Title
Bandwidth and rate allocation tradeoffs of source : channel coding, packetization and modulation in unequally protected multimedia communication systems

Permalink
https://escholarship.org/uc/item/97c3938x

Author
Arslan, Suayb S.

Publication Date
2012

Peer reviewed|Thesis/dissertation
UNIVERSITY OF CALIFORNIA, SAN DIEGO

Bandwidth and Rate Allocation Tradeoffs of Source–Channel Coding, Packetization and Modulation in Unequally Protected Multimedia Communication Systems

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

in

Electrical Engineering (Communication Theory and Systems)

by

Suayb S. Arslan

Committee in charge:
Professor Pamela C. Cosman, Chair
Professor Laurence B. Milstein, Co-Chair
Professor Bruce K. Driver
Professor William S. Hodgkiss
Professor Bhaskar D. Rao

2012
The dissertation of Suayb S. Arslan is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

__________________________________________

__________________________________________

__________________________________________

Co-Chair

__________________________________________

Chair

University of California, San Diego

2012
DEDICATION

To my beloved parents.
## TABLE OF CONTENTS

Signature Page ................................... iii  
Dedication ...................................... iv  
Table of Contents .................................. v  
List of Figures ................................... viii  
List of Tables .................................... xi  
Acknowledgements .................................. xii  
Vita ............................................ xv  
Abstract of the Dissertation ............................ xvi  

**Chapter 1**  
Introduction ...................................... 1  
1.1 Basic Concepts .................................. 1  
1.1.1 Source Coding .................................. 1  
1.1.2 Channel Coding ................................. 5  
1.1.3 Joint Source-Channel Coding ................. 7  
1.1.4 Digital Modulation and Hierarchy .......... 8  
1.2 Motivation for the Research .................. 10  
1.3 Problem Definition .............................. 12  
1.4 Thesis Outline ................................. 13  

**Chapter 2**  
Coded Hierarchical Modulation for Wireless Progressive Image Transmission ............................ 16  
2.1 Related Work and Contributions ............. 17  
2.2 The System Model ............................... 19  
2.3 Bits-to-Symbol Assignment Methodologies .... 22  
2.4 Performance Analysis ........................... 24  
2.4.1 Packet Error Rate Approximation .......... 27  
2.4.2 Construction of the Optimization Problem . 29  
2.4.3 Optimization of Hierarchical Parameter Set: $\alpha$ . 31  
2.4.4 A Lower Bound on the Performance Improvement of the Proposed System ................... 32  
2.4.5 Numerical Results for Memoryless Channels . 36  
2.4.6 Computational Complexity .................... 44  
2.5 Proposed Scheme under Frequency Selective Channels .......................... 45  
2.5.1 Channel Model ............................... 46  
2.5.2 Numerical Results for a Frequency Selective Channel 49
<table>
<thead>
<tr>
<th>Chapter 3</th>
<th>Concatenated Block Codes for Unequal Error Protection of Embedded Bit Streams</th>
<th>58</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Related Work and Contributions</td>
<td>60</td>
</tr>
<tr>
<td>3.2</td>
<td>Proposed Formatting and System Model</td>
<td>63</td>
</tr>
<tr>
<td>3.3</td>
<td>Algorithm Design and Optimization</td>
<td>65</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Optimization</td>
<td>68</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Concatenated Coding with LDPC Codes</td>
<td>73</td>
</tr>
<tr>
<td>3.3.3</td>
<td>Complexity of the Proposed Scheme</td>
<td>75</td>
</tr>
<tr>
<td>3.4</td>
<td>Numerical Results and Discussions</td>
<td>76</td>
</tr>
<tr>
<td>3.4.1</td>
<td>Simulations with RCPC Codes</td>
<td>76</td>
</tr>
<tr>
<td>3.4.2</td>
<td>Simulations with RC-LDPC Codes</td>
<td>90</td>
</tr>
<tr>
<td>3.4.3</td>
<td>Discussion on the Advantages of the Proposed Transmission Scheme</td>
<td>91</td>
</tr>
<tr>
<td>3.5</td>
<td>Summary and Conclusions</td>
<td>92</td>
</tr>
</tbody>
</table>

| Chapter 4  | Generalized Unequal Error Protection Fountain Codes for Progressive Data Transm. | 95 |
| 4.1       | Related Work and Contributions                                                  | 96 |
| 4.2       | Background Overview                                                             | 99 |
| 4.2.1     | Progressive Source Coding                                                       | 100 |
| 4.2.2     | Digital Fountain Codes                                                          | 101 |
| 4.2.3     | UEP DF Code Designs                                                             | 105 |
| 4.3       | UEP Generalized LT (UEP GLT) Coding                                             | 107 |
| 4.3.1     | Generalization of “weighted approach”                                           | 108 |
| 4.3.2     | Generalization of EWF Codes                                                     | 111 |
| 4.3.3     | Unequal Iteration Time Property                                                 | 112 |
| 4.3.4     | Optimization Problem                                                            | 114 |
| 4.4       | Progressive Source Transmission System and Design Parameter Selection            | 116 |
| 4.4.1     | Progressive Source Transmission System Description                              | 116 |
| 4.4.2     | Comparison with the “weighted approach”                                         | 119 |
| 4.4.3     | Comparison with UEP EWF Codes                                                   | 120 |
| 4.5       | Numerical Results                                                               | 121 |
| 4.5.1     | Comparisons with the “weighted approach”                                        | 122 |
| 4.5.2     | Comparisons with UEP EWF Codes                                                  | 131 |
| 4.6       | Conclusions                                                                     | 134 |

Appendix A  | Hard Decision Upper Bounds for Coded $M$-Ary Hierarchical Modulation             | 137 |
| A.1       | Hard Decision Upper Bound                                                       | 138 |
| A.1.1     | $n$ odd                                                                        | 141 |
## A.1.2 \( n \) even ........................................ 142  
## A.2 Simulations ........................................ 144

