes reduce the bit error rates introduced by the channel. When bursty errors occur which cause an LDPC decoding failure, the error resilience capabilities provided by JPEG2000 can detect the residual errors and localize their impact.

In our scheme, an LDPC codeword generally corresponds to a quality layer and it contains all the packets for that layer. The packets serve as resynchronization points for the source decoding. With the employment of the error resilient mode *ERTERM*, residual errors can be detected at the end of each coding pass by the source decoder with high probability. Thus our scheme does not need to employ error detection codes as in [60]. However we can provide finer error detection capability with reduced system complexity. Furthermore, with the proposed structure, both correct bits before and after the erroneous part can be decodable by the source decoder. If there

are more channel codewords after the one with decoding failure, the source bits inside those codewords can still be selectively utilized by the source decoder.

4.4 Experimental Results

In the following experiments, Lenna and Goldhill are chosen as our test images over Gilbert-Elliot channels (GECs) with different channel parameters. Each image is 512×512 with 8-bit depth, and they are transmitted at the total bit rates of 1.00, 0.50 and 0.25 bpp, respectively.

For the GECs in the following experiments, from [39] (and other results from the literature), at most three protection levels are enough to obtain most UEP gain. To obtain the UEP gain at each total bit rate of interest, while keeping the codeword size as large as possible to have good code performance, an LDPC codeword of size about 21845 should be chosen, which corresponding to the 0.25 bpp case for the test images. Together with the size constraint imposed from our LDPC code construction, a base block size of 149 is selected, which gives a set of LDPC codes with a fixed codeword length of 22201 bits.

As discussed in Section 4.3, because of the "built-in interleaving" property, the LDPC design procedure for a bursty channel can be carried out based on its "interleaved" channel model. For the GECs used in our experiments, BSC can serve as the surrogate. Furthermore, as pointed out in [62], a degree distribution pair (λ, ρ) optimized for BIAWGN is usually a very good degree distribution pair for a large

classes of channels, including BSC. Because the density evolution based on GA for BIAWGN is less complex than it is for BSC (as mentioned in Section 4.2), so the design procedure laid out in Table 4.1 for BIAWGN can be applied to the LDPC codes for BSC, and further for GEC.

In Section 4.2, the design criterion is to find an irregular LDPC code for a given threshold, such that it has the highest code rate and its degree distribution pair (λ, ρ) satisfies (4.10) and Constraint 1. In order to obtain a set of LDPC codes with different protection levels and code rates, different thresholds σ are substituted into (4.10) during construction. Table 4.2 shows the degree distribution of the LDPC codes C_1 through C_{10} obtained by the proposed construction algorithm, with each codeword length equal to 22201 bits. In the table, σ^2 denotes the noise variance of a specific BIAWGN channel and $p = Q(1/\sigma)$, where $Q(x)$ is the Q-function. The values of the thresholds are chosen according to the channel parameters of the GECs in the following experiments. The LDPC codes thus constructed have the advantage of efficient representation of their parity matrices. In the examples of C_1 through C_{10}, a saving in size of 22201 times is achieved to store the H matrices.

The sum-product algorithm is employed for LDPC decoding. Through extensive simulation over GECs, it has been observed that in the case of an LDPC decoding failure, the residual error rate is effectively reduced and no extra bit errors are introduced in the decoded codeword. This is a desirable property because a source decoder can maximally utilize the remaining correct bits in the codeword, as discussed before.

Table 4.2: LDPC code construction results.

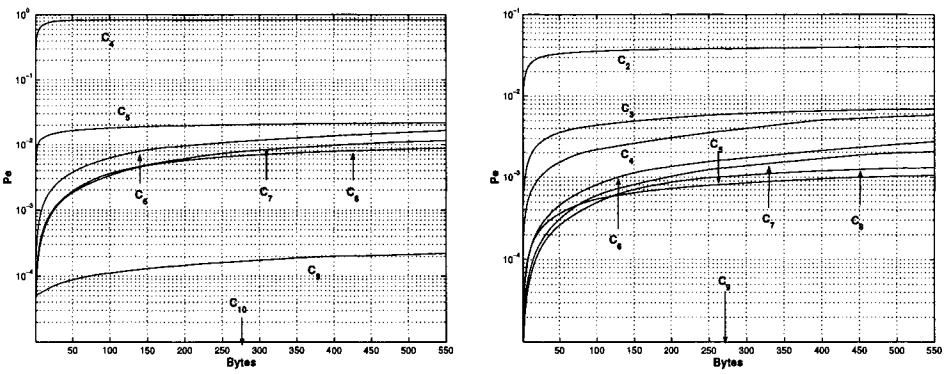
	C_1	C_2	C_3	C_4	C_5	C_6	C_7	C_8	C_9	C_{10}
λ_2	0.2790	0.2359	0.2019	0.2226	0.1975	0.1965	0.2212	0.2175	0.2037	0.1738
λ_3	0.2181	0.1744	0.2452	0.2099	0.2586	0.1804	0.1545	0.1116	0.1214	0.1284
λ_4	0.0236	0.0598	0.0256	0.0382	0.0251	0.0059	0.0364	0.1087	0.0418	0.1245
λ_5	0.3045	0.1412	0.0321	0.0238	0.0157	0.1026	0.0758	0.0072		
λ_6										0.0233
λ_7	0.0275	0.0116	0.1234	0.0779	0.0549	0.0411	0.0636	0.0100	0.0457	
λ_8									0.0209	
λ_9									0.0587	0.0117
λ_{14}		0.3023	0.1795	0.3561	0.3072	0.1437	0.0848	0.1803		
λ_{15}	0.1473	0.0748	0.1923	0.0715	0.1411	0.3299	0.3636	0.3648		
λ_{19}									0.2728	0.1232
λ_{20}									0.2350	0.4150
ρ_8		0.1217	0.1789	0.3427	0.5593	0.4280	0.9527	0.7464	0.4792	0.4334
ρ_9	0.0206	0.0748	0.1923	0.6573	0.4407	0.5720	0.0473	0.2536	0.0215	0.4621
ρ_{10}	0.9794								0.4993	0.1045
σ	0.7377	0.8874	0.9084	0.9350	0.9551	0.9678	0.9939	1.0126	1.0391	1.0574
p	0.06	0.12	0.13	0.14	0.15	0.16	0.17	0.18	0.19	0.20
rate	0.6577	0.5913	0.5726	0.5101	0.4899	0.4631	0.4497	0.4296	0.4161	0.3960

The GECs tested in the following experiments are listed in Table 4.3 with their parameters. Specifically, channel 3 has fading parameters corresponding to GSM with a carrier frequency of 900 MHz, transmission rate of 33.8kb/s/user, and a mobile at 35 mph, as indicated in [57].

Table 4.3: Channel parameters of GECs.

	ϵ_G	ϵ_B	P_{GB}	P_{BG}	Avg. BER
channel 1	0.001	0.1	1/3600	1/400	0.011
channel 2	0.01	0.111	9/100	1/100	0.1
channel 3	0.001	0.12	0.005	0.0031	0.0742

EEP rate allocation is considered first. Based on channel simulation, codes C_1, C_9 and C_{10} yield extremely small probabilities of decoding failure for channels 1, 2 and 3, respectively. According to the code rate of the selected channel code, a source bitstream is generated corresponding to each total bit rate and test channel. Then the source bitstream is channel coded by a number of same LDPC codewords.



(a) channel 2 (b) channel 5

Figure 4.2: Coding pass error rate ($P(r_i, \frac{l_i}{r_i})$) vs. coding pass length.

The rate allocation of UEP is considered next. Because of the relatively low bit error rate of channel 1, the LDPC codes designed for the channel do not have significant differences in their code rates. This results in marginal UEP gains with our proposed scheme. Therefore we consider applying our scheme to channels 2 and 3. Based on their performance, the code set $\{C_i\colon\ 4 \leq i \leq 10\}$ is provided for the rate allocation for channel 2 and the code set $\{C_i\colon\ 2 \leq i \leq 9\}$ is provided for channel 3. Their error statistics were simulated and tabulated. Fig. 4.2 shows the coding pass error rates $P(r_i, \frac{l_i}{r_i})$ for the LDPC codes (similar to Fig. 3.3 in Chapter 3), where $P(r_i, \frac{l_i}{r_i})$ is the probability that there are one or more uncorrected errors after channel decoding for coding pass i with length l_i and channel code rate r_i. After the rate allocation optimization, codes $\{C_{10}, C_5\}$ are chosen for channel 2, and codes $\{C_9, C_3\}$ are chosen for channel 3 for both images.

JPEG2000 based Kakadu software is used as our source coder. All test images are generated with a 5-level (9,7) wavelet transform, a codeblock size of 64 × 64. 200 decoding iterations were performed for $\{C_{10}, C_5\}$ over channel 2, and 100 iterations were performed for $\{C_9, C_3\}$ over channel 3. For EEP cases, one layer was generated for each image (to achieve best performance). For UEP cases, *ERTERM* + *RESTART* modes were turned on and *PPM* marker segments were employed. Layering functionality was invoked in the UEP cases to generate scalable bitstreams, with the number of layers equal to the number of different channel codes selected for the bitstream. For each case, at least 2000 trials were conducted. Simulation results for the proposed scheme are listed in Tables 4.4 to 4.6. The reported PSNR values were calculated by converting from the average decoded MSE values. Our simulation results are compared with those from [54] and [55] with the same channels. The results from [54] and [55] are based on EEP rate allocation and different product codes (P.C.).

Table 4.4: PSNR (dB) performance for channel 1.

	Lenna			Goldhill		
Bit rate (bpp)	P.C. [54] EEP	P.C. [55] EEP	LDPC EEP	P.C. [54] EEP	P.C. [55] EEP	LDPC EEP
1.00	36.56	37.50	38.45	32.66	33.49	34.43
0.50	33.37	34.34	35.40	30.15	30.72	31.57
0.25	30.19	31.10	32.21	27.89	28.58	29.22

Because the schemes from [54] and [55] are based on the EEP rate allocation, their performance is determined by the channel code rates that guarantees extremely low

Table 4.5: PSNR (dB) performance for channel 2.

Bit rate (bpp)	Lenna			Goldhill		
	P.C. [55] EEP	LDPC EEP	LDPC UEP	P.C. [55] EEP	LDPC EEP	LDPC UEP
1.00	35.25	36.09	36.47	31.46	32.48	32.62
0.50	32.35	32.83	33.26	29.21	30.03	30.13
0.25	29.06	29.82	30.24	27.27	27.89	28.03

Table 4.6: PSNR (dB) performance for channel 3.

Bit rate	Lenna			Goldhill		
	P.C. [55] EEP	LDPC EEP	LDPC UEP	P.C. [55] EEP	LDPC EEP	LDPC UEP
1.00	35.25	36.40	37.04	31.46	32.48	33.06
0.50	32.26	33.22	33.82	29.21	30.03	30.43
0.25	29.06	29.96	30.63	27.27	27.89	28.30

channel residual error probability. With reported channel rates, their performance can be reproduced with JPEG2000 source coder and different total bit rates for comparison. As shown in Table 4.4, comparing the product codes based on RSC+RSC in [55] with the product codes based on RCPC+RS in [54], RSC+RSC scheme provides stronger protection and achieves error free performance at a higher code rate. This channel code rate saving results in an increase in source coding rate, and therefore [55] provides a consistent performance improvement over [54] for about 0.5 to 1.0 dB. With our LDPC codes, channel coding protection is further strengthened. Comparing our LDPC/EEP to P.C./EEP [55] for the three channels, an additional 0.6 to 1.2 dB improvement can be achieved for both images.

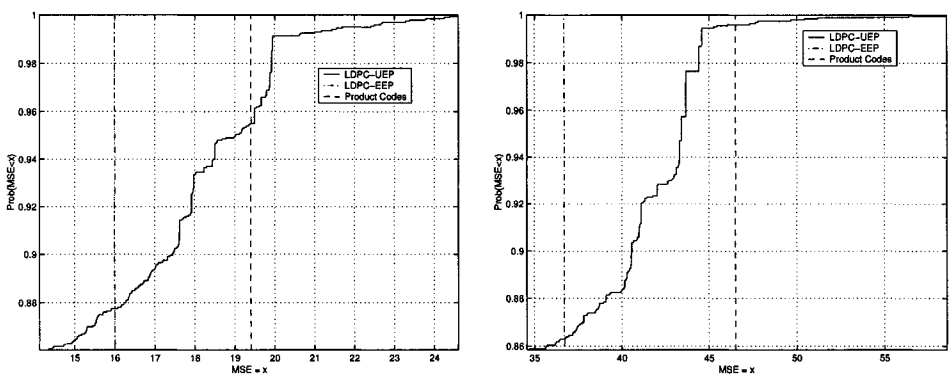
Further improvement can be achieved with the proposed UEP scheme. From Tables 4.5 and 4.6, comparing the results based on UEP and EEP schemes both with

LDPC, about 0.4 dB and 0.6 dB improvement are obtained for the Lenna image for channels 2 and 3, respectively. And for the Goldhill image, the gains are about 0.2 dB and 0.4 dB, respectively.

Furthermore, we evaluate our scheme based on the MSE cumulative distribution function (CDF) of received images, as was suggested by [59]. The function is defined by $F(x) = \text{Prob}(\text{Decoded Distortion} < x)$. This evaluation provides further information about the overall performance of a scheme over noisy channels. Because our goal is to achieve high probability of received images with low MSE, a sharp rise at a low MSE in the CDF curve is desirable. The results for Lenna and Goldhill at the total bit rate 1.00 bpp over channels 2 and 3 are shown in Fig. 4.3. The CDFs based on EEP schemes from [55] together with the proposed LDPC-based EEP and UEP schemes are plotted in the figures. From these figures, it is clear that our UEP scheme performs better than our EEP scheme most of the time, and in turn, our EEP scheme performs better than the scheme from [55] most of the time. For example, for channel 2 and the Lenna image, our UEP scheme is 87% of the time better than our EEP scheme and 96% of the time better than the scheme from [55]. Similar conclusions hold for the other two lower rates.

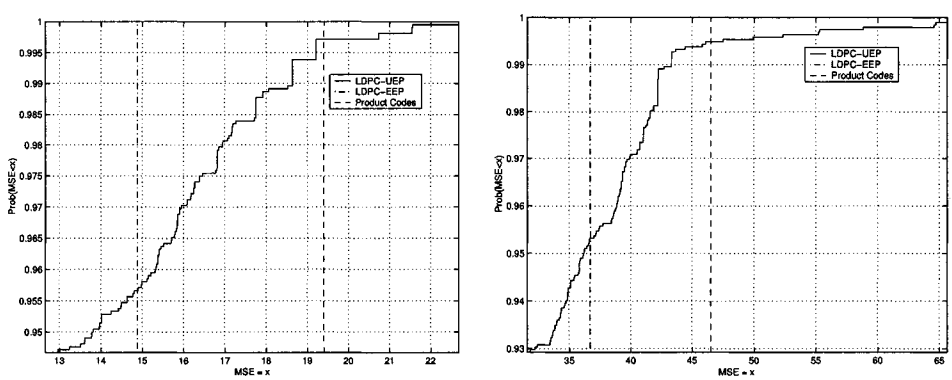
4.5 Conclusion

In this chapter, a joint source/channel coding scheme is proposed for image transmission over channels with memory based upon LDPC codes and JPEG2000. The



(a) Lenna with channel 2

(b) Goldhill with channel 2



(c) Lenna with channel 3

(d) Goldhill with channel 3

Figure 4.3: Cumulative distribution of decoded MSE for Lenna and Goldhill at total bit rate of 1.00 bpp.

LDPC codes are proposed to replace the traditional product codes for GECs. Because of the effectiveness of combating bursty errors and the capacity-approaching performance, LDPC codes provide stronger protection than the product codes under comparison. Further performance improvement is achieved through UEP gains by combining LDPC codes with the scalability and the error resilience mechanisms provided by JPEG2000. Finally, a novel LDPC code construction method is proposed in this chapter. It has the advantage of efficient representation of large parity check matrices for LDPC codes and is suitable for image transmission applications.

CHAPTER 5

JOINT SOURCE/CHANNEL CODING FOR MULTIPLE SOURCES TRANSMISSION

5.1 Introduction

Today's multimedia applications often require images and videos to be transmitted over noisy channels. With the rapid advances in source coding and channel coding techniques, joint source/channel coding has attracted a lot of research efforts recently. Many schemes have been proposed for the efficient transmission of different multimedia sources. The introduction sections in Chapter 3 and Chapter 4 have provided an overview of these schemes, with an emphasis on image transmission over channels without and with memory, respectively.

Besides image content, the demand for video transmission is also growing fast, which has attracted much research effort lately. In [72], an algorithm was presented for a video encoder to estimate the average channel distortion at the pixel level at a video decoder with error concealment. Based on this decoder simulation, the encoder is able to optimize bit allocation and perform rate control at the macroblock (MB) level for packet erasure channels. The authors in [73] used a video rate-distortion model and extended it for MB intra refreshing for the purpose of error resilience.

With this model, statistical analysis of channel distortion was performed for adaptive intra refreshing over channels with different packet loss rates. Dependence between video coding units was taken into consideration for joint source/channel coding by [74]. Models for both channel coding and video coding performance subject to channel errors were developed for AWGN channels. The scheme is able to provide unequal error protection for different frames in a video sequence.

With the current state-of-the-art data compression techniques and channel spectral utilization techniques, communication channels are now capable of delivering several compressed image or video bitstreams concurrently. This makes it possible for some applications to transmit multiple images or video sequences together sharing a common channel. Such applications may include multimedia client/server systems, wireless sensor networks and so forth.

However, all the joint source/channel coding schemes introduced so far consider the coding for only one image or video sequence. One way to utilize these existing schemes in the multiple sources transmission case is to code each image or video source separately, while keeping the aggregate bit rate below the channel capacity. However, as pointed out in [75], there are several shortcomings with separate coding over a constant bit rate (CBR) channel. For example, such shortcomings include inefficient use of channel capacity and large variations in picture quality among video sequences.

The problem of multiplexing multiple variable bit-rate (VBR) videos has been studied previously. For example, the authors of [75] proposed a joint coding scheme based on MPEG-2 to dynamically distribute the total bit rate among several video sequences and achieve a more uniform picture quality. A similar problem was tackled in [76] using the fine scalability of Motion JPEG2000. In [77], an algorithm was presented for controlling multiple VBR video encodings under buffer and channel bandwidth constraints in order to improve the overall quality. However, these schemes do not take channel coding into account.

In this chapter, a joint source/channel coding algorithm is proposed to code multiple image or video sources jointly. The proposed algorithm exploits the rate-distortion diversity of different sources in order to dynamically allocate limited bandwidth of a CBR channel among all the sources. At the receiver, in addition to unequal error protection gains, both further improved and more uniform visual quality can be achieved for the reconstructed sources.

Scalable source coders are of interest in the proposed algorithm. In particular, JPEG2000 (JP2) is employed as the image coder, where each image is regarded as an independent coding unit. 3D-JPEG2000 (3D-JP2) is employed to code video sequences at relatively low resolutions and low bit rates. A fixed number of frames from one video sequence is coded as a group of frames ("GOF") and each GOF is an independent coding unit. Finally, Motion-JPEG2000 (MJP2) is employed for HDTV sequences, which have high resolutions and high bit rates. Since each frame is coded

separately in MJP2, a frame is an independent coding unit in this case. The proposed algorithm jointly codes multiple source units of one type.

This chapter is organized as follows: in Section 5.2, an overview of the proposed joint coding algorithm is presented. The potential gains provided by joint coding of multiple sources are demonstrated in Section 5.3 and a specific algorithm that can achieve those gains is developed in Section 5.4. In Section 5.5, experimental results are presented for different types of sources and Section 5.6 draws some conclusions.

5.2 Joint Coding Algorithm

Consider that multiple sources are to be transmitted over a noisy channel within a given total bit rate. At the transmitter, each source is first progressively encoded by a source coder. For each source, its resulting bitstream is partitioned into several segments and each segment is then channel encoded to form a fixed-length channel packet to be sent over the channel. Each packet is assigned a specific channel code rate and the code rate in turn determines the number of source and parity bytes in the packet. At the receiver end, for each source, channel decoding is first performed to recover source bytes from its received channel packets. Channel decoding stops for a given source whenever a channel decoding failure occurs or all the channel packets for the source have been correctly decoded. The recovered source bytes are then decoded by a source decoder to reconstruct the source. This practice may result in decoding only a prefix of an original source bitstream. However, it prevents the

otherwise possibly catastrophic reconstruction effects due to loss of synchronization during source decoding. With the proposed joint coding algorithm, a possible system structure diagram is shown in Fig. 5.1.

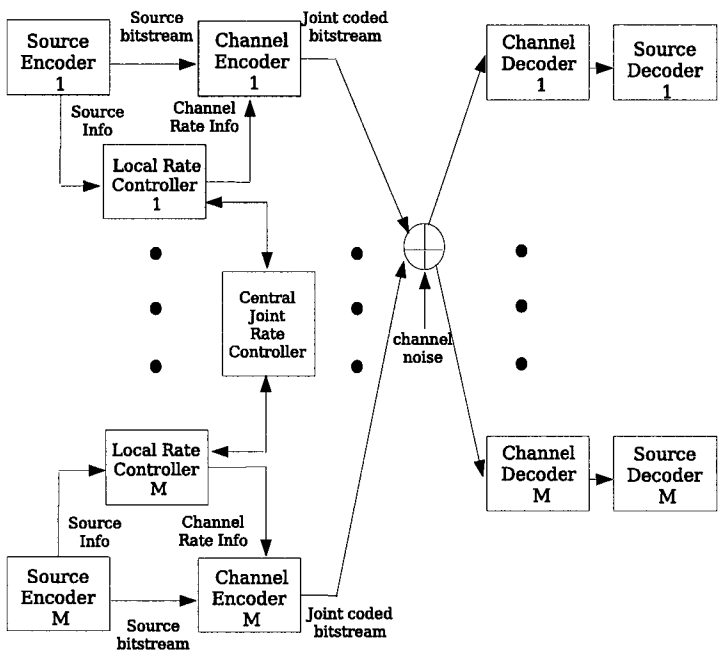


Figure 5.1: A JSCC system structure for multiple sources.

The proposed joint coding scheme dynamically distributes a total bit rate among multiple sources in order to minimize the overall expected distortion for the reconstructed sources at the receiver. In the following sections, we first prove that by

jointly coding multiple sources in such a way, it is possible to achieve both reduced overall expected distortion and a more uniform quality across the sources for the reconstructed sources at the receiver. Subsequently, a specific joint source/channel coding scheme is proposed.

5.3 Joint Coding Gains

To begin, we define $\bar{R}_i$ as the expected effective source decoding rate for source i at the receiver and it can be written as

$$\bar{R}_i = c_i^1 R_i^s = c_i^1 c_i^2 R_i = c_i R_i \tag{5.1}$$

In this expression, R_i is the total rate assigned to source i at the transmitter. This rate accounts for both source and parity data. Defining the channel coding rate as c_i^2, the portion allocated to just source data is $R_i^s = c_i^2 R_i$. Furthermore, we define c_i^1 as the expected fraction of R_i^s that is decodable by a source decoder at the receiver considering the effect of residual errors after channel decoding. Finally $c_i = c_i^1 c_i^2$ represents the expected fraction of the assigned rate R_i that can be effectively used by a source decoder to reconstruct the source at the receiver.

With (5.1), the expected distortion (MSE) for a reconstructed source i at the receiver can be modeled as [18]

$$ED_i(R_i) \approx \epsilon^2 \sigma_i^2 2^{-2\bar{R}_i} = \epsilon^2 \sigma_i^2 2^{-2c_i R_i} \tag{5.2}$$

with the condition that $\bar{R}_i = c_i R_i$ is large enough. In (5.2), σ_i^2 is the variance of source i and ϵ^2 is a constant related to the performance of a practical compression algorithm.

Let M be the number of sources to be coded jointly, and let R be the average bit rate over all sources. The objective of a joint coding scheme is

$$\min \sum_{i=1}^{M} ED_i(R_i) \qquad \text{s.t.} \qquad \sum_{i=1}^{M} R_i \leq MR \tag{5.3}$$

To find the optimal rate for each source, we may use the Lagrangian method and set

$$\frac{\partial}{\partial R_i}\{\sum_{i=1}^{M} ED_i(R_i) + \lambda \sum_{i=1}^{M} R_i\} = 0 \tag{5.4}$$

This gives

$$\begin{aligned} \lambda &= \epsilon^2 \sigma_i^2 (c_i + c_i' R_i) 2^{-2c_i R_i} \\ &\approx \epsilon^2 c_i \sigma_i^2 2^{-2c_i R_i} \end{aligned} \tag{5.5}$$

with the assumption that $c_i \gg |c_i'| R_i$, where c_i' is the derivative of c_i with respect to R_i. This assumption simplifies the derivation and clearly it is true when the coefficient c_i is insensitive to changes of R_i.

From (5.5), bit rate R_i for source i is

$$R_i = \frac{1}{2c_i} \log_2 \frac{\epsilon^2 (c_i \sigma_i^2)}{\lambda} \tag{5.6}$$

By imposing the average bit rate constraint, λ can be re-written as

$$\lambda = \epsilon^2 \left[\prod_{i=1}^{M} (c_i \sigma_i^2)^{\frac{1}{c_i}} \right]^c 2^{-2cMR} \tag{5.7}$$

where c is defined as $(\sum_{i=1}^{M} \frac{1}{c_i})^{-1}$.

Together with (5.2) and (5.6), the optimal rate assigned to source i is given by

$$R_i^* = \frac{cM}{c_i} R + \frac{1}{2c_i} \log_2 \frac{c_i \sigma_i^2}{\left[\prod_{i=1}^{M} (c_i \sigma_i^2)^{\frac{1}{c_i}}\right]^c} \tag{5.8}$$

and the optimal distortion corresponding to rate R_i^* is

$$ED_i(R_i^*) = \frac{\epsilon^2}{c_i} \left[\prod_{i=1}^{M} (c_i \sigma_i^2)^{\frac{1}{c_i}}\right]^c 2^{-2cMR} \tag{5.9}$$

According to (5.2), when the same rate R is assigned to each source (separate coding), the ratio between the expected distortions of two arbitrary sources i and j is

$$\frac{ED_i(R)}{ED_j(R)} = \frac{\sigma_i^2}{\sigma_j^2} 2^{-(c_j - c_i)R} \tag{5.10}$$

and from (5.9), the corresponding ratio with joint coding is

$$\frac{ED_i(R_i^*)}{ED_j(R_j^*)} = \frac{c_j}{c_i} \tag{5.11}$$

Suppose that c_i remains constant independent of a specific source i and is insensitive to changes of R_i. With these two assumptions, we see from (5.10) and (5.11) that joint coding provides constant quality regardless of the variances of the original sources. We call the gain in terms of quality variance reduction among reconstructed sources at the receiver "variance multiplexing" gain hereafter and define it as

$$\text{Variance Multiplexing Gain} = 1 - \frac{\text{Var}(\text{MSE}_{\text{joint}})}{\text{Var}(\text{MSE}_{\text{separate}})} \tag{5.12}$$

From previous results in the literature (also verified below), for effective JSCC schemes, $\bar{R}_i$ varies in a small range close to R_i^s for different sources and total rates. This implies that the coefficient c_i^1 also varies in a small range close to 1. Based on this observation, it suffices to analyze c_i^2 to determine if our two assumptions are satisfied.

As discussed above, the optimization criterion is expected distortion. Solutions to such problems are called "distortion-based optimal solutions" in [46]. Meanwhile, as proposed in [40], for quality scalable source coders, maximizing the expected number of correctly received source bits can be an alternative. Solutions of such problems are called "rate-based solutions" in [46]. One unique feature of the rate-based criterion is that its solution is independent of specific source statistics or source coding performance. The rate-based criterion is proven to yield suboptimal solutions to the distortion-based problem [46]. However, in the context of fixed length channel packets and a source coder with nonincreasing and convex operational distortion-rate function, an error bound between the MSE achieved under the two criteria was derived, and the gap was shown to be small [46]. Therefore, a distortion-based scheme has solutions close to those based on the rate-optimal criterion, which is independent of a specific source. This implies that c_i^2 is approximately independent of a specific source, which satisfies our first assumption (it is also verified by our experiments below.)

For a rate-based optimal solution, it has been shown in [71] that an optimal rate allocation typically has a long run of channel packets assigned the same code rate. This is due to the fact that the number of available code rates is usually much smaller than the number of transmitted packets, and also to the property that the code rates are nonincreasing for an optimal solution [71]. Even at the points where the code rate changes, these changes are usually small. This is due to the fact that the suitable rates inside a channel code family for a specific channel are not far apart. In Fig. 5.2, we plot c_i^2 as functions of R_i for the rate-based optimal solution based on the experimental results in [71], with the total bit rate ranging up to 4.00 bpp. These results were obtained for the rate compatible punctured turbo (RCPT) and convolutional (RCPC) codes, for binary symmetric channel (BSC) with bit error rate (BER) 0.05 and 0.1 respectively. From the figure, except at very low rates for RCPC, c_i^2 remains almost constant. Among these curves, the maximum of $|c_i^2|'$ occurs at R_i = 0.0078 bpp with a value of 0.0143 and c_i^2 = 0.5714 bpp. Obviously $c_i^2 \gg |c_i^2|' R_i$. Therefore, it can be concluded that c_i^2 (and so c_i) is weakly dependent on R_i, which satisfies our second assumption.