### Appendix B Lower Bound on the Performance Improvement of the Proposed System in Chapter 2  
- B.1 \( N \) is divisible by four ................................ 150  
- B.2 \( N \) is not divisible by four ............................. 152

### Bibliography .............................................. 153
<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.1</td>
<td>Progressive coding and image refinement.</td>
</tr>
<tr>
<td>1.2</td>
<td>Modulated symbols in a signal constellation representation. Hierarchical modulation is obtained by moving the symbols in both in phase and quadrature axis directions.</td>
</tr>
<tr>
<td>1.3</td>
<td>A typical communication system.</td>
</tr>
<tr>
<td>2.1</td>
<td>Baseline system model with related block diagrams. Explanation for each block is given in the text. System is adaptive according to channel conditions.</td>
</tr>
<tr>
<td>2.2</td>
<td>Two different assignments of packetized bits to symbols (out of $\prod_{i=1}^{N-1} \binom{N-2i}{2}$ different possible ways) in a progressively encoded source. $P_i$: $i$th chunk of information.</td>
</tr>
<tr>
<td>2.3</td>
<td>Proposed NLSR versus simulation results and upper bounds derived in this paper for both HP and LP priority classes using 4PAM and an RCPC code rate $\beta = 1/2$.</td>
</tr>
<tr>
<td>2.4</td>
<td>Proposed NLSR versus simulation results and upper bounds derived in this paper for both HP and LP priority classes using 16QAM and an RCPC code rate $\beta = 1/2$.</td>
</tr>
<tr>
<td>2.5</td>
<td>Functional flow diagram of the proposed optimization scheme.</td>
</tr>
<tr>
<td>2.6</td>
<td>HP and LP BER performances using uncoded and coded ($r = 1/2$) conventional 4PAM/16QAM for AWGN channel.</td>
</tr>
<tr>
<td>2.7</td>
<td>HP and LP BER performances using uncoded and coded ($r = 1/3$) conventional 4PAM/16QAM for AWGN channel.</td>
</tr>
<tr>
<td>2.8</td>
<td>Performance of different systems for an independent flat Rayleigh fading channel. JSCC EEP is also drawn for comparison. This figure uses the approximate PER expressions by nonlinear regression for both Lena and Barbara images.</td>
</tr>
<tr>
<td>2.9</td>
<td>Optimal hierarchical parameters ($\alpha^*$) for different channel code rate pairs at SNR=7dB for an AWGN channel. It also shows the corresponding calculated values of MSE. (2/3,2/5) is the optimal code rate pair.</td>
</tr>
<tr>
<td>2.10</td>
<td>PER assignment among the packets of $\alpha$-adaptive system can provide hundreds of different UEP levels whereas UEP achieved by packetization provides only two layers of unequal PERs.</td>
</tr>
<tr>
<td>2.11</td>
<td>Proposed system using H-4PAM, H-16QAM constellations and the corresponding upper bounds for $PER_l$ given in (2.7) in formulation of the cost function instead of using the approximation $\hat{PER}_l$, $l = 1, 2, \ldots, N$ and the optimal values in (A.1).</td>
</tr>
<tr>
<td>2.12</td>
<td>Performance upper bound is calculated for different transmission rates (in bpp) and packet sizes, illustrating the effect of transmission rate and packet size on the proposed system.</td>
</tr>
</tbody>
</table>
Figure 2.13: A sample energy decay profile. ....................... 49
Figure 2.14: Effect of the equalization for various frequency selective channels \( h_1, h_2 \) and \( h_3 \). .......................... 51
Figure 2.15: CDF of PSNR for the frequency selective channel \( h_3 \) using equalization under frequency selective channel without link breakage. 53
Figure 2.16: CDF of PSNR for the frequency selective channel \( h_3 \) using equalization under frequency selective channel with link breakage. 53
Figure 2.17: Performance of different systems for a frequency selective channel with link breaks using \( h_3 \) and \( p_1 = 0.81 \) and \( p_2 = 0.33 \). JSCC EEP is also drawn for comparison. The results correspond to the Lena image. 55

Figure 3.1: Two methods of packet formatting for progressive source transmission. .................................................. 62
Figure 3.2: Proposed formatting consists of \( M \) stages of encoding as shown above. ............................................... 63
Figure 3.3: Each payload in the embedded bit stream is exposed to a BSC with different crossover probabilities. ...................... 66
Figure 3.4: Functional flow diagram of the proposed optimization and transmission scheme. ....................................... 70
Figure 3.5: Distortion versus number of source bits illustrating \( \Delta_1, \Delta_2 \ldots \Delta_l \) as functions of \( b_1, b_2, \ldots, b_l \). .......................... 72
Figure 3.6: For several different sets \( \mathcal{R}, \epsilon_0 \) and \( r_{tr} \), \( D_2 \left( \left\{ b_1, r_2 \widehat{B} - \frac{b_1}{r_1} \right\}, \mathcal{R} \right) \) is plotted for a SPIHT source encoder. As can be seen, it is a concave function of \( b_1 \) and has a single maximum point. .... 74
Figure 3.7: The effect of interleaving with different size of the random permutation tables. ............................................. 78
Figure 3.8: Different systems are compared over a BSC with BER=0.01 and BER=0.05 using 512 \times 512 Lena image. .............. 82
Figure 3.9: Different systems are compared over a BSC with BER=0.01 and BER=0.05 using 512 \times 512 Goldhill image. .......... 83

Figure 4.1: Belief propagation algorithm for LT decoding. \( k=4, n=5 \). Squares (□) are check nodes (coded symbols) and circles (○) represent variable nodes (information symbols). .......................... 104
Figure 4.2: Non-uniform selection of variable nodes. Black squares represent coded symbols (check nodes) and circles represent information symbols (variable nodes). White and gray circles represent two different priority classes. ........................................ 110
Figure 4.3: The block diagram of the progressive transmission scheme. .......................... 116
Figure 4.4: The demultiplexing of the source bit stream. ......................... 117
Figure 4.5: Each information block is of size $\lfloor B/k \rfloor$ information bits. The $z$th $k$-bit segment is generated by collecting the $z$th bit of each information block. After forming each $k$-bit segment, a realization of the LT coding is applied to each $k$-bit segment.