From (5.2) and (5.9), together with the two assumptions made in this section, the ratio between the overall expected distortion for M sources coded separately and jointly is

$$\frac{\sum_{i=1}^{M} ED_i(R)}{\sum_{i=1}^{M} ED_i(R_i^*)} = \frac{\frac{1}{M}(\sum_{i=1}^{M} \sigma_i^2)}{(\prod_{i=1}^{M} \sigma_i^2)^{\frac{1}{M}}} \tag{5.13}$$

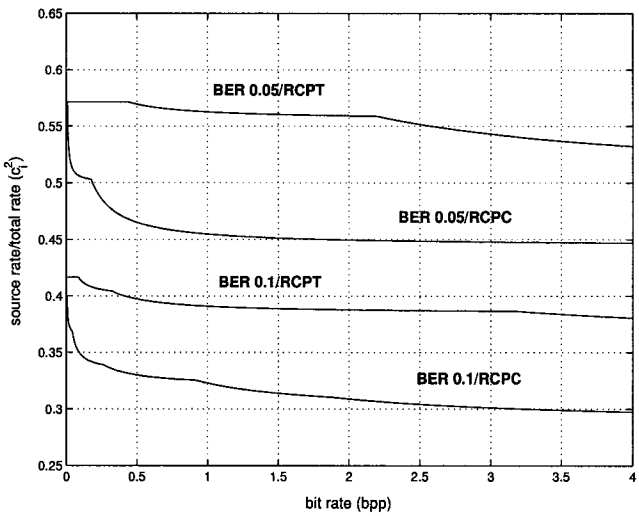


Figure 5.2: The coefficient c_i^2 as functions of total rate R_i.

which is the ratio between the arithmetic mean and the geometric mean of the source variances. Since the arithmetic mean is always equal to or larger than the geometric mean, the overall expected distortion can be reduced if multiple sources are coded jointly. This gain is called "quality multiplexing" gain hereafter and it is defined as

$$\text{Quality Multiplexing Gain} = \text{PSNR}_{joint} - \text{PSNR}_{separate} \tag{5.14}$$

Based on (5.11) and (5.13), it is clear that by jointly coding multiple sources with a goal to minimize the overall expected distortion, it is possible to obtain both improved and more uniform quality of reconstructed frames at the receiver. A specific joint coding algorithm to achieve these goals is described next.

5.4 Joint Source/Channel Coding System

5.4.1 Single Source Case

Let L_s be the bitstream length (corresponding to a given bit rate R_s) and let N_s be the corresponding number of channel packets allocated to a source s. Because each channel packet has a fixed length, N_s can be obtained by dividing L_s by the packet length. Let $V_s = [r_s^1 r_s^2 \cdots r_s^{N_s}]$ be a vector representing the channel code rates assigned to the N_s channel packets. Therefore, V_s determines, for source s, the rate allocation between its source coder and channel coder, and the protection level each source segment receives from the channel codes.

As discussed before, for a source s, its channel decoding stops whenever a channel decoding failure occurs or all the channel packets for the source have been correctly decoded. Based on this practice, the expected distortion for source s is

$$\begin{aligned} ED_s(V_s) &= D_{s,0} - E[\Delta D_s(V_s)] \\ &= D_{s,0} - \sum_{i=1}^{N_s}(\sum_{j=1}^{i} d_s^j(r_s^j))(\prod_{j=1}^{i}[1 - Pe(r_s^j)])Pe(r_s^{i+1}) \end{aligned} \tag{5.15}$$

where $D_{s,0}$ is the zero rate distortion of source s, $E[\Delta D_s(V_s)]$ is the expected distortion reduction when V_s is employed. Also $d_s^j(r_s^j)$ is the distortion reduction brought by the source bytes included in the jth channel packet with code rate r_s^j. And $Pe(r_s^i)$ denotes the probability of channel packet decoding failure when code rate r_s^i is used to protect the ith packet (note that $Pe(r_s^{N_s+1})$ is defined to be 1 to indicate the end of the bitstream.)

The joint rate allocation problem for the single source case is to find a code rate vector V_s which maximizes $E[\Delta D(V_s)]$. The optimal solution can be obtained by a brute-force search. However, when the number of channel packets is large, the search is computationally prohibitive. The forward dynamic programming [the Viterbi algorithm (VA)] based algorithm from [45] can be used to solve this problem efficiently. With this algorithm, each VA stage corresponds to a channel packet, and each trellis state corresponds to an available channel code rate. As noted in [45], the VA is suboptimal in this case. However, it has low complexity and its solutions are close to optimal most of the time.

Compared to using one channel code rate to protect a source bitstream, the scheme discussed above can provide a performance improvement by applying unequal amounts of channel protection to different segments of the source bitstream according to V_s. The gain obtained in this manner for a single source is called "UEP" gain hereafter.

5.4.2 Multiple Sources Case

Let M be the number of sources to be transmitted through a common channel. Denote L_T as the aggregated bitstream length corresponding to the total bit rate MR to be shared by the M sources.

One possibility is to divide the total bit rate equally into M pieces and statically assign each piece to a source. Each source can perform the joint rate allocation as described in Section 5.4.1 to obtain a UEP gain.

However, multiple sources can be drastically different in terms of their rate-distortion characteristics. By taking this factor into account and dynamically allocating the total bit rate among multiple sources, the overall distortion reduction for the multiple reconstructed sources could be improved compared to the static rate allocation case. Meanwhile, a quality variance reduction could also be achieved when the overall expected distortion is minimized, as derived in Section 5.3.

Therefore, the goal of the proposed dynamic rate allocation algorithm for the multiple sources case is

$$\min \sum_{s=1}^{M} (D_{s,0} - E[\Delta D_s(V_s)]) \qquad \text{s.t.} \qquad \sum_{s=1}^{M} L_s(V_s) \leq L_T$$

or equivalently

$$\min\{-\sum_{s=1}^{M} E[\Delta D_s(V_s)]\} \qquad \text{s.t.} \qquad \sum_{s=1}^{M} L_s(V_s) \leq L_T \tag{5.16}$$

(5.16) can be converted into an unconstrained version by the Lagrangian multiplier method

$$\min\{-\sum_{s=1}^{M} E[\Delta D_s(V_s)] + \lambda \sum_{s=1}^{M} L_s(V_s)\} \tag{5.17}$$

Similar to [50], for a given $\lambda \geq 0$, the solution to (5.17) can be obtained by solving each term (corresponding to each source) independently. By sweeping λ from 0 to

infinity, sets $\{V_s(\lambda)\}$ and $\{L_s(\lambda)\}$ can be created for each source s. If for some λ, $\sum_{s=1}^{M} L_s(\lambda)$ equals L_T, then an optimal solution has been found.

For a given $\lambda \geq 0$, minimizing each term in (5.17) corresponds to an optimization task for each source. Combining the result from (5.15), the objective function to be minimized for source s can be written as

$$\begin{aligned} f(V_s) &= -E[\Delta D(V_s)] + \lambda L_s(V_s) \\ &= -\sum_{i=1}^{N_s}(\sum_{j=1}^{i} d_s^j(r_s^j))(\prod_{j=1}^{i}[1 - Pe(r_s^j)])Pe(r_s^{i+1}) + \lambda L_p N_s \end{aligned} \tag{5.18}$$

where L_p is the fixed channel packet length. Notice that unlike (5.15) where N_s is fixed, to minimize (5.18), $V_s = [r_s^1 r_s^2 \cdots r_s^{N_s}]$ and N_s must be optimized jointly according to the λ value.

By enumerating all permissible V_s and N_s over their search space, an optimal solution can be found for (5.18). But, when the number of channel packets or the number of channel code rates are large, this approach quickly becomes infeasible. A forward dynamic programming (VA) based approach is therefore proposed to reduce the complexity in solving (5.18). The notation is based on that in [51]. The derivation in the remainder of this subsection is for a single source s, and thus the subscript s is dropped for simplicity.

By proper rearrangement and modification, (5.18) can be rewritten as

$$f(V) = -\sum_{i=1}^{N'}\{(\prod_{j=1}^{i}[1 - Pe(r_j)])(-d_i(r_i)) + \lambda L_p \bar{\delta}(r_i)\} \tag{5.19}$$

where N' is the maximum number of channel packets that can possibly be allocated to the source, and function $\bar{\delta}(x)$ is defined as

$$\bar{\delta}(x) = 1 - \delta(x) = \begin{cases} 1\,, x \neq 0 \\ 0\,, x = 0 \end{cases}$$

In light of (5.19), for the proposed VA algorithm, the cost function at stage k, state r_k is defined as

$$g_k(r_k) = [-d_k(r_k)][1 - Pe(r_k)] + \lambda L_p \bar{\delta}(r_k) \tag{5.20}$$

where stage k corresponds to the kth channel packet ($1 \leq k \leq N'$), and state r_k belongs to the set of available channel code rates at stage k denoted by $\mathbb{R}(k)$.

The cost-to-arrive function at stage k, state r_k is defined as

$$J_1(r_1) = g_1(r_1) \qquad \text{when } k = 1$$

and

$$\begin{aligned} J_k(r_k) \;=\; & \min_{r_{k-1} \in \mathbb{R}(k-1)} \Big\{ J_{k-1}(r_{k-1}) + (\prod_{j=1}^{k-2}[1 - Pe(r_j^*)])[1 - Pe(r_{k-1})] \times \\ & [g_k - \lambda L_p \bar{\delta}(r_k)] + \lambda L_p \bar{\delta}(r_k) \Big\} \\ & \text{when } 2 \leq k \leq N' \end{aligned} \tag{5.21}$$

where

$$r_j^* = \begin{cases} \underset{r_{k-2} \in \mathbb{R}(k-2)}{\text{argmin}} \; J_{k-1}(r_{k-1}) \,, j = k-2 \\ \underset{r_j \in \mathbb{R}(j)}{\text{argmin}} J_{j+1}(r_{j+1}^*) \quad , 1 \leq j \leq k-3 \end{cases} \tag{5.22}$$

The quality scalable property of the source bitstream implies a non-decreasing channel code rate assignment. This can be imposed as $r_i \leq r_{i+1}$ $(1 \leq i \leq N')$. At each stage of the cost-to-arrive function $J(r_k)$, in addition to the states corresponding to the available channel code rates, a 0-value entry is added to the state space to indicate that only the first $(k-1)$ channel packets are required to minimize (5.18).

Therefore, for each source s and a given λ, the proposed VA algorithm proceeds from the first channel packet to the last one that can be possibly included by the source, computing the cost-to-arrive using (5.21) for each packet (stage) and the available states at that stage. The minimal cost associated with a state in the last stage becomes the optimal cost, and the path which leads to this state from the first stage given by (5.22) determines the optimal rate allocation for source s.[1]

From the derivation for the multiple sources case, it is easy to observe that when $\lambda = 0$ and N_s is fixed (this also leads to the exclusion of the 0-value state from the code rate space for each stage), the algorithm becomes the VA optimization for the single source case as given in [45]. So single source optimization is a special case of the multiple sources case when N_s is specified and λ is set to 0.

[1]The VA is not always strictly guaranteed to find the optimal solution. This point is discussed carefully in Section 5.4.3.

5.4.2.1 Rate Allocation Adjustment

Most practical scalable source coders can provide only a finite number of operational points on a source's distortion rate curve. A scalable bitstream cannot generally be truncated at any arbitrary point while still achieving optimal distortion performance [2] . Furthermore, unequal error protection is provided mainly based on a finite set of channel codes. Because of the discrete nature of the problem, with the use of Lagrangian optimization, for any given total bit rate L_T, it may be impossible to find λ such that $\sum_{i=1}^{M} L_s(\lambda)$ is exactly equal to L_T. Consequently, the optimal solution cannot always be achieved by the above Lagrangian optimization based method. A rate allocation adjustment algorithm is proposed to address this problem.

Suppose with Lagrangian optimization, a λ value is found which gives $\sum_{i=1}^{M} L_s(\lambda)$ that is closest to L_T. Denote L_G as the gap between L_T and $\sum_{i=1}^{M} L_s(\lambda)$, which can be positive or negative. Because the goal of the algorithm is to maximize the overall distortion reduction for the multiple images, when $L_G > 0$, it is desirable to include more from the source which gives the greatest distortion reduction. On the other hand, when $L_G < 0$, some bits should be trimmed from the source which brings the smallest distortion reduction.

Suppose after the Lagrangian optimization, a total given bit rate cannot be achieved. A λ value can be chosen which gives the smallest $|L_G| \neq 0$. Each source s

[2]For example, most embedded bitstreams can only be truncated optimally at the end of a "coding pass" (e.g., SPIHT).

is allocated N_s channel packets and there is a distortion reduction $(\Delta d)_s$ associated with its last included packet. Based on the $(\Delta d)_s$ information, a source s can be chosen according to the criteria of the previous paragraph [3] . This source is then augmented or trimmed to decrease L_G. One channel packet is added to or subtracted from the source.

Adding or subtracting a packet to/from the chosen source requires the the re-optimization of source s with N_s+1 (or N_s-1) channel packets. Because the number of channel packets is then specified, the problem can be readily solved using the single source optimization. Recall this is just a special case of the multiple sources case. To be more specific, the optimization can be done by setting $\lambda = 0$ and specifying a new fixed number of packets (N_s+1 or N_s-1) in the corresponding multiple sources algorithm. This selection/optimization procedure can be carried out one channel packet a time until the total bit rate requirement is satisfied.

5.4.2.2 Algorithm Summary

The proposed dynamic rate allocation for multiple sources can be divided into two levels. At the lower level, each source performs joint rate allocation with a dynamic programming algorithm for a given λ to achieve UEP gain. At the higher level, the value of λ is adjusted to achieve the desired total bit rate. The higher level is aimed to give both quality and variance multiplexing gains in addition to the UEP gain

[3]Note that when adding more data, we decide which source to augment based on Δd_s even though we should use Δd_{s+1}. This reduces complexity with negligible suboptimality.

provided by the lower level. Note that other optimization schemes which solve (5.18) can be used instead of dynamic programming.

When the algorithm in the previous paragraph cannot provide a solution that exactly satisfies a given total bit rate, the rate adjustment part of the algorithm is called for. It executes (in an iterative fashion) to augment or trim one channel packet for one source at a time. This procedure is repeated until the total desired rate is reached.

Because all the lower-level source optimizations are independent of each other, they can be done in parallel, which leads to a total algorithm processing time possibly independent of the number of sources. In this case a central controller is needed to coordinate the rate allocation for the multiple sources, as shown in Fig. 5.1. Its job is simply to assign choices of λ to all the local rate allocators and to collect and add the returned bitstream lengths from each source, and compare with the given total bit rate. Because of the monotonic relationship between λ and the resulting total length, a bisection strategy can be adopted by the controller to find the proper λ. Therefore, the controller has very low complexity.

5.4.3 An Optimality Improvement

When the dynamic programming approach is employed in the above optimization, a term $(\prod_{j=1}^{k-2}[1 - Pe(r_j^*)])[1 - Pe(r_{k-1})]$ associated with all the previous stages is multiplied with the adjusted cost function in the current stage as shown in (5.21).

With the presence of this term, the cost of each stage is not simply additive and hence dynamic programming cannot guarantee an optimal solution [45]. Depending on the actual optimization space and the number of stages involved, in some cases the solution given by the dynamic programming approach could be significantly worse than the optimal one. When this happens, some sources may yield substantially suboptimal rate allocations which affect their UEP gains and also both quality and variance multiplexing gains.