Figure 4.6: Performance comparisons using Lena and $k = 100$.

Figure 4.7: Performance comparisons using Goldhill and $k = 100$.

Figure 4.8: Performance comparisons using Lena and $k = 1000$.

Figure 4.9: Performance comparisons using Lena and a overhead of $\epsilon = 0.4$ for a range of $B$.

Figure 4.10: UIT Performance comparisons using Lena for $n = 150$, $\alpha_1 = 0.3$ and $B = 50000$ bits.

Figure 4.11: Performance comparisons with EWF codes using Lena with $k = 100$.

Figure 4.12: Performance comparisons with EWF codes using Goldhill with $k = 100$.

Figure A.1: Double sums in (A.8) and (A.9) are equivalent.

Figure A.2: Conventional 4PAM ($\alpha = 3$) coded with NASA standard code.

Figure A.3: Hierarchical 16QAM coded with RCPC 8/9 code with $M=6$ from [30]. Bold curves are the upper bounds. U.B.: Upper Bound.

Figure A.4: Two concentric 2/4 PSKs coded with several RCPC codes with $M=4$ from [30].

Figure A.5: RCPC 2/3 code used in conjunction with H-16QAM under flat Rayleigh fading channel.
LIST OF TABLES

Table 2.1: Some sample values of $A^*$, $B^*$ that are used in our simulations for a set of RCPC code rates ........................................... 28
Table 2.2: Simulation parameters ($T$: Symbol duration) ............................... 50

Table 3.1: Decoding thresholds for the semi-random rate compatible punctured LDPC codes. .................................................. 75
Table 3.2: Optimal allocation: $R^*$ for $M$-packet transmission. Optimal parameter ($M^*$) is shown in bold. ............................ 85
Table 3.3: Number of possibilities with and without the first condition. We obtain complexity reductions with increasing $M$. For example when $M = 5$, it is enough to search only $1/62$ of all possibilities. .......................... 86
Table 3.4: Number of possibilities with two conditions when $\epsilon_0 = 0.01$, $r_{tr} = 0.7$bpp. We obtain more complexity reductions with increasing $M$. For example when $M = 5$, it is enough to search only $1/3055$ of all possibilities. .............................. 86
Table 3.5: PSNR in dB for $512 \times 512$ images transmitted over BSC at various transmission rates. Ordinary Viterbi decoder is replaced with an LVA. .................................................................................. 87
Table 3.6: PSNR in dB for $512 \times 512$ Lena and Goldhill images (JPEG2000). Bold denotes the winning scheme. ConRCPC uses an RCPC code family with ordinary Viterbi decoding and ConLDPC uses an RC-LDPC code family with a belief propagation algorithm. .................. 88

Table 4.1: Number of coded symbols which need to be sent for a PSNR of 31dB using Lena. ......................................................... 124
Table 4.2: Optimal parameters of various UEP designs shown in Figs. 4.6, 4.8. ..................................................................................... 125
Table 4.3: Parameters of the UEP GLT and Comparison of UEP DF designs. 130
Table 4.4: Optimal parameters of various UEP designs for various $k$ and $\epsilon$. ................................. 134

Table A.1: Parameters used in the evaluation of the bound for HP bits for different channel and constellation assumptions. ................. 144
Table A.2: Parameters of the codes. GP: Generator polynomial. .................................... 145
ACKNOWLEDGEMENTS

First and foremost, I would like to thank my advisors. It is nearly impossible for me to overstate my gratitude to my Ph.D. supervisors Dr. Pamela Cosman and Dr. Laurence Milstein for their enthusiasm, inspiration and ever-lasting guidance over the past several years. It was an honor to work with them and receive their advice. I am impressed not only for their dedication to research but also for their incessant love for teaching. They kindly take time to answer questions and share tips for making their classes attractive. I am grateful for their mentorship, direction and leadership.

I wish to thank the members of my thesis committee, Prof. Baskar Rao, Prof. Bruce Driver and Prof. William Hodgkiss, for their time, assistance and advice. In particular, I am thankful to Prof. Bruce Driver for discussions on probability theory and to Prof. Kenneth Zeger for making the source coding class both fun and a pleasant experience. I also wish to thank Prof. Kadri Ozcaldiran and Prof. Burak Gurel from Bogazici University for numerous conversations and discussions in my pursuit of the Ph.D..

I gratefully acknowledge the funding sources that made my Ph.D. work possible. I was funded by the Scientific and Technological Research Council of Turkey (TUBITAK) for my first year and I was privileged to receive funding in part by Intel Inc. for years 2 and 3. During that time, various funding agencies supported my research, including the Center for Wireless Communications at the University
of California, San Diego, and the University of California Discovery Grant Program of the State of California. The last 2 years of my research work was largely funded by LG Electronics Inc. and by the National Science Foundation (NSF) under grant CCF-09IS727.

During my Ph.D. studies, I have been involved with various collaborators across the industry in diverse yet similar research disciplines. My sincere thanks goes to Dr. Fatih Porikli from Mitsubishi Electric Research Laboratory (MERL) for putting so much trust in me and giving me growing knowledge about the broad class of imaging techniques. I am also indebted to Turguy Goker who has provided me a chance to develop advanced detection algorithms for high capacity tape drives. His friendly attitude deserves mention. I am also indebted to many student colleagues, particularly people in the wireless communications laboratory at UCSD, for providing a stimulating and fun environment in which to learn and grow.