To that end, an improvement over the dynamic programming optimization is proposed. It is based on a variation of the list Viterbi decoding algorithm (LVA) [26]. With conventional dynamic programming, at each state of a stage, only the minimal cost and the associated path are stored. In contrast, with the proposed LVA based approach, each state keeps a list of the L minimal costs and their paths. For each state, the new trellis search involves the computation of all L candidate states from the previous stage and the resulting L best costs (and their paths) are selected and ordered to form the list associated with the current state. With conventional LVA, at the final stage, its best metric is the same as the VA. If the path associated with this metric cannot satisfy some constraint (such as an inner CRC code), the LVA searches for the path corresponding to the next best metric, until all stored metrics and their paths are exhausted. However, with the proposed LVA, the minimal cost (and its path) at the final stage can be equal or better than the VA. And only the

minimal cost and its associated path of all the states at the final stage is chosen to give the desired solution.

An example of the proposed LVA based approach is shown in Fig. 5.3. It has 3 discrete channel rates r_1, r_2 and r_3 ($r_1 > r_2 > r_3$) corresponding to 3 states at each stage, and each stage corresponds to a channel packet to be assigned with a channel rate. The non-decreasing channel code rate constraint is imposed such that a state in the current stage can only be accessed by the states of equal and lower channel code rates from the previous stage.

Because the VA is a special case of the proposed LVA with its list depth being 1, the proposed approach gives a consistent but flexible method to adjust the optimality and complexity for both single and multiple source optimization. Since the lower-level optimization is independent for each source, when a source has enough computational resources, it can choose a LVA with large depth to achieve most of the UEP gain; while with a resource limited source, a faster but possibly more suboptimal VA optimization can be called.

5.5 Experimental Results

In the following experiments, multiple sources coded by JPEG2000 are transmitted through binary symmetric channels (BSC) with different crossover probabilities and total bit rates. Experiments are conducted with and without a temporal wavelet transform.

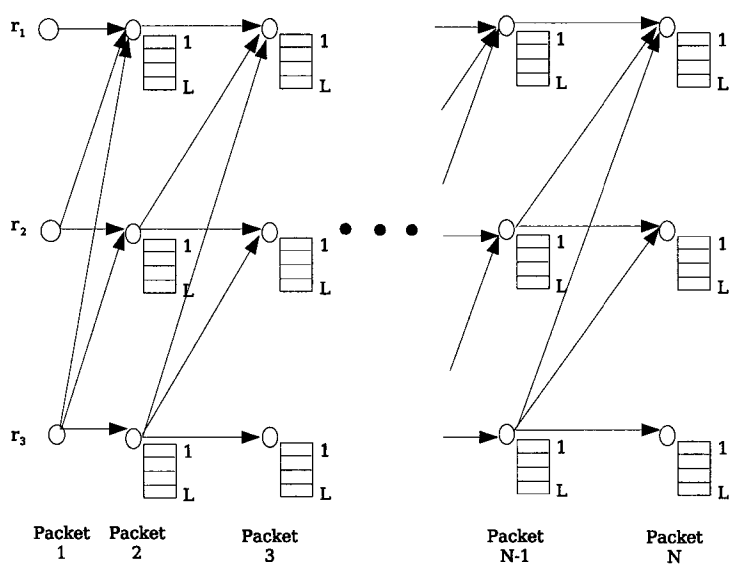


Figure 5.3: A constrained 3-state, N-stage LVA optimization with list depth of L.

Rate compatible punctured turbo (RCPT) codes [28] are employed as our channel codes. A rate $\frac{1}{3}$ parallel concatenated convolutional coder with generator polynomial $(33/31)_{oct}$ is chosen as the encoder, and a BCJR-MAP decoder is used with a maximum of 20 iterations. An S-random interleaver [52] is used with $s = \lfloor \sqrt{N/2} \rfloor$, where N is the number of message bits plus memory flush bits in a codeword. Puncturing patterns are chosen from [28] with a puncturing period of 8 to provide a set of codes for each channel, with trade-offs between rates and protection levels. Each channel packet has a fixed length of 500 bytes. Inside each packet, 4 bytes are used for CRC-32 with generator polynomial $(40460216667)_{oct}$ as an inner code to indicate a turbo decoding failure, and 1 byte is reserved for transmitting side information (such as the turbo decoding rate for the next channel packet and the source ownership of the received channel packet.)

For BSC 0.01, code rates $\{\frac{8}{9}, \frac{4}{5}, \frac{8}{11}, \frac{2}{3}\}$ are chosen with $\{439, 394, 358, 327\}$ source bytes included in the corresponding channel packets according to the packet structure adopted. Their probabilities of packet decoding failure $[Pe(r_i)]$ for the channel are: $\{0.8730, 1.1675 \times 10^{-2}, 1.2037 \times 10^{-3}, 1.30 \times 10^{-4}\}$ obtained based on 2×10^5 simulations for each rate. Similarly, for BSC 0.1, code rates $\{\frac{4}{9}, \frac{8}{19}, \frac{2}{5}, \frac{8}{21}, \frac{4}{11}, \frac{1}{3}\}$ are chosen with $\{216, 206, 194, 185, 176, 161\}$ source bytes included within each corresponding channel packet. Their probabilities of packet decoding failure for the channel are:

(a) Lenna (b) Whitehouse

$\{5.119\times10^{-3}, 1.125\times10^{-3}, 8.650\times10^{-4}, 5.15\times10^{-4}, 2.10\times10^{-4}, 5.00\times10^{-5}\}$, respectively. In the following experiments, the reported mean PSNR values were calculated by averaging the decoded MSE values and then converting the mean MSE to PSNR.

In the first experiment, the assumptions made in Section 5.3 are verified. Two images Lenna and Whitehouse (each is a 512×512, 8-bit gray-level image) are chosen as our test images and they are shown in Fig. 5.5. They have very different distortion-rate characteristics, as demonstrated in Fig. 5.4.

JSCC is performed separately for the two images for BSC 0.01 at total bit rates of 1.00, 0.50 and 0.25 bpp respectively, with a LVA depth of 25. The coefficients c_i^1, c_i^2 and c_i are obtained for both images and are listed in Table 5.1. The coefficient c_i^2 is obtained directly from the JSCC scheme, while c_i^1 and c_i are obtained by extensive

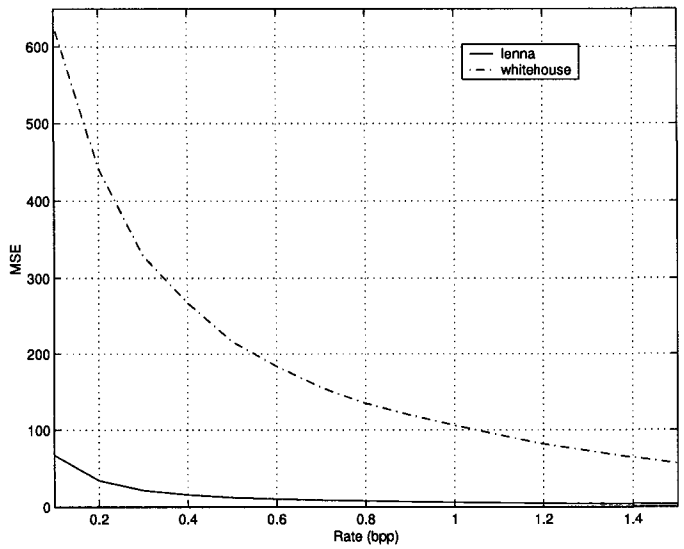


Figure 5.4: Distortion-Rate curves of Lenna and Whitehouse.

channel simulations with at least 2000 trials for each case. It can be seen that the coefficient c_i^1 is varying in a small range (an interval of about 0.02) close to 1. Additionally, c_i^2, c_i are seen as insensitive to the different images and total bit rates. A maximum ratio of only 1.04 between the values of c_i is observed for the two images. This should be compared to the ratio of image variances, which is 1.82. This demonstrates the potential of obtaining the variance multiplexing gain as derived in Section 5.3.

Table 5.1: The coefficients for different images at different total bit rates.

Images	1.00 bpp			0.50 bpp			0.25 bpp		
	c_i^1	c_i^2	c_i	c_i^1	c_i^2	c_i	c_i^1	c_i^2	c_i
Lenna	0.94	0.71	0.67	0.96	0.72	0.69	0.94	0.71	0.67
Whitehouse	0.94	0.71	0.67	0.92	0.73	0.67	0.93	0.72	0.67

To show the utility of the LVA approach, optimization is performed with the Lenna image for BSC 0.01. A small number of stages are used to make a manageable comparison with exhaustive search. Fig. 5.5 shows the difference of $f(V_s)$ values when $\lambda = 0$ (see (5.16)) between exhaustive search (EX), the Viterbi algorithm (VA) and the proposed List VA (LVA) with list depths of 10, 20 and 30 respectively. Although the figure shows only the single source case, the results are similar for multiple sources. From the figure, when the list depth is large enough, the proposed LVA approach always achieves much smaller margin than the VA. In most cases, the LVA approach achieves the optimum given by the exhaustive search.

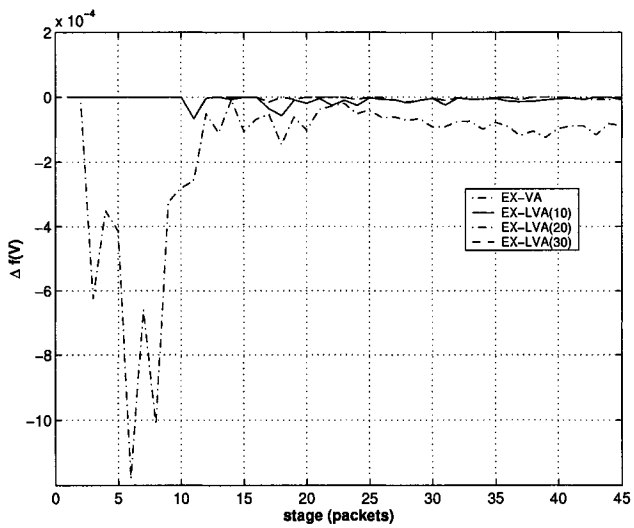


Figure 5.5: Differences of optimization results between different schemes for the Lenna image over BSC 0.01.

Five images, Lenna, Peppers, Goldhill, Baboon and Whitehouse, are chosen as test images. Each image is a 512 × 512, 8-bit gray-level image. These images are transmitted with total bit rates 1.00, 0.50 and 0.25 bpp over BSC with crossover

probabilities of 0.01 and 0.1 respectively. JPEG2000 is selected as the source coder, and layering functionality is employed during the source encoding to generate quality scalable bitstreams.

The performance based on the proposed LVA algorithm with a list depth of 25 is further compared with that of the VA for both channels and all total bit rates. The comparisons are made for single source cases, where both the VA and LVA algorithms are employed to provide UEP gains. All the results are obtained based on simulation with at least 2000 trials for each image and the results are listed in Tables 5.2 and 5.3. Up to 0.22 dB/source has been observed for BSC 0.01 and up to 0.45 dB/source for BSC 0.1 for the three total bit rates.

Table 5.2: Differences of optimization performance between LVA(25) and VA for BSC 0.01.

PSNR (dB)	1.00 bpp		0.50 bpp		0.25 bpp	
	VA	LVA	VA	LVA	VA	LVA
Lenna	38.61	38.54	35.55	35.60	32.18	32.25
Peppers	36.66	36.63	34.60	34.63	31.85	32.01
Goldhill	34.52	34.56	31.62	31.73	29.22	29.44
Baboon	27.03	26.98	24.12	24.09	22.24	22.29
Whitehouse	25.76	25.96	23.34	23.31	21.30	21.27
$\sum(\text{PSNR}_{\text{LVA}} - \text{PSNR}_{\text{VA}})$	0.09 dB		0.13 dB		0.47 dB	

5.5.1 Joint Coding of Multiple Images with JPEG2000

In the following experiments, joint source/channel coding is performed for both the static and dynamic rate allocations, which are based on the single source and multiple sources cases in Section 5.4.1 and 5.4.2, respectively. In the single source case, the

Table 5.3: Differences of optimization performance between LVA(25) and VA for BSC 0.1.

PSNR (dB)	1.00 bpp		0.50 bpp		0.25 bpp	
	VA	LVA	VA	LVA	VA	LVA
Lenna	35.57	35.55	32.54	32.74	29.69	29.72
Peppers	34.34	34.56	32.14	32.38	29.37	29.42
Goldhill	31.53	31.98	29.45	29.66	27.50	27.68
Baboon	24.38	24.39	22.51	22.54	21.23	21.21
Whitehouse	23.38	23.57	21.52	21.62	20.05	20.11
$\sum(\text{PSNR}_{\text{LVA}} - \text{PSNR}_{\text{VA}})$	0.85 dB		0.78 dB		0.3 dB	

UEP results based on the LVA algorithm with a list depth of 25 are compared with those in the corresponding equal error protection (EEP) cases. The EEP scheme is realized by protecting an entire bitstream with code rate $\frac{2}{3}$ for BSC 0.1 and code rate $\frac{1}{3}$ for BSC 0.01 respectively. These code rates are the highest that provide close to error free decoding with their corresponding channels, consistent with the criterion used in [38]. In the multiple sources case, additional quality and variance multiplexing gains are compared with UEP gains provided by the single source case. For each case, at least 2000 simulations are conducted.

In every case, joint allocation is performed based on the LVA algorithm with a list depth of 25. Corresponding to 1.00, 0.50 and 0.25 bpp/source and 500 bytes/packet, there are a total of 330, 165 and 80 channel packets to be allocated among 5 images respectively. With static bit allocation, each image gets an equal number of channel packets. In contrast, the number of channel packets allocated to each image can be very different in the dynamic bit allocation case.

The bit allocation results for the static and dynamic rate allocations for the two test channels and different total bit rates are listed in Tables 5.4 to 5.9. Along with them are the simulation results for the average MSE (with its corresponding PSNR) and MSE variance for each of the static and dynamic case.

From these tables, unequal error protection can provide a performance improvement in the static bit allocation case. However, a substantial quality multiplexing gain can be achieved in addition by coding these images jointly in the corresponding dynamic cases. For the cases with BSC 0.01, channel multiplexing gains are 6.60, 3.14 and 2.90 times as much as the corresponding UEP gains. For BSC 0.1, the ratios are 2.11, 1.28 and 0.55, respectively. At the same time, the variance multiplexing gains between the joint coding cases and the corresponding UEP coding cases are 90%, 79% and 68% for BSC 0.01, and 72%, 62% and 50% for BSC 0.1, respectively.

Note that both gains decrease along with the total bit rate for both channels. This is because with a reduced total bit rate, the corresponding source rate is effectively reduced as well, which limits the rate-distortion diversity that can be exploited by the proposed algorithm. It also explains the gain drop in BSC 0.1 compared to BSC 0.01. For the same total bit rate, more bits are dedicated to the channel coding for BSC 0.1 and hence the source rate becomes smaller.

Table 5.4: Performance comparison at BSC 0.01 and 1.00 bpp/source for multiple images.

1.00 bpp/source	single source			multiple sources	
	number of packets	EEP MSE	UEP MSE	number of packets	UEP MSE
Lenna	66	9.53	9.09	23	25.71
Peppers	66	14.13	14.04	24	29.50
Goldhill	66	24.31	22.78	35	41.91
Baboon	66	140.64	131.31	130	53.74
Whitehouse	66	172.88	164.98	118	82.42
Ave. MSE	–	72.30	68.33	–	46.66
PSNR (dB)		29.54	29.79		31.44
Var. MSE	–	6.09×10^3	5.46×10^3	–	5.21×10^2

Table 5.5: Performance comparison at BSC 0.01 and 0.50 bpp/source for multiple images.

0.50 bpp/source	single source			multiple sources	
	number of packets	EEP MSE	UEP MSE	number of packets	UEP MSE
Lenna	33	19.90	17.91	15	40.34
Peppers	33	23.78	22.38	15	43.81
Goldhill	33	46.21	43.69	16	74.15
Baboon	33	274.63	253.37	59	145.68
Whitehouse	33	313.73	301.17	60	181.27
Ave. MSE	–	135.65	125.04	–	97.05
PSNR (dB)		26.81	27.16		28.26
Var. MSE	–	2.12×10^4	1.90×10^4	–	4.01×10^3

Table 5.6: Performance comparison at BSC 0.01 and 0.25 bpp/source for multiple images.

0.25 bpp/source	single source			multiple sources	
	number of packets	EEP MSE	UEP MSE	number of packets	UEP MSE
Lenna	16	41.59	38.72	8	73.45
Peppers	16	44.48	40.94	8	77.07
Goldhill	16	80.99	74.00	6	137.25
Baboon	16	402.80	383.68	32	254.90
Whitehouse	16	502.60	485.49	26	352.53
Ave. MSE	–	214.49	204.57	–	179.04
PSNR (dB)		24.82	25.02		25.60
Var. MSE	–	4.88×10^4	4.56×10^4	–	1.48×10^4

Table 5.7: Performance comparison at BSC 0.1 and 1.00 bpp/source for multiple images.

1.00 bpp/source	single source			multiple sources	
	number of packets	EEP MSE	UEP MSE	number of packets	UEP MSE
Lenna	66	20.12	18.14	34	34.27
Peppers	66	24.14	22.73	30	41.55
Goldhill	66	46.76	41.19	40	61.64
Baboon	66	274.63	236.63	119	137.04
Whitehouse	66	319.02	286.08	107	193.42
Ave. MSE	–	136.93	120.95	–	93.58
PSNR (dB)		26.77	27.30		28.42
Var. MSE	–	2.16×10^4	1.68×10^4	–	4.78×10^3

Table 5.8: Performance comparison at BSC 0.1 and 0.50 bpp/source for multiple images.

0.50 bpp/source	single source			multiple sources	
	number of packets	EEP MSE	UEP MSE	number of packets	UEP MSE
Lenna	33	40.97	34.56	15	77.43
Peppers	33	43.60	37.62	23	55.42
Goldhill	33	79.01	70.28	17	107.38
Baboon	33	397.31	362.37	59	256.21
Whitehouse	33	497.81	447.81	51	335.08
Ave. MSE	–	211.74	190.53	–	166.30
PSNR (dB)		24.87	25.33		25.92
Var. MSE	–	4.78×10^4	3.95×10^4	–	1.51×10^4

Table 5.9: Performance comparison at BSC 0.1 and 0.25 bpp/source for multiple images.

0.25 bpp/source	single source			multiple sources	
	number of packets	EEP MSE	UEP MSE	number of packets	UEP MSE
Lenna	16	86.84	69.35	11	95.16
Peppers	16	94.49	74.38	11	107.68
Goldhill	16	124.24	110.84	10	142.51
Baboon	16	535.57	492.18	20	448.60
Whitehouse	16	717.53	634.58	28	469.08
Ave. MSE	–	311.73	276.72	–	258.05
PSNR (dB)		23.19	23.72		24.01
Var. MSE	–	8.69×10^4	7.15×10^4	–	3.58×10^4

5.5.2 Joint Coding of Multiple Video Sequences with 3D-JP2

In this experiment, the transmission of video sequences is considered for relatively low resolutions and bit rates. 3D-JPEG2000 is selected as the source coder, and layering functionality is employed during source encoding to generate quality scalable bitstreams.

Three video sequences, "Carphone", "Mother&Daughter" and "Foreman" are chosen as our test sequences. Each video sequence is in QCIF format (176×144) and has 8-bit graylevel depth. 16 consecutive frames from a sequence are combined to form a GOF as an independent coding unit for employing a temporal wavelet transform with JPEG2000, as in [16], [17] and [78]. The first 16 frames are taken as a GOF for both "Carphone" and "Foremen" sequences and frame 101 to frame 116 are taken as a GOF for "Mother&Daughter" sequence.

The three video sequences are transmitted over BSC with a crossover probability of 0.01. An aggregated bit rate of 90 kbps at 10 fps is shared by three GOFs, each coming from a different video sequence. With the same channel packet setting as Section 5.5.1, the total bit rate corresponds to 36 channel packets shared by three GOFs. Therefore, with the static allocation, each GOF is assigned with 30 kbps (or 12 channel packets). In contrast, with dynamic allocation, each GOF can be assigned a different bit rate (or number of channel packets), with the aim to minimize the overall expected distortion.

Joint source/channel coding is performed for static allocation to obtain UEP gain, and for dynamic allocation to obtain both quality and variance multiplexing gains, respectively. The proposed LVA is employed with a list depth of 25 in both cases, and at least 2000 trials are conducted for each case. The mean decoded MSE value for each GOF was obtained by averaging the decoded MSE values over the 16 frames in the GOF and then averaging over all the trials. The simulation results are shown in Table 5.10. From the table, UEP can provide about 0.3 dB/sequence improvement, and joint coding can provide an additional 0.3 dB/sequence quality multiplexing gain and about 62% variance multiplexing gain compared to the corresponding UEP case.

Table 5.10: Performance comparison at BSC 0.01 and 90 kbps for multiple video sequences.

90 kbps	single source			multiple sources	
	number of packets	EEP MSE	UEP MSE	number of packets	UEP MSE
Carphone	12	59.04	55.64	10	65.47
Mother&Daughter	12	25.87	23.58	9	30.97
Foreman	12	135.42	128.38	17	97.42
Ave. MSE	–	73.44	69.20	–	64.62
PSNR (dB)		29.47	29.73		30.03
Var. MSE	–	3.15×10^3	2.88×10^3	–	1.10×10^3

5.5.3 Joint Coding of Multiple HDTV Sequences with MJP2

The experiments in this section are conducted for HDTV sequences coded by Motion JPEG2000. Three progressive HDTV sequences Blue_sky, River_bed and Tractor are chosen as out test sequences. Only the luminance components of the

first 200 frames from each sequence are considered, where each luminance component (frame) is a 1920×1080 8-bit gray-level image. The following multiplexing dimensions are considered and listed along with their acronyms:

1. (s,s): separate coding of each frame from the 3 sequences (600 encodings)
2. (j,s): joint coding of 3 frames, one from each sequence (200 encodings)
3. (s,j): joint coding of 200 frames from one sequence (3 encodings)
4. (j,j): joint coding of all 600 frames from all sequences (1 encoding).

It is clear then, that (j,s) exploits the diversity between the sequences one frame at a time, (s,j) exploits the diversity within one sequence and (j,j) exploits the diversity across both sequences and frames.

Because of the vast amount of data involved with HDTV sequences, real channel simulation is prohibitive. However, from the previous observation that the coefficient c_i^1 changes only in a small range close to 1, it is reasonable to use the error free performance based on R_i^s at the transmitter as an indicator of the noisy performance that would be achieved at the receiver.

The three HDTV sequences are coded at 25 and 50 Mbps, 30 fps, and for BSC 0.1 with the four multiplexing strategies. The VA based algorithm is employed to reduce computational complexity. The reported PSNR values are calculated by converting from the corresponding average MSE values. In order to estimate the UEP gain in the static allocation case, two frames from the sequence Blue_sky are picked (frame 2 and frame 191) which have different rate-distortion characteristics. Joint source/channel

coding is performed according to Section 5.4.1 for both frames at 25 Mbps and the resulting bitstreams are channel simulated with at least 500 trials for each case. For frame 2, EEP rate allocation gives 36.43 dB while UEP rate allocation gives 36.85 dB. For frame 191, they are 26.81 dB and 27.12 dB, respectively. Thus, about 0.3 to 0.4 dB UEP gains can be obtained by static rate allocation.

Error free performance of coding the HDTV sequences with different multiplexing strategies is listed in Tables 5.11 and 5.12. Compared to the non-multiplexed (s,s) case, about 0.32 to 0.79 dB per frame quality multiplexing gain and about 68% to 83% variance multiplexing gain can be achieved at 25 Mbps. At 50 Mbps, about 0.52 to 0.76 dB per frame quality multiplexing gain and about 87% to 92% variance multiplexing gain can be obtained. These quality multiplexing gains are obtained in addition to the UEP gains given by the usual (s,s) JSCC.

Table 5.11: Performance comparison among different multiplexing dimensions at 25 Mbps for multiple HDTV sequences.

25 Mbps	(s,s)		(s,j)		(j,s)		(j,j)	
	MSE	var MSE	MSE	var MSE	MSE	var MSE	MSE	var MSE
Blue_sky	40.99	962.96	36.88	285.41	25.79	174.55	25.07	101.19
River_bed	20.07	2.97	20.19	6.93	22.50	2.14	21.77	2.70
Tractor	14.09	67.30	12.82	31.92	14.82	45.57	15.92	68.69
Average	25.05	344.41	23.30	108.09	21.04	74.09	20.92	57.53
PSNR (dB)	34.14	-	34.46	-	34.90	-	34.93	-

Finally, the noise free performance for each frame of Blue_sky sequence is shown in Fig. 5.6 for all four multiplexing strategies at 50 Mbps. The original sequence has hard-to-compress frames at the end, which causes a large MSE variation at the

Table 5.12: Performance comparison among different multiplexing dimensions at 50 Mbps for multiple HDTV sequences.

50 Mbps	(s,s)		(s,j)		(j,s)		(j,j)	
	MSE	var MSE	MSE	var MSE	MSE	var MSE	MSE	var MSE
Blue_sky	15.59	172.87	12.82	20.34	8.49	17.74	8.57	11.69
River_bed	9.50	38.35	9.44	0.73	10.58	0.63	10.44	0.47
Tractor	6.28	40.15	5.58	2.86	7.30	2.75	7.29	2.57
Average	10.45	61.65	9.28	7.98	8.79	7.04	8.76	4.91
PSNR (dB)	37.94	-	38.46	-	38.69	-	38.70	-

receiver when each frame is coded separately. However, by joint coding with the three different multiplexing strategies, the MSE variation is significantly reduced, resulting in more uniform quality across the reconstructed frames at the receiver.

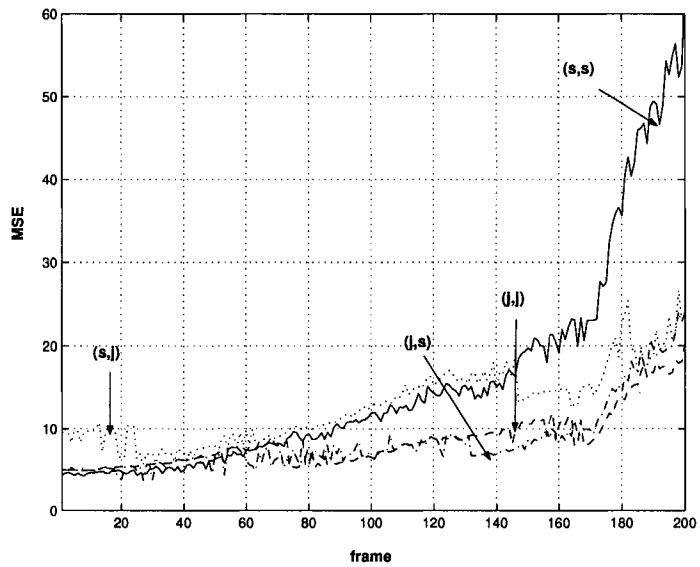


Figure 5.6: Blue_sky performance with different multiplexing strategies at 50 Mbps.

5.6 Conclusion

In this chapter, a joint source/channel coding algorithm is proposed for the transmission of multiple image or video sources. It exploits the rate-distortion diversity among different sources and optimally distributes a given total bit rate to each source. It is demonstrated that by coding multiple sources jointly, quality improvement and a reduction in quality variation are achievable at the same time. Experimental results show the advantage of the proposed algorithm with sources coded by JPEG2000, JPEG2000 with temporal wavelet transform and Motion JPEG2000.

CHAPTER 6

ROBUST TRANSMISSION OF JPEG2000 CODESTREAMS OVER PACKET ERASURE CHANNELS

6.1 Introduction

With the fast development of computer networks, more and more data, including compressed images, are transmitted through packet switched networks, such as the Internet. However, during transmission, channel packets could be lost at random when the number of packets sent exceeds transmission capacity. New generation image coders, such as SPIHT [37] and JPEG2000 [18], can provide efficient compression for transmission purposes. Through the extensive use of context-based entropy coding, their codestreams are very sensitive to packet losses. So it is important to make these image coders more error resilient.

Recently, several methods have been proposed for wavelet based image coders to improve their error resilience and packetization efficiency over noisy channels. In [34], a modified embedded zerotree wavelet (EZW) [79] image compression algorithm was proposed. Its encoder partitions the wavelet coefficients into several groups and then quantizes and codes each of them independently. The resulting multiple codestreams are interleaved and then sent through the channel. The corresponding decoder can

use all of the bits received before the occurrence of the first error in each codestream to reconstruct the image. So the impact of a single bit error is localized to only one codestream and this codestream is decoded partially to its reduced fidelity, while the other codestreams are unaffected and can be decoded to their full fidelity. By forming more independent codestreams, the dependencies among wavelet coefficients is exploited in a smaller region by the encoder, but a single bit error affects a smaller part of the total codestream. Therefore, error robustness and source coding efficiency can be traded off by forming differing numbers of independent codestreams.

In [33] , the authors proposed a combined wavelet zerotree coding and packetization method (PZW) for packet erasure channels. It modifies the set partitioning in hierarchical trees (SPIHT) [37] coder such that complete trees of wavelet coefficients are contained within each fixed-length network packet. It provides graceful degradation in image quality as packet losses increase because each packet is independently decodable. This comes at the cost of source coding efficiency. Efficient packetization for embedded bitstreams is further studied in [80]. It solves the problem of minimizing packetization inefficiency due to bitstream alignment, which is brought up by partitioning encoded bitstreams into channel packets for error resilience purposes. Theoretically optimal packetization (OP) schemes were proposed for packet erasure channels, along with suboptimal schemes with less complexity.

All the schemes discussed above combat channel noise by modifying source coders to introduce error resilience to the coded bitstreams. The underlying idea is to

localize the error influence to a small segment of the bitstream so that it will not propagate to derail the entire decoding. This is achieved by various data partitioning and resynchronization techniques.