Last but not least, I would like to thank my family for all their love and encouragement—especially my parents who raised me with a love of science and supported me in all my pursuits.

pp. 4299–4313, July 2011, and in S. S. Arslan, P. C. Cosman and L. B. Milstein, “On Hard Decision Upper Bounds for Coded M-Ary Hierarchical Modulation,” in Proceedings of IEEE Conference on Information Sciences and Systems (CISS), Baltimore, MD, USA, March 2011. I was the primary author of all these publications; the coauthors were Prof. Cosman and Prof. Milstein who directed and supervised the research which forms the basis for Chapter 2.

Chapter 3 of this dissertation, in full, is a reprint of the material as it appears in S. S. Arslan, P. C. Cosman, and L. B. Milstein, “Concatenated Block Codes for Unequal Error Protection of Embedded Bit Streams,” IEEE Transactions on Image Processing, vol. 21, no. 3, pp. 1111–1122, March 2011. I was the primary author of this publication. My coauthors Prof. Cosman and Prof. Milstein directed and supervised the research which forms the basis for Chapter 3.

Chapter 4 of this dissertation, in full, is a reprint of the material that has been submitted for publication in S. S. Arslan, P. C. Cosman, and L. B. Milstein, “Generalized Unequal Error Protection LT codes for Progressive Transmission,” in IEEE Transactions on Image Processing, Jan. 2012. I was the primary author of this publication and the coauthors Prof. Cosman and Prof. Milstein directed and supervised the research which forms the basis for Chapter 4.
VITA

2006       B. S. in Electrical Engineering (Communication Theory and Systems) cum laude, Bogazici University, Istanbul, Turkey.

2006-2012  Graduate Research Assistant, Department of Electrical and Computer Engineering, University of California, San Diego, La Jolla, CA.

2009       M. S. in Electrical Engineering (Communication Theory and Systems), University of California, San Diego, La Jolla, CA.

Summer 2009 Research Intern, Mitsubishi Electric and Research Laboratory (MERL), Cambridge, MA.

Summer 2011 Research Intern, Quantum Corporation, Irvine, CA.

2012       Ph. D. in Electrical Engineering (Communication Theory and Systems), University of California, San Diego, La Jolla, CA.

PUBLICATIONS


ABSTRACT OF THE DISSERTATION

Bandwidth and Rate Allocation Tradeoffs of Source–Channel Coding, Packetization and Modulation in Unequally Protected Multimedia Communication Systems

by

Suayb S. Arslan

Doctor of Philosophy in Electrical Engineering
(Communication Theory and Systems)

University of California, San Diego, 2012

Professor Pamela C. Cosman, Chair
Professor Laurence B. Milstein, Co-Chair

A conventional approach to the design of wireless multimedia communications is the layered approach, in which the network layers function independent of each other. This kind of layered approach is inspired partly by Shannon’s separation theorem in which the optimization of each block is equivalent to optimization of the overall source-channel coding operation. However, the separation theorem is valid only in a point-to-point communication scenario in the case of asymptotically long block lengths of data and assumes huge amounts of processing power and delay.
Therefore, current practical communication systems strive to jointly design building blocks of a multimedia system for better performance. The focus of this dissertation is therefore to present various joint designs for different channel models and systems, although limited by physical constraints such as bandwidth, power and complexity.

First, a robust coded scheme for progressive multimedia transmission is considered for an additive white Gaussian noise channel, a Rayleigh fading channel, and a frequency selective channel using in combination different unequal protection methods. We investigate the judicious use of the limited bandwidth through the combination and optimization of a progressive source coder, a rate compatible punctured convolutional code and a hierarchical modulation.

Next, we investigate a novel packet formatting scheme for progressive sources using interleavers and various channel codes. The source coder is combined with a concatenated block coding mechanism to produce a robust transmission system for embedded bit streams. The objective is to create embedded codewords such that, for a particular information block, the necessary protection is obtained via multiple channel codings, contrary to the conventional methods which use a single code rate per information block. We show that near-capacity performance can be achieved using the proposed scheme in conjunction with low density parity check codes in a binary symmetric channel scenario.

We finally focus on coding strategies for multimedia where the channel state information is missing. A generalized Digital Fountain (DF) code is proposed to provide efficient universal forward error correction solution for lossy packet networks.
with increased unequal error protection and unequal recovery time properties. We propose a progressive source transmission system using this generalized code design. We apply the generalized DF code to a progressive source and show that it has better unequal protection and recovery time properties than other published results.
Chapter 1

Introduction

1.1 Basic Concepts

1.1.1 Source Coding

Multimedia sources generally require large memory as well as a wide bandwidth for reliable communication. In many multimedia applications for instance, the transmission of the original source might cause prolonged transmission and waiting times. Bandwidth is typically a scarce resource for communication channels in general, and wireless channels in particular [1]. For an optimal system design, every single communication resource should be utilized efficiently. To achieve this objective, one of the obvious solutions is to represent the source using as few symbols as possible so that the communication system will not need to allocate large resources for transmission. Compression of the data is accomplished by exploiting the
redundancies within the source. This operation is known as “source coding” in the literature [2].

Source coding can be classified into two major categories: Lossy and Lossless. “Lossless compression” allows an exact copy of the original data to be reconstructed from the compressed version. In this case, the decoder can reconstruct the source perfectly. This type of source coding however, does not provide much compression for use with a multimedia source. It is the preferred coding strategy however, for data compression scenarios (ex: text, executables) in which any loss of data is considered intolerable. On the other hand, lossy compression is a data compression methodology which allows some loss of the original data, in order to obtain better compression efficiency. Since some parts of the images or videos cannot be perceived by the human eye, “lossy compression” is the desired solution for efficient representation of multimedia sources. The theoretical foundations for lossy compression are given by a major branch of information theory called Rate-Distortion (R–D) theory [3].