Another approach to combat noise is to use channel coders concatenated with source coders. A channel coder adds controlled redundancy into the source-encoded bitstreams [35] [81] [82] . With this extra redundancy, errors or erasures can be not only detected but also recovered. However, within a total transmission rate, channel coding decreases the rate available to the source coding, and it also adds extra complexity to both sender and receiver due to the channel coding.

In this chapter, a new scheme is proposed for robust image transmission over ATM packet erasure channels with JPEG2000. It achieves source coder robustness against packet loss by utilizing some error resilience functionalities provided by JPEG2000. It does not require any modification of standard-compliant JPEG2000 coders, and no channel coding technique is employed. This scheme results in a high complexity decoder. However, the encoder is spared the extra complexity of channel coding as commonly used to improve robustness. Experimental results show the effectiveness of the scheme compared with other schemes.

This chapter is organized as follows: in Section 6.2, the proposed algorithm is described. Section 6.3 gives the experimental results and Section 6.4 draws conclusions.

6.2 Algorithm Description

As discussed in Chapter 2, the JPEG2000 standard provides a rich set of error resilience mechanisms to detect and isolate errors, and conceal the error impact within a certain coding structure. However, there is no forward error correction capability. Nevertheless, we will show that by appropriately utilizing these error resilience mechanisms, together with our proposed interleaving scheme, erasures can be recovered.

Consider a JPEG2000 codestream formed with modes *ERTERM* and *RESTART*. When *ERTERM* and *RESTART* are used together, an encoder creates a separate, predictably terminated codeword segment for each coding pass. If an error occurs in the codestream, a decoder can detect it with high probability at the end of the coding pass in which it occurred. With the length information signaled in the packet header, the decoder can discard only the corrupt coding pass (and all subsequent coding passes from that same codeblock), thereby concealing the possible visual artifacts which would otherwise result from such corruption. At the same time, *packed packet header* functionality is invoked to pack all the headers at different structural levels into the main header at the beginning of the codestream.

The proposed scheme partitions the codestream into ATM network packets in two different sections, which we call the horizontal section and the vertical section, as shown in Fig. 6.1. In this figure, each row corresponds to one 48-byte ATM packet payload.

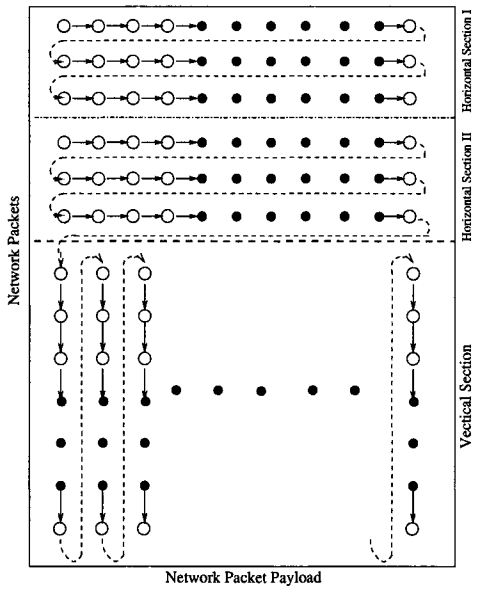


Figure 6.1: Proposed interleaving scheme.

In the first horizontal section, the main header is packed horizontally byte-wise into the payload of ATM packets. Since the main header contains critical information for correct decoding, it needs to be protected adequately. With the assumption that no channel coding is involved, we take the approach to repeat the main header to increase its robustness against packet erasures.[1] Thus the horizontal section contains two copies of the main header. Since the main header takes a relatively small part of the codestream, the duplication does not decrease compression efficiency significantly, while it can dramatically increase the protection for the main header. This approach is also used in the MPEG-4 header extension code (HEC), where vital information for correct decoding is repeated after the HEC, and has proven successful [49]. We

[1]We ignore that this can be considered as a crude form of channel coding.

note here that if desired, the repeated data can be put in a JPEG2000 "COMMENT" marker segment so that the result is a valid codestream.

In the vertical section, the compressed image data are interleaved bit-wise into the remaining packets (vertically). That is, the data are filled into the first bit of each packet continuously in the vertical direction, then the second bit, and so on, until the end of either the bitstream or the packets is reached.

Consider the situation when the packet erasure rate is relatively low. With the proposed interleaving scheme, in the horizontal section, if any packet is erased, it is very likely that its duplicate survives. Thus the main header can be reconstructed by combining the two parts from that section together, and the reconstruction can be expected to succeed with high probability.

In the vertical section, compressed image data are fully interleaved across the packets. For every erased packet in this section, the 48-byte erasures are spread out into 384 bit erasures in many coding passes from different codeblocks within the codestream. So at relatively low packet erasure rates, every coding pass may only contain a few erased bits. When the main header is reconstructed perfectly, coding pass length information can be retrieved from the corresponding packet headers. So the decoder knows the position of each coding pass within the codestream. Together with the knowledge of which packets are erased, the decoder can deduce the positions of the erased bits inside each coding pass (if any).

As discussed before, when the *ERTERM* and *RESTART* modes are used, the decoder can detect an error inside a coding pass with high reliability. Therefore, if the number of erasures inside an affected coding pass is small, the decoder can try to recover the erasures by simply brute-force bit-flipping with 0 or 1 at these erased bits until the attempted coding pass is decoded with a consistent termination state. Consequently, with high probability, a coding pass with erasures can be completely recovered. This procedure can be carried out within each codeblock from the first coding pass containing erasures and then repeated sequentially for the rest of coding passes. Suppose some coding pass contains more erasures than the decoder is willing to attempt to recover (for complexity reasons). In this case, the decoder simply stops decoding and discards the corrupted coding pass and all subsequent coding passes within the same codeblock. The decoded coding passes from the same codeblock and the decoding of other codeblocks are unaffected.

Obviously, the chance of successful decoding of a coding pass depends on the number of erasures it contains, and the computational capacity of the decoder. In the worst case, for n erasures, the decoder must test 2^n cases. Fortunately, the data partitioning of JPEG2000 can be adjusted to control of the tradeoff between complexity and system performance. As mentioned before, the codeblock is the smallest independent coding unit. With a large codeblock size, correlation can be exploited in a larger area so high compression performance can be achieved. However, this will generate long coding passes, which span more packets vertically and therefore

contain more erasures. This weakens the recovery capability of the decoder and increases the likelihood that these coding passes will be discarded. On the other hand, with smaller codeblock sizes, source coding performance degrades. However, coding passes are generally shorter and contain fewer erasures, and so are more recoverable by the decoder. Also, with smaller codeblocks, there are more codeblocks resulting in more independently decodable sub-codestreams. A key benefit of this is that any unsuccessful recovery of a coding pass is contained to a smaller region.

In summary, codeblock size can serve as a parameter to adjust the tradeoff between source efficiency and robustness. When the packet erasure rate is low, large codeblock sizes can be chosen to achieve good compression performance; as the erasure rate increases, the codeblock size can be reduced to generate more robust codestreams accordingly.

6.3 Experimental Results

In the following experiments, Lenna and Goldhill (512×512) are used as our test images. Transmission is simulated through an ATM network with 48-byte packet payload, with total transmission rates 1.0005, 0.5024 and 0.2095 bpp, which correspond to 683, 343 and 143 channel packets respectively. We choose the codeblock size parameters from the set $\{8 \times 8, 16 \times 16, 32 \times 32\}$. Each JPEG2000 codestream is generated with a single layer, the desired codeblock size, *ERTERM+RESTART* modes turned on and *PPM* marker segment employed. The source compression rate

Table 6.1: Error free source coding performance for Lenna: Source Rate (bpp)/PSNR (dB).

settings	Total Rate		
	1.0005 bpp	0.5024 bpp	0.2095 bpp
32x32/ERTERM+RESTART/Repeat Header	0.95/39.80	0.47/36.65	0.19/32.50
16x16/ERTERM+RESTART/Repeat Header	0.91/39.07	0.45/35.91	0.18/31.88
8x8/ERTERM+RESTART/Repeat Header	0.85/37.87	0.42/34.66	0.18/30.65
64x64/ERTERM+RESTART	1.00/40.25	0.50/37.17	0.21/33.05
64x64/20-layer	1.00/40.33	0.50/37.26	0.21/33.13

Table 6.2: Error free source coding performance for Goldhill: Source Rate (bpp)/PSNR (dB).

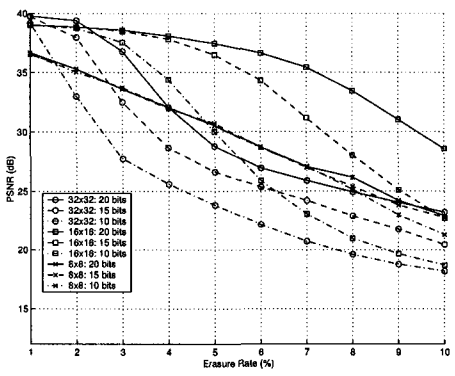
settings	Total Rate		
	1.0005 bpp	0.5024 bpp	0.2095 bpp
32x32/ERTERM+RESTART/Repeat Header	0.95/35.95	0.47/32.68	0.19/29.52
16x16/ERTERM+RESTART/Repeat Header	0.91/35.19	0.45/32.09	0.18/29.13
8x8/ERTERM+RESTART/Repeat Header	0.85/33.77	0.42/31.06	0.17/28.38
64x64/ERTERM+RESTART	1.00/36.45	0.50/33.14	0.21/29.94
64x64/20-layer	1.00/36.53	0.50/33.19	0.21/29.99

is chosen such that after the main header duplication, the total length does not exceed the allowed transmission budget.

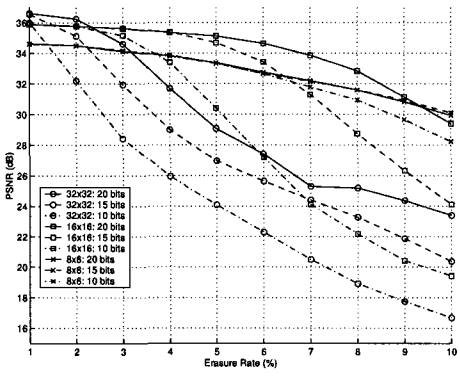
Source coding rates and PSNRs for the error free case (no erasures) are listed in Tables 6.1 and 6.2. For example, for a total rate of 1.0005 bpp, our scheme (using 32×32 codeblocks) allocates 0.95 bpp for source rate and achieves 39.80 dB PSNR in the error free case. The difference between 1.0005 bpp and 0.95 bpp accounts for the replicated main header. Also listed in the tables (for source efficiency comparison) are two more cases. Each uses 64×64 codeblocks. The first case uses *ERTERM+RESTART*, while the second case uses 20 quality layers.

The resulting codestreams were transmitted through a memoryless packet erasure channels. Each channel was simulated with an erasure rates between 1% and 10%. Kakadu software was used as the source coder, and each case was tested with 2000 trials. The results are direct average of PSNRs in order to compare with other results. In the experiments, we allow the decoder to be able to correct 10, 15 or 20 erasures in each coding pass respectively, corresponding to different levels of decoding effort. The results for different total transmission rates, codeblock sizes, and maximum recoverable erasures are shown in Fig. 6.2. From the figures, codeblock size 32×32 can give the best performance at very low erasure rates (1% or 2%) with high decoding effort. However, the performance drops quickly as the erasure rate is increased. In the middle erasure rate range (3% to 8%), codeblock size 16×16 is the best choice. We note that there are significant gaps in performance as a function of different numbers of recoverable erasures (decoding effort). Throughout the entire testing range, a codeblock size of 8×8 gives most stable performance, most of the time within 2 dB of the best possible performance. Additionally, the 8×8 case begins to show its advantage at high erasure rates (9% to 10%). Furthermore, the performance difference between different numbers of recoverable erasures are negligible, which implies that high decoding effort is unnecessary in the 8×8 case.

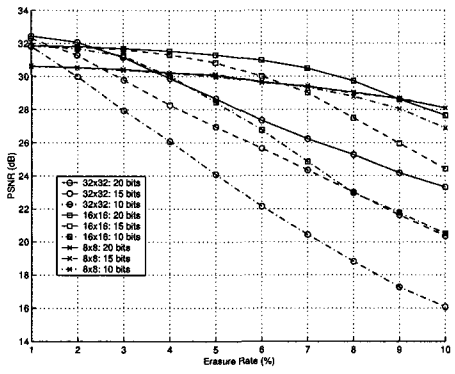
Fig. 6.3 gives the overall best performance among 3 codeblock size choices based on the results in Fig. 6.2. In other words, Fig. 6.3 represents the best possible performance by adjusting the codeblock size parameter given the maximum number of



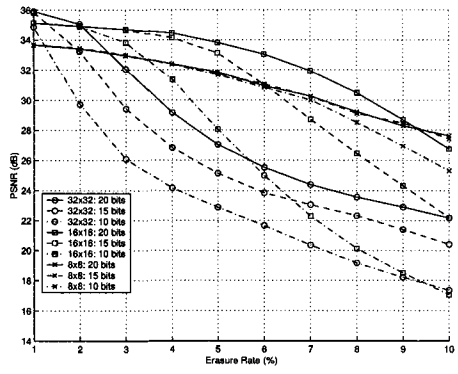
(a) Lenna 1.0005 bpp.



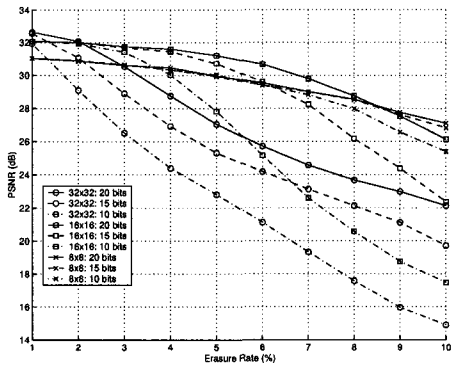
(b) Lenna 0.5024 bpp.



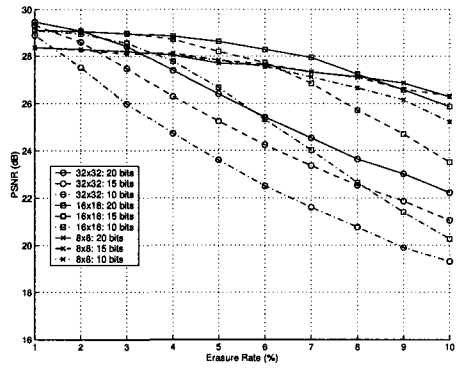
(c) Lenna 0.2095 bpp.



(d) Goldhill 1.0005 bpp.



(e) Goldhill 0.5024 bpp.

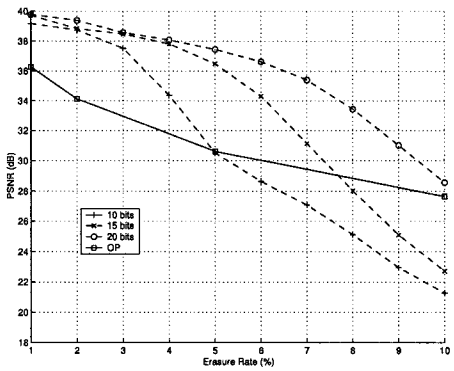


(f) Goldhill 0.2095 bpp.

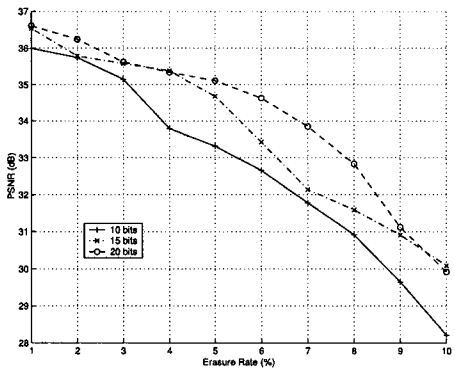
Figure 6.2: Lenna and Goldhill (512x512) with different codeblock sizes and number of recoverable erasures.

recoverable erasures at each erasure rate. At the same time, our results are compared with those from schemes OP [80] and PZW [33] in Fig. 6.3 (a) and (c). In Fig. 6.3(a) which has a high rate (1.0005 bpp), our scheme performs better than OP by 3 to 4 dB at low erasure rates even with the least decoding effort. This margin diminishes until the erasure rate reaches 5% and thereafter OP begins to perform better if higher decoding effort is not used in our scheme. In Fig. 6.3(c) with low rate (0.2095 bpp), our scheme consistently gives better performance than OP and PZW at erasure rates from 1% to 9% even with the least decoding effort.

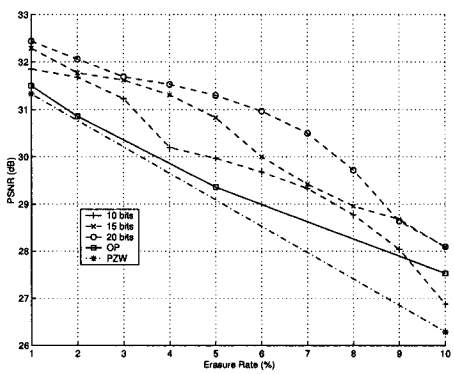
Finally, our scheme is compared with two other codestream organization methods for JPEG2000 in Fig. 6.4. The first is to generate a codestream with a single layer using *ERTERM+RESTART* and packing the codestream into channel packets horizontally. Decoding of each codeblock is attempted, with all data prior to the first erasure in each codeblock retained. This method differs from our scheme mainly in that we spend bits reproducing the header, and special interleaving is employed such that some erasures can be recovered. The second scheme generates a quality-progressive codestream with 20 layers and the codestream is packed horizontally into the channel packets. Source decoding stops with the first erased packet. As mentioned previously, the error free performance of these schemes is listed in Tables 6.1 and 6.2. From Fig. 6.4, we see significant gains of our scheme compared with the other two in all cases. This shows that the gain we have over OP and PZW is due to our proposed scheme, and not just the inherent error resilience of JPEG2000.



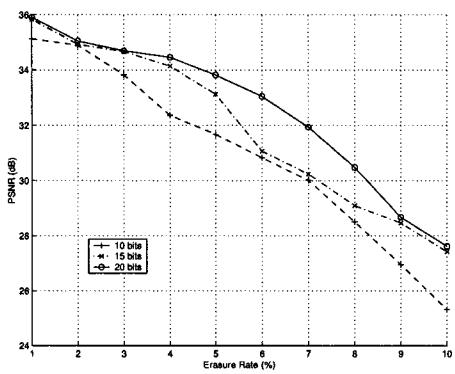
(a) Lenna 1.0005 bpp.



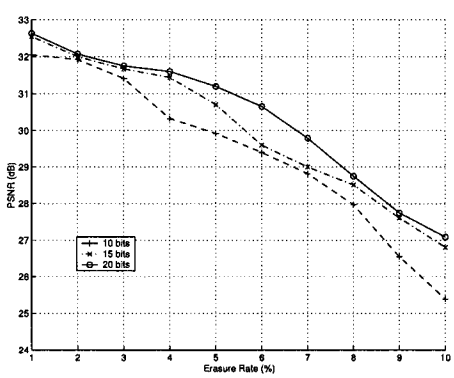
(b) Lenna 0.5024 bpp.



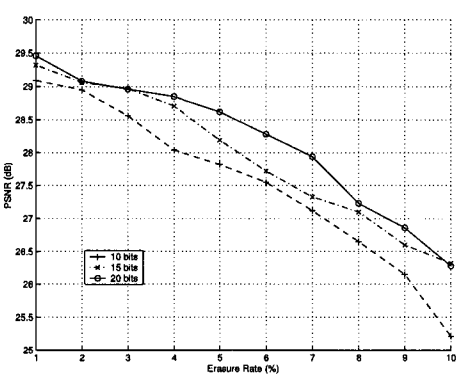
(c) Lenna 0.2095 bpp.



(d) Goldhill 1.0005 bpp.

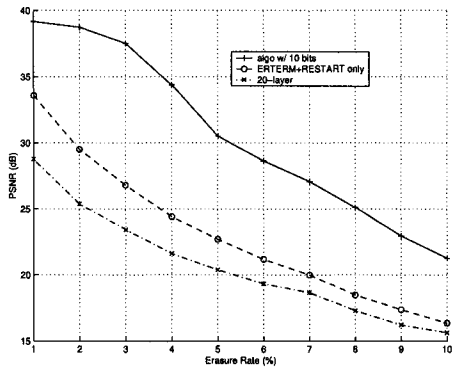


(e) Goldhill 0.5024 bpp.

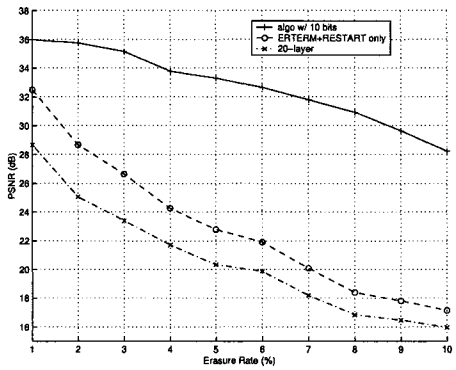


(f) Goldhill 0.2095 bpp.

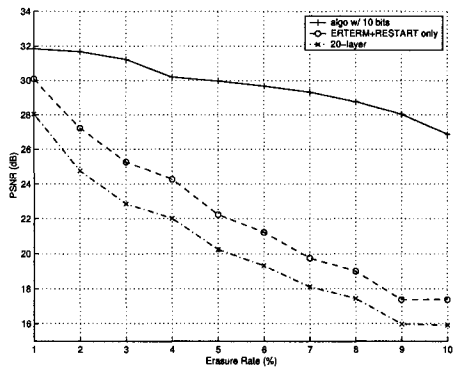
Figure 6.3: Lenna and Goldhill (512x512) best achievable performance with 10,15, and 20 recoverable erasures.



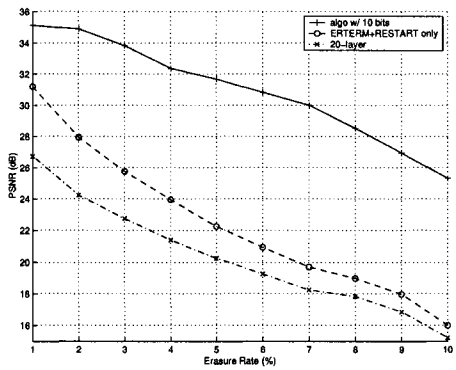
(a) Lenna 1.0005 bpp.



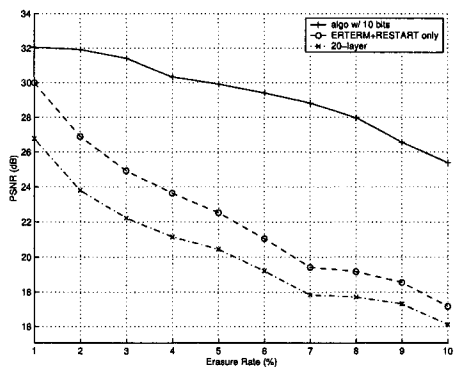
(b) Lenna 0.5024 bpp.



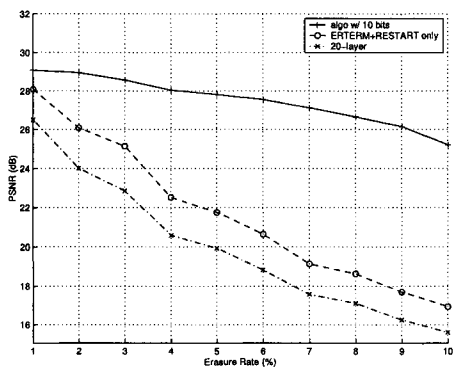
(c) Lenna 0.2095 bpp.



(d) Goldhill 1.0005 bpp.



(e) Goldhill 0.5024 bpp.



(f) Goldhill 0.2095 bpp.

Figure 6.4: Lenna and Goldhill (512x512) comparison among 3 schemes.

The applicability of the proposed scheme relies heavily on the efficient decoding of a JPEG2000 codestream. With Kakadu v3.4, the decoding of all codeblocks for Lenna and Goldhill (512×512) at 1.00 bpp takes about 0.04 seconds on a Pentium III 1GHz PC. With 10 maximum recoverable erasures, assuming the worst case of every coding pass needing recovery, the average decoding time is about 21 seconds.

Although the present work uses no channel coding, it is possible to combine it with channel coding to improve system performance. Channel coding can reduce the number of erasures for the source decoding to handle, and some erasures unrecoverable by the channel decoding can be salvaged by the source decoding, thus leading to improvement over the original channel coding capability. In [83] , a similar idea is explored for LDPC codes with AWGN channels, where the redundancy introduced for error resilient source decoding is utilized by channel decoders to improve performance and convergence speed. Besides proposing a practical method for robust image transmission, this work reveals some potential contributions of error resilience functionalities other than those in the standard, in the case of packet erasure channels. This will help form a more complete evaluation of their importance.

6.4 Conclusion

In this chapter, a scheme is proposed based on JPEG2000 for robust image transmission over packet erasure channels. The robustness against packet erasures is achieved by exploiting only the properties of error resilience functionalities provided

by JPEG2000 combined with a unique interleaving strategy. At the same time, it is also possible to combine this scheme with other channel coding techniques to give better performance. Furthermore, the importance of some error resilience functionalities of JPEG2000 is explored in a wider perspective.

CHAPTER 7

CONCLUSIONS AND FUTURE WORK

7.1 Conclusions

In this dissertation, the problem of joint source/channel coding (JSCC) is studied for image and video transmission over different communication channels.

A JSCC rate allocation scheme tailored for JPEG2000 is proposed for image transmission over channels with and without memory in Chapter 3 and Chapter 4, respectively. The goal is to transmit an image within the total bit rate given by a bandwidth limited channel such that the expected distortion of the received image is minimized. The proposed scheme generates quality scalable and error resilient source bitstreams. It combines the forward error protection capability from the channel codes and the error detection/localization capability from JPEG2000 in an effective way.

For memoryless channels, RCPC and RCPT channel codes are employed and our results are compared with other schemes. Experimental results show that the proposed scheme can provide competitive performance in matched channel condition cases. When mismatched channel conditions occur, due to the error resilience structure employed in our source bitstream, more graceful quality degradation can be

achieved. For channels with memory, LDPC codes are proposed to replace the product codes employed by most existing schemes. Due to the "built-in interleaving" property, LDPC codes are shown to be effective for correcting bursty errors. Compared to product codes, LDPC codes can provide stronger error protection with the same code rates, which gives a performance improvement. Further gains are realized by exploiting unequal error protection when the LDPC codes are combined with the proposed joint rate allocation scheme. Moreover, an LDPC construction method is proposed which gives efficient representation of the constructed LDPC codes.

In Chapter 5, a joint source/channel coding scheme is proposed for the transmission of multiple sources sharing a total bandwidth given by a common channel. The proposed algorithm exploits the rate-distortion diversity among multiple sources and optimally distributes the total bandwidth among them in order to minimize the overall expected source distortion at the receiver. It is shown that improved visual quality as well as reduced quality variation at the receiver are achievable at the same time by the proposed algorithm. Experimental results based on sources including image, video and HDTV demonstrate the effectiveness of the algorithm.

Finally, a robust image transmission scheme is proposed for packet erasure channels. The proposed scheme relies on the error resilience mechanisms provided by JPEG2000 for robust source decoding. Together with the proposed interleaving scheme, some erasures can be recovered.

7.2 Future Work

In Chapters 3, 4 and 6, a set of error resilience (ER) mechanisms from the JPEG2000 standard are employed. With further evolution of the standard towards wireless communications [61], more ER mechanisms may be proposed and adopted by the standard in the near future. This gives hopes to design more effective joint source/channel coding schemes.

In Chapter 5, joint coding is considered for multiple sources. In the HDTV case, the best performance is achieved when all the frames of the sequences are coded jointly. However, this practice requires the largest buffer size and delay. A sliding-window based algorithm could be applied to limit the number of frames being processed each time, thus alleviating the above problems, with some small loss in performance.

With the use of error resilience during source coding in some of the proposed schemes in the dissertation, there are certain redundancies embedded in the resulting source bitstreams. As pointed out in Chapter 1.1, this redundancy can also be utilized by the channel decoder. Some research effort has been made, such as [83], to exploit this redundancy by iterative decoding at the receiver. Possible improvement could be achieved by taking this into account for the joint rate allocation at the transmitter and designing well-matched channel codes.

REFERENCES

[1] C. E. Shannon, "A mathematical theory of communication", *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, July 1948.

[2] B. Hochwald and K. Zeger, "Tradeoff between source and channel coding", *IEEE Trans. Info. Theory*, vol. 43, pp. 1412–1424, Sept. 1997.

[3] M. J. Ruf and W. Modestino, "Operational rate-distortion performance for joint source and channel coding of images", *IEEE Trans. Image Processing*, vol. 8, pp. 305–320, Mar. 1999.

[4] G. Cheung and A. Zakhor, "Bit allocation for joint source/channel coding of scalable video", *IEEE Trans. Image Processing*, vol. 9, pp. 340–356, Mar. 2000.

[5] J. Hagenauer, "Source-controlled channel decoding", *IEEE Trans. Commun.*, vol. 43, pp. 2449–2457, Sept. 1995.

[6] B. Pettijohn, M. W. Hoffman, and K. Sayood, "Joint source/channel coding using arithmetic codes", *IEEE Trans. Commun.*, vol. 49, pp. 826–836, May 2001.

[7] J. Chou and K. Ramchandran, "Arithmetic coding-based continuous error detection for efficient ARQ-based image transmission", *IEEE Journal on Selected Areas in Commun.*, vol. 18, pp. 861–867, June 2000.

[8] T. Guionnet and C. Guillemot, "Soft decoding and synchronization of arithmetic and quasi-arithmetic codes: application to image transmission over noisy channels", *IEEE Trans. Image Processing*, vol. 12, pp. 1599–1609, Dec. 2003.

[9] D. Taubman, "JPEG2000: standard for interactive imaging", *Proceedings of the IEEE*, vol. 90, pp. 1336–1357, Aug. 2002.