There are plenty of compression algorithms developed to date. Although there is no generic compression algorithm that works well for all sources, many practical compression algorithms have been developed that give good compression results. The very well known JPEG compression algorithm for images comes with numerous selection features that give satisfying results. A higher quality compression can be achieved using more advanced transform coders such as Set-Partitioning-In-Hierarchical-Trees (SPIHT). A popular compression algorithm for video is the H.264 Advanced Video Coder (AVC) which generates high-quality compressed video. It
is the current state-of-art ubiquitous video compression algorithm used over the internet as well as in advanced storage media.

**Progressive Source Coding**

*Progressiveness* is a term used to describe a specific property of some of the state-of-art image coding mechanisms. Progressive or embedded source encoders have the attractive feature that the coding bitstream can be truncated at any point and still be decoded to a perceptible image. As more and more bits are reliably received, they are used to refine the decoded source at any decoding instant. Progressiveness employs different names within the video encoding terminology (such as *Scalability*: Scalable Video Encoding (SVC)). In many contexts however, the terms progressiveness and scalability are used interchangeably.

In an embedded bit stream, the sensitivity of each bit location is different i.e., the bits are ordered in importance. For example, a binary representation of the decimal number (a pixel value) 180 (10110100) is an embedded bit stream in the sense that the most significant bit is the left most bit 1 and gives a coarse description (10000000 ↔ 128) of the number 180. Note that a bit error in the third and sixth bit locations produce $32^2$ and $4^2$ squared errors, respectively. Thus, the third bit is more sensitive and therefore more important than the sixth bit.

An important application of embedded coders can be found in internet multimedia browsing, multimedia database, digital camera and streaming videos over the web, etc. In 1993, Shapiro [4] proposed a progressive source coder called the
Embedded Zero Wavelet (EZW) encoder that produces an embedded bit stream. It encodes the image from the most significant bitplane to the least significant bitplane, and within each bitplane, it used a zerotree structure to group the coefficients of the insignificant bits. Said and Pearlman improved EZW and developed the scheme of SPIHT [5]. Later, SPIHT was successfully applied to video sources and referred as 3D-SPIHT [6].

**Figure 1.1**: Progressive coding and image refinement.

In this dissertation, the source images are usually encoded using the lossy image compression algorithms SPIHT or JPEG2000. They are embedded source coders that produce a progressive bit stream, i.e., every successive bit contributes a certain amount of refinement after decoding. An example is shown in Fig. 1.1 using the image *Lena*. As can be seen, 4% of the total bit stream gives a coarse description of the source at around 22.55dB Peak Signal-to-Noise Ratio (PSNR), where PSNR is a fidelity criterion. As more and more source bits are used to decode the image i.e., 20%, 40% and 100% of the whole bit stream, the quality of the image is refined,
giving 27.17dB, 29.81dB and 33.68dB PSNRs, respectively. This partial decoding property comes with the cost that the usefulness of correctly received bits depends on the reliable reception of the previous bits. This makes progressive bit streams vulnerable to noisy channel effects, i.e., any error renders the remaining information bits useless, even if they are reliably received.

1.1.2 Channel Coding

Channel coding is the name for the operation of adding extra redundancy to source symbols in order to achieve a form of protection for transmitted symbols. The receivers exploit this redundancy to correct errors due to channel noise.

Perhaps the simplest error correcting code is the repetition code. The transmitter sends the data bit \( m \) times. Because the channel error probability is always less than 0.5, it is probable that more of the bits will be correct. Simple majority voting of the received bits determines the transmitted bit more accurately than sending the bit alone. A rate \( 1/m \) binary repetition code can then correct the bit error as long as the total number of channel errors do not exceed \( \lfloor m/2 \rfloor \). As can be seen, in order to send a bit with 0 error probability, the transmitter should repeat that bit indefinitely, which makes repetition codes inefficient.

Claude E. Shannon proved that there is a certain limit, called channel capacity, to the information rate of reliable transmission over a noisy channel. His proof however based on probability theory and was not constructive. In other words, he did not explicitly present a good physical channel code that has a vanishing prob-
ability of decoding error with growing block lengths as well as practically efficient encoding and decoding structures. Since then, numerous efforts have been made to find good codes that achieve the capacity with reasonable encoding and decoding complexity. In the past, there have been successful channel code designs, such as linear block and convolutional codes, that give satisfactory performance. It was not until early 90s, with the advent of turbo codes, that near-capacity encoding was seen to be practical. The novelty of turbo codes lies in using pseudorandom interleavers in the encoding algorithm and in the efficient iterative decoding algorithm. Soon after the introduction of turbo codes, Low Density Parity Check (LDPC) codes were introduced which have excellent performance, comparable to and often exceeding that of the turbo codes. LDPC codes and their iterative decoding algorithms have been widely adopted and analysed for their near-capacity performance.

Another important class of block codes is known as Reed-Solomon (RS) codes have found important applications from deep-space communication to consumer electronics. Since the minimum distance has the maximum value possible for a linear code of size \((n, k)\), such codes are referred as optimal. However, such codes are practical only for small \(n\) and \(k\) in addition to their complex decoding architectures. Alternative codes such as Digital Fountain (DF) codes have been introduced to alleviate some of the disadvantages of RS codes. DF codes have significantly less complex encoding and decoding structures than the more traditional Reed Solomon (RS) codes and exhibit high erasure correction performance for large block lengths [7].
1.1.3 Joint Source-Channel Coding