[10] Y. Wang, S. Wenger, J. Wen, and A. K. Katsaggelos, "Error resilient video coding techniques", *IEEE Signal Processing Magazine*, pp. 61–82, July 2000.

[11] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: Turbo codes", in *Proc. Int'l Conf. Commun. (ICC)*, 1993, pp. 1064–1070.

[12] D. J. C. Mackay and R. M. Neal, "Good codes based on very sparse matrices", in *Cryptography and Coding, 5th IMA Conference, C. Boyd, Ed.*, 1995, vol. 1025, pp. 110–111.

[13] S. Benedetto and G. Montorsi, "Design of parallel concatenated convolutional codes", *IEEE Trans. Commun.*, vol. 44, pp. 591–600, May 1996.

[14] S. Chung, G. D. Forney, T. J. Richardson, and R. Urbanke, "On the design of low-density parity-check codes with 0.0045 dB of the Shannon limit", *IEEE Commun. Letters*, vol. 5, pp. 58–60, Feb. 2001.

[15] *JPEG2000 Image Coding System: Core Coding System*, ISO/IEC 15444-1, 2004.

[16] B. Kim, Z. Xiong, and W. A. Pearlman, "Low bit-rate scalable video coding with 3-D set partitioning in hierarchical trees (3-D SPIHT)", *IEEE Trans. Circuits System. Video Tech.*, vol. 10, pp. 1374–1387, Dec. 2000.

[17] S. Cho and W. A. Pearlman, "A full-featured, error-resilient, scalable wavelet video codec based on the set partitioning in hierarchical trees (SPIHT) algorithm", *IEEE Trans. Circuits System. Video Tech.*, vol. 12, pp. 157–171, Mar. 2002.

[18] D. S. Taubman and M. W. Marcellin, *JPEG2000: Image Compression Fundamentals, Practice and Standards*, Kluwer Academic Publishers, Massachusetts, 2002.

[19] D. Taubman, "High performance scalable image compression with EBCOT", *IEEE Trans. Image Processing*, vol. 9, pp. 1158–1170, July 2000.

[20] A. Secker and D. Taubman, "Lifting-based invertible Motion adaptive transform (LIMAT) framework for highly scalable video compression", *IEEE Trans. Image Processing*, vol. 12, pp. 1530–1542, Dec. 2003.

[21] A. Secker and D. Taubman, "Highly scalable video compression with scalable motion coding", *IEEE Trans. Image Processing*, vol. 13, pp. 1029–1041, Aug. 2004.

[22] *JPEG2000 Image Coding System: Part 3: Motion JPEG2000*, ISO/IEC 15444-3, 2004.

[23] D. Marpe, V. George, H. L. Cycon, and K. U. Barthel, "Performance evaluation of Motion-JPEG2000 in comparison with H.264/AVC operated in pure intra coding mode", in *Proc. SPIE: Wavelet Applications in Industrial Processing*, 2004, vol. 5266, pp. 129–137.

[24] S. Fößel, G. Föttinger, and J. Mohr, "Motion JPEG2000 for high quality video systems", *IEEE Trans. Consumer Electronics*, vol. 49, pp. 787–791, Nov. 2003.

[25] J. Hagenauer, "Rate-compatible punctured convolutional codes (RCPC codes) and their applications", *IEEE Trans. Commun.*, vol. 36, pp. 389–400, Apr. 1988.

[26] N. Seshadri and C. Sunderg, "List Viterbi decoding algorithm with applications", *IEEE Trans. Commun.*, vol. 42, pp. 313–323, Feb. 1994.

[27] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate", *IEEE Trans. Info. Theory*, pp. 284–287, Mar. 1974.

[28] D. N. Rowitch and L. B. Milstein, "On the performance of hybrid FEC/ARQ systems using rate compatible punctured turbo (RCPT) codes", *IEEE Trans. Commun.*, vol. 48, pp. 948–959, June 2000.

[29] O. Acikel and W. E. Ryan, "Punctured turbo-codes for BPSK/QPSK channels", *IEEE Trans. Commun.*, vol. 47, pp. 1315–1323, Sept. 1999.

[30] R. G. Gallager, *Low-Density Parity Check Codes*, MIT Press, Cambridge, MA, 1963.

[31] F. Kschischang, B. Frey, and H. Leoliger, "Factor graphs and the sum-product algorithm", *IEEE Trans. Info. Theory*, vol. 47, pp. 498–519, Feb. 2001.

[32] R. M. Tanner, "A recursive approach to low complexity codes", *IEEE Trans. Info. Theory*, vol. 27, pp. 533–547, Sept. 1981.

[33] J. K. Rogers and P. C. Cosman, "Wavelet zerotree image compression with packetization", *IEEE Signal Processing Letters*, vol. 5, pp. 105–107, May 1998.

[34] C. D. Creusere, "A new method of robust image compression based on the embedded zerotree wavelet algorithm", *IEEE Trans. Image Processing*, vol. 6, pp. 1436–1442, Oct. 1997.

[35] A. Mohr, E. Riskin, and R. Ladner, "Unequal loss protection: graceful degradation of image quality over packet erasure channels through forward error correction", *IEEE Journal on Selected Areas in Commun.*, pp. 819–828, June 2000.

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

[37] A. Said and W. A. Pearlman, "A new, fast, and efficient image codec based on set partitioning in hierarchical trees", *IEEE Trans. Circuits Syst. Video Tech.*, vol. 6, pp. 243–250, June 1996.

[38] P. G. Sherwood and K. Zeger, "Progressive image coding for noisy channels", *IEEE Signal Processing Letters*, vol. 4, pp. 189–191, July 1997.

[39] M. Zhao and A. N. Akansu, "Optimization of dynamic UEP schemes for embedded image sources in noisy channels", in *Proc. Int'l Conf. Image Processing*, 2000, vol. 1, pp. 383–386.

[40] V. Chande and N. Farvardin, "Progressive transmission of images over memoryless channels", *IEEE Journal on Selected Areas in Commun.*, vol. 18, pp. 127–131, June 2000.

[41] N. V. Boulgouris, N. Thomos, and M. G. Strintzis, "Image transmission using error-resilience wavelet coding and forward error correction", in *Proc. Int'l Conf. Image Processing*, 2002, vol. 3, pp. 549–552.

[42] P. G. Sherwood, X. Tian, and K. Zeger, "Channel code blocklength and rate optimization for progressive image transmission", in *Proc. Wireless Commun. Networking Conference (WCNC)*, 1999, vol. 2, pp. 978–982.

[43] V. Stankovic, R. Hamazoui, and D. Saupe, "Fast algorithm for optimal error protection of embedded wavelet codes", in *Proc. IEEE Workshop on Multimedia Signal Processing*, 2001, pp. 593–598.

[44] T. Chuu, Z. Liu, Z. Xiong, and X. Wu, "Joint UEP and layered source coding with application to transmission of JPEG-2000 coded images", in *Proc. Global Commun. Conference*, 2001, vol. 3, pp. 2036–2039.

[45] B. A. Banister, B. Belzer, and T. R. Fischer, "Robust image transmission using JPEG2000 and turbo-codes", *IEEE Signal Processing Letters*, vol. 9, pp. 117–119, Apr. 2002.

[46] R. Hamzaoui, V. Stankovic, and Z. Xiong, "Rate-based versus distortion-based optimal joint source-channel coding", in *Proc. Data Compression Conference*, 2002, pp. 63–72.

[47] C. Lan, K. R. Narayanan, and Z. Xiong, "Scalable image and video transmission using irregular repeat-accumulate codes with fast algorithm for optimal unequal error protection", *IEEE Trans. Commun.*, pp. 1092–1101, July 2004.

[48] T. Stockhammer, M. Hannuksela, and T. Wiegand, "H.264/AVC in wireless environments", *IEEE Trans. Circuits System. Video Tech.*, vol. 13, pp. 657–673, July 2003.

[49] I. Moccagatta, S. Soudagar, Jie Liang, and Homer Chen, "Error-resilience coding in JPEG-2000 and MPEG-4", *IEEE Journal on Selected Areas on Commun.*, vol. 18, pp. 899–914, June 2000.

[50] Y. Shoham and A. Gersho, "Efficient bit allocation for an arbitrary set of quantizers", *IEEE Trans. Acous. Sig. Proc.*, vol. 36, pp. 1445–1453, Sept. 1988.

[51] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, Athena Scientific, Belmont, Mass, 1995.

[52] D. Divsalar and F. Pollara, "Multiple turbo codes for deep-space communication", *JPL TDA Progressive Report*, pp. 42–121, May 1995.

[53] B. A. Banister, *Robust image and video transmission across heterogeneous networks with packet erasures and bit errors*, PhD thesis, Washington State University, 2002.

[54] P. G. Sherwood and K. Zeger, "Error protection for progressive image transmission over memoryless and fading channels", *IEEE Trans. Commun.*, vol. 46, pp. 1555–1559, Dec. 1998.

[55] L. Cao and C. Chen, "A novel product coding and recurrent alternate decoding scheme for image transmission over noisy channels", *IEEE Trans. Commun.*, vol. 9, pp. 1426–1431, Sept. 2003.

[56] D. G. Sachs, R. Anand, and K. Ramchandran, "Wireless image transmission using multiple-description based concatenated codes", in *Proc. Data Compression Conference*, Mar. 2000.

[57] A. Nosratinia, J. Lu, and B. Aazhang, "Source-channel rate allocation for progressive transmission of images", *IEEE Trans. Commun.*, vol. 51, pp. 186–196, Feb. 2003.

[58] B. S. Srinivas, R. E. Ladner, M. Azizoglu, and E. A. Riskin, "Progressive transmission of images using MAP detection over channels with memory", *IEEE Trans. Image Processing*, vol. 8, pp. 462–475, Apr. 1999.

[59] P. C. Cosman, J. K. Rogers, P. G. Sherwood, and K. Zeger, "Combined forward error control and packetized zerotree wavelet encoding for transmission of images over varying channels", *IEEE Trans. Image Processing*, vol. 9, pp. 982–993, June 2000.

[60] C. Lan, K. R. Narayanan, and Z. Xiong, "Source-optimized irregular repeat accumulate codes with inherent unequal error protection capabilities and their application to image transmission", in *Proc.* 37^{th} *Asilomar Conference on Signals, Systems and Computers*, Nov. 2003.

[61] *JPEG2000 Image Coding System: Part 11: Wireless JPEG2000*, ISO/IEC 15444-11, 2004.

[62] T. J. Richardson and R. Urbanke, "Design of capacity-approaching irregular low-density parity-check codes", *IEEE Trans. Info. Theory*, vol. 47, pp. 619–937, Feb. 2001.

[63] J. Hou, P. H. Siegel, and L. B. Milstein, "Performance analysis and code optimization of low density parity-check codes on Rayleigh fading channels", *IEEE Journal on Selected Areas in Commun.*, vol. 19, pp. 924–934, May 2001.

[64] Y. Kou, S. Lin, and M. P. C. Fossorier, "Low-density parity check codes based on finite geometries: a rediscovery and new results", *IEEE Trans. Info. Theory*, vol. 47, pp. 2711–2736, Nov. 2001.

[65] B. Vasic, E. M. Kurtas, and A. V. Kuznetsov, "Kirkman systems and their application in perpendicular magnetic recording", *IEEE Trans. Magnetic*, vol. 38, pp. 1705–1710, July 2002.

[66] B. Vasic and O. Milenkovic, "Combinatorial construction of low-density parity-check codes for iterative decoding", *IEEE Trans. Info. Theory*, vol. 50, pp. 1156—1176, June 2004.

[67] S. Chung, J. Richardson, and R. Urbanke, "Analysis of sum-product decoding of low-density parity-check codes using a gausssian approximation", *IEEE Trans. Info. Theory*, vol. 47, pp. 657–670, Feb. 2001.

[68] K. Price and R. Storn, "Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces", *Journal of Global Optimization*, pp. 341–359, 1997.

[69] A. Sholkrollahi and R. Storn, "Design of efficient erasure codes with differential evolution", in *Proc. Int. Symp. Info. Theory*, June 2000, pp. 5–5.

[70] M. Yang and W. E. Ryan, "Design of LDPC codes for two-state fading channel models", in *5th Int'l Symp. on Wireless Personal Multimedia Commun.*, Oct. 2002.

[71] V. Stankovic, R. Hamzaoui, and D. Saupe, "Fast algorithm for rate-based optimal error protection of embedded codes", *IEEE Trans. Commun.*, vol. 51, pp. 1788–1795, Nov. 2003.

[72] R. Zhang, S. Regunathan, and K. Rose, "Video coding with optimal inter/intra-mode switching for packet loss resilience", *IEEE Journal on Selected Areas in Commun.*, vol. 18, pp. 966–976, June 2000.

[73] Z. He, J. Cai, and C. Chen, "Joint source channel rate-distortion analysis for adaptive mode selection and rate control in wireless video coding", *IEEE Trans. Circuits and Systems for Video Tech.*, vol. 12, pp. 511–523, June 2002.

[74] M. Bystrom and T. Stockhammer, "Dependent source and channel rate allocation for video transmission", *IEEE Trans. Wireless Commun.*, vol. 3, pp. 258–268, Jan. 2004.

[75] L. Wang and A. Vincent, "Bit allocation and constraints for joint coding of multiple video programs", *IEEE Trans. Circuits and Systems for Video Tech.*, vol. 9, pp. 949–959, Sept. 1999.

[76] J. C. Dagher, A. Bilgin, and M. W. Marcellin, "Resource constrained rate control for Motion JPEG2000", *IEEE Trans. Image Processing*, pp. 1522–1529, Dec. 2003.

[77] M. Balakrishnan and R. Cohen, "Global optimization of multiplexed video encoders", in *Proc. Int'l Conf. Image Processing*, 1997, vol. 1, pp. 377–380.

[78] B. A. Banister, B. Belzer, and T. R. Fischer, "Robust video transmission over binary symmetric channels with packet erasures", in *Proc. Data Compression Conference*, Mar. 2002.

[79] J. M. Shapiro, "Embedded image coding using zerotrees of wavelet coefficients", *IEEE Trans. Signal Processing*, vol. 41, pp. 3445–3462, Dec. 1993.

[80] X. Wu, S. Cheng, and Z. Xiong, "On packetization of embedded multimedia bitstreams", *IEEE Trans. Multimedia*, vol. 3, pp. 132–140, Mar. 2001.

[81] V. Stankovic, R. Hamzaoui, and Z. Xiong, "Packet loss protection of embedded data with fast local search", in *Proc. Intl. Conf. Image Processing*, 2002, vol. 2, pp. 165–168.

[82] J. Thie and D. Taubman, "Optimal erasure protection assignment for scalable compressed data with small channel packets and short channel codewords", *EURASIP Journal on Applied Signal Processing – Special issue on Multimedia over IP and Wireless Networks*, pp. 207–219, Feb. 2004.

[83] L. Pu, Z. Wu, A. Bilgin, M. Marcellin, and B. Vasic, "Iterative joint source-channel decoding for JPEG2000", in *Asilomar Conference on Signals, Systems, and Computers*, Nov. 2003.