In Shannon’s source-channel coding theorem, both source compression and channel coding are performed separately. This independent operation between source and channel coders is the reason for this theorem to be known as the separation theorem. According to the separation theorem, optimization of each coding block to reach Shannon’s bounds (meaning entropy for the source coding, and capacity for the channel) is equivalent to optimization of the overall source-channel coding operation. Thus, the source coding block aims at the elimination of all the redundancy embedded in the source, whereas the channel coding aims at adding the necessary redundancy to enable error-free communication over noisy channels. Shannon’s results however are obtained under the implicit assumption of stationary and ergodic channels. Also, the separation theorem is valid only in a point-to-point communication scenario in the case of asymptotically long block lengths of data, assuming huge amounts of processing power, complexity and delay. In practical applications, the conditions under which the Shannon’s separation theorem holds are not satisfied. In the presence of practical constraints such as complexity and processing delay, the optimal system designs are generally not the simple concatenation of system sub-components optimized under physical design constraints. In such cases, combined design or Joint Source-Channel Coding (JSCC) can provide greater flexibility and yield better performance over separately designed and optimized source and channel coders.
Practical source coders are often slightly suboptimal and cannot eliminate all the redundancy in a given source. An effective overall system can always be constructed by cascading a good source coder with a good error correction coding mechanism. As argued, such a separation of coding functions may not provide the best overall performance, but there is also no fundamental obstacle preventing separated system designs from being near optimal. In fact, most of the practical communication systems today are designed and based on this separation concept. For example in multimedia transmission systems, current state-of-art source coders are aimed at eliminating the redundancy as much as possible, and the source bit streams are encoded with asymptotically optimal channel codes to combat channel noise.

1.1.4 Digital Modulation and Hierarchy

Modulation is the process of facilitating the transfer of information over a physical medium. A bandlimited range of frequencies that comprise the baseband message is translated to a higher range of frequencies to better match the frequency characteristics of the channel. This involves a variation of some parameter of a carrier (such as amplitude or phase) in accordance with a modulating wave (message). If the information consists of a train of digital pulses, the variation of the parameters is called “keying”.

Hierarchical Modulation

In various communication scenarios, such as multimedia transmissions or interactive systems, different sections of the source bit stream have different sensitivities against channel errors. Providing the same level of protection for the entire bit stream is power inefficient and therefore waste of resources. An example is shown in Section 1.1.2, where it is more efficient to protect just the most significant bit rather than the rest.

Fig. 1.2 illustrates 4 pulse amplitude modulation (4PAM), Binary Phase Shift Keying (BPSK) and a hierarchical constellation, where the hierarchy is established by adjusting the distances of the symbol points in the constellation through the use of a hierarchical parameter. As shown in Fig. 1.2, the ratio of the distances of the symbols to the origin on one side of the constellation can be changed while...
the average symbol energy is kept constant. In going from one constellation to another, High Priority (HP) bits and Low Priority (LP) bits will have different error probabilities. Therefore, a high priority (more sensitive) bit will have better protection than the low priority (less sensitive) bit when modulated symbols are transmitted over noisy channels.

1.2 Motivation for the Research

As the next century of technology goes wireless, multimedia transmission over wireless links is attracting a tremendous amount of interest in the research community. The current research activities include designing flexible transmission schemes and related hardware implementations. In general, multimedia transmissions call for extremely demanding communication systems. This is usually due to the enormous amounts of information to process and transmit. With the ubiquitous use of High Definition (HD) multimedia, which is one of the promising services of 3G and beyond technologies in the market, the computational load both on the user and the server sides has dramatically increased. Current wireless standards are constrained by the limited communication resources. the increased information load requires greater capacity for reliable communication. Thus, the transmission of multimedia over such bandlimited wireless networks becomes a challenging problem, in that one needs to maintain an acceptable quality of the performance at all times.

In traditional non-progressive image coding, the receivers are either able to
reconstruct the multimedia source at high quality, or to receive no reliable information for any source reconstruction. Given the random nature of the wireless channels, this property leads to excessive quality fluctuations, which result in increased user dissatisfaction. Progressive image and video coding techniques alleviate such undesirable effects of wireless links and allow graceful degradations of the source at the expense of small coding inefficiency. Any error due to the channel however may render the whole embedded bitstream useless. Therefore, the main problem is the joint design of typical communication system components giving unequal error protection for sources that exhibit this progressive property. This general problem involves a wide range of small subproblems being addressed, such as

- good choice of packetization of source bits,
- the redundancy allocation tradeoff between source and channel coding,
- the objective of picking the right set of channel codes for a given source coder, packetization and channel model,
- unequal error protection using a joint design of source-channel-modulation coding,
- generalization of coding strategies for progressive transmission.

It is hard to find a solution that will optimize all of these objectives at the same time. Previous research in the literature proposed solutions to some of the subproblems mentioned above under specific assumptions and channel models. Those
studies are usually restricted in several aspects, for example, some of these studies consider only the redundancy allocation problem in a joint source-channel coding setting. Some of the following system-related questions constitute the main motivation for this dissertation: (a) investigation of the relationship of the three system subcomponents: source coding, channel coding and modulation in an unequal protection setting (b) the relationship between packetization paradigms, source and channel coding in a progressive source transmission scenario and (c) more generalized coding designs for progressive sources under various channel models.

1.3 Problem Definition

![Diagram of a typical communication system](image)

**Figure 1.3:** A typical communication system.

A typical communication system diagram is shown in Fig. 1.3. Let us assume $r_s$ (in bits-per-source symbol) and $r_c$ (in source symbol-per-channel symbol) denote the source and channel code rates, respectively. Coded source symbols are packetized before modulation and transmission over the communication channel. We use an $M$-ary hierarchical modulation defined by the set of hierarchical parameters
(\alpha_1, \alpha_2, \ldots, \alpha_m). Therefore, for each source symbol, there are \( r_s/(\log_2 M \times r_c) \) channel symbols to be transmitted per unit time. For a rate of \( C \) channel symbols per source symbol per unit time, the communication system is subject to the following constraint: \( r_s/(\log_2 M \times r_c) \leq C \). The overall design goal is to minimize the distortion between the original source and the reconstructed source through finding the optimal packetization strategy and the bandwidth allocation i.e., optimal system parameters \((r_s^*, r_c^*, \alpha_1^*, \alpha_2^*, \ldots, \alpha_m^*)\). This general optimization problem can be simplified (such as using simpler channel models or fixing some of the parameters to reduce the set of parameters subject to optimization) and solved using various methods such as dynamic programming or Lagrangian methods. Note that the cost function is a function of the packetization methodology, the rate-distortion characteristics of the source, the type and rate of channel codes, the channel model and the modulation parameters. Depending on the size of the subset of problems jointly being considered, we will see that the overall optimization problem can be very complicated to solve analytically.

1.4 Thesis Outline

In Chapter 2, we consider a progressive source transmission system consisting of a progressive source coder, a Rate Compatible Punctured Convolutional (RCPC) channel coder and a hierarchical modulation module. Under a bandwidth constraint, we study the performance over an AWGN channel, a flat Rayleigh fading channel and
a frequency selective channel with link breaks. We consider a single carrier system with a Minimum Mean Square Error (MMSE) equalizer at the receiver to cope with the intersymbol interference. The parameters of the system are optimized in a mean distortion sense. The performance is presented using both an average PSNR and a cumulative distribution function of the PSNR for all the channels in consideration. A Lagrangian multiplier method is adapted to find a set of non-linear equations and to solve for optimal system parameters using numerical techniques. Optimal system parameters are used to simulate the overall system and we present the simulated performance results. In addition, we derive a lower bound for the performance improvement of the proposed system over some of the well known transmission schemes.

In Chapter 3, we introduce a novel packetization paradigm for progressive source transmissions. This transmission system uses an embedded bit stream, and both RCPC and semi-random RC-LDPC codes, random block interleavers and a novel packetization methodology before transmitting the coded bit stream over a Binary Symmetric Channel (BSC) with varying cross over probability. In this chapter, we develop a tractable functional optimization procedure that allows us to use a constrained exhaustive search to find a global optimum solution to the optimization problem.

In chapter 4, we investigate a progressive transmission system using a fountain code for lossy packet networks. We note that previous designs assume that perfect Channel State Information (CSI) is available at the transmitter. However, in some
of the point-to-point communication and data multicast scenarios, channel state information might be missing. We first introduce a more generalized Luby Transform (LT) code structure, then use it for the transmission of progressive sources over the Binary Erasure Channel (BEC). We introduce a swapping technique to tailor the generalized code structure for a progressive bit stream transmission. Under a minimum distortion criterion, we find the optimal parameters of the proposed code and compare it to some of the published methodologies.

Some of the technical details, including proofs, derivations and optimal parameters of Chapter 2 and Chapter 3, are given in Appendices A and B. Each chapter gives a detailed overview of the previous research activities and states the motivation of the related work. Results, summaries and conclusions are included at the end of each chapter.
Chapter 2

Coded Hierarchical Modulation for Wireless Progressive Image Transmission

In this chapter, we combine different UEP schemes (hierarchical modulation, JSCC and packetization) together in a distortion-optimal way, and address their interaction in a progressive transmission scenario under various channel models including a frequency selective channel model. The performance of the proposed system is evaluated using an average as well as a cumulative distribution function of the PSNR.

A problem with JSCC is that adding more redundancy constrains the available bandwidth for the source bits. In contrast, hierarchical modulation can provide UEP without constraining the bandwidth. Although progressive multimedia sources
can be protected using each method separately, in this study we will show combining channel coding with hierarchical modulation can take advantage of both. We consider RCPC codes, but the proposed methodologies can be applied to more powerful codes such as Turbo and LDPC codes for improved performance using iterative decoding algorithms at the expense of complexity. The remainder of this chapter is organized as follows. In Section 2.1, we review the related studies in the literature and state explicitly the contribution of this chapter. In Section 2.2, the system model is described in detail. In Section 2.3, different bits-to-symbol assignment strategies are summarized. In Section 2.4, hard decision upper bounds for coded hierarchical modulation are given and the optimization problem is constructed and solved. A lower bound is derived for the performance improvement of the proposed system. Some of the performance results for memoryless channels are given at the end of the section. Section 2.5 discusses a frequency selective channel that models a link breakage scenario and introduces Intersymbol Interference (ISI). Finally, conclusions follow in Section 2.6.

### 2.1 Related Work and Contributions

JSCC-only mechanisms are considered extensively in the literature for progressive coding. In [8] and [9], UEP is provided with non-uniform channel codes for different parts of the encoded image. In [10]–[15], RCPC channel codes, Turbo codes, LDPC codes and Irregular Repeat and Accumulate (IRA) codes are applied
to source packets to improve performance for memoryless channels. RCPC codes are also considered in [16] by applying different channel code rates for different kinds of bits (e.g., sign bits, wavelet coefficient bits, etc) of SPIHT [5] encoded data. However, those studies consider simple channel models such as BSC or AWGN with a binary alphabet. Studies such as [18] divide the UEP task between both the channel encoder and the modulator to help reduce the convolutional coding complexity and to provide unequal protection using a single code rate for the transmission. More recently, studies such as [19] considered modulation-assisted UEP-LDPC codes to achieve a good tradeoff between reliability and spectrum efficiency using a 3-level LDPC code and a QPSK/16QAM mapping for UEP. Also, JSCC is considered in an OFDM setting in [20] to combat ISI, and is combined with space time codes in [21] to provide diversity gain.

Hierarchical modulation is yet another popular strategy to provide UEP. It has been included in various standards [22] and used to give unequal transmission reliability to High Priority (HP) and Low Priority (LP) bits [23]– [25]. The idea of hierarchical modulation combined with progressively compressed signals is not new. In [24], the authors consider hierarchical modulation and progressive image layers encoded using an adaptive discrete cosine transform. Hierarchical modulation is used without channel coding in wireless relay networks in [26] and is shown to yield good unequal protection capability. Adaptive modulation is considered in [27] and [28] within the framework of JSCC. Also, [29] considers both hierarchical modulation and channel coding to provide UEP for 2-layer video transmission. The unequal
protection is achieved by one of the UEP mechanisms while respecting the delay limitations of the transferred video. To the best of our knowledge, the proposed work is the first study that combines different UEP schemes (hierarchical modulation, JSSC and packetization) together in a distortion-optimal way, and addresses their interaction in a progressive transmission scenario.

The main contribution of this study is to address how to combine different UEP mechanisms effectively in a progressive transmission scenario for different wireless channel models for a given bandwidth constraint. A novel packetization strategy is introduced to combine encoded packets of different importance levels through hierarchical modulation to provide a more flexible system than existing coded transmission schemes. We consider a single-carrier system for transmitting fixed-length packets (but variable payload size within the packet). We initially consider AWGN and slowly varying flat Rayleigh fading channels, and then extend some of the simulation results to frequency selective channels with equalization. We show that different UEP methods can be combined to provide enhanced progressive source transmission using efficient combining strategies.

### 2.2 The System Model

A block diagram of the progressive transmission system is shown in Fig. 2.1. Next, we will discuss each component in detail.

A progressive source encoder produces the bit stream. A decoder imple-
Figure 2.1: Baseline system model with related block diagrams. Explanation for each block is given in the text. System is adaptive according to channel conditions. 

mented at the encoder reconstructs the compressed source and extracts the Rate-Distortion (R-D) characteristics. The distortion \( d_l \) that results upon receiving packets up to and including packet \( l \) is determined. We use Mean Squared Error (MSE) as our distortion metric. The physical meaning of MSE is the cumulative squared error between the reconstructed image at the receiver and the original image. In particular, \( d_l \) is given by

\[
d_l = \frac{1}{L_x L_y} \sum_{k=1}^{L_x} \sum_{s=1}^{L_y} |I(k, s) - R_l(k, s)|^2.
\]  

(2.1)

where \( L_x \) and \( L_y \) are the horizontal and vertical sizes of the image in pixels, \( I(k, s) \) is the original image pixel value at \( (k, s) \), and \( R_l(k, s) \) represents the reconstructed image pixel value using only the first \( l \) packets of the encoded packet stream.

We denote the set of bits that are in the \( l \)th packet \( \mathcal{P}_l \) and the sets of bits in the first and second halves of the packet stream \( \mathcal{P}'_U = \bigcup_{l=1}^{N/2} \mathcal{P}_l \) and \( \mathcal{P}''_U = \bigcup_{l=N/2+1}^N \mathcal{P}_l \), respectively. \( N \) is the total number of packets in the image, and is assumed to be
even. Two bytes of Cyclic Redundancy Check (CRC) code are appended to the
$\ell_{s(l)}$ bits of information for $\mathcal{P}_l$, where $l = 1, 2, \ldots, N$, along with $m$ additional bits
to flush the memory and terminate the decoding trellis in the all-zero state. A to-tal of $\ell_{s(l)} + 16 + m$ bits are then encoded using RCPC code rate $\omega_l$ for packet $l$.
Packets are ordered and modulated using one of the packetization and hierarchical
modulation techniques described in Section III. Finally, the output symbol stream
is sent through the wireless channel. Perfect CSI is assumed at the receiver. It is
used to determine the optimal parameters of the system. Finally, the hierarchical
demodulator receives the optimized parameter information, demodulates the incom-
ing symbols and decodes the codewords using a Viterbi decoder. In the proposed
setting, the receiver is assumed to feed the channel parameters back to the trans-
mmitter over a reliable channel. In practice, the channel parameters are encoded with
heavy protection and placed in packet header sections. We assume that there is not
a significant loss in throughput due to the transmission of the CSI.

We define the vectors $\mathbf{d} := [d_0 \ d_1 \ \ldots \ d_l \ \ldots \ d_N]$ and $\mathbf{r} := [R_1, R_2]$, where $R_1$ and
$R_2$ are the code rates of the first half and the second half of the packets. In other
words, we use $R_1$ for the bits in $\mathcal{P}_l'$ ($\omega_l = R_1$ for $l = 1, \ldots, N/2$) and $R_2$ to protect
the bits in $\mathcal{P}_l''$ ($\omega_l = R_2$ for $l = N/2 + 1, \ldots, N$). As will be clear in Section III,
the packetization and combining strategy using only hierarchical modulation puts a
natural constraint on the Bit Error Rate (BER) performance of the bits in $\mathcal{P}_l''$ and
$\mathcal{P}_l'$. That is, once a hierarchical parameter is selected to achieve the target HP BER,
it will also determine the LP BER. The purpose of using two different channel code
rates is to alleviate this constraint and make the system more flexible in terms of assigning target BERs.

We use a finite discrete set of RCPC codes $C_r = \{c_1, c_2, ..., c_n\}$ [30]. Our system uses packets of $\nu$ bits with code rates $R_1 = \frac{a_1}{b_1}$ and $R_2 = \frac{a_2}{b_2}$ with $\text{g.c.d}\{a_1, b_1\} = \text{g.c.d}\{a_2, b_2\} = 1$. The following procedure is used to determine the packet size, given $R_1$ and $R_2$. We begin with a nominal value $\nu_m$. If $\nu_m$ is divisible by $\text{l.c.m.}\{b_1, b_2\}$, then the packet size $\nu = \nu_m$ is used. Otherwise, we use $\nu = \left\lfloor \frac{\nu_m}{\text{l.c.m.}\{b_1, b_2\}} \text{l.c.m.}\{b_1, b_2\} \right\rfloor$ where $\left\lfloor . \right\rfloor$ is the floor function\(^1\). For a given transmission rate $r_{tr}$ in bits per pixel (bpp), the number of packets is $N = \left\lceil \frac{r_{tr} \times L_x \times L_y}{\nu} \right\rceil_{\text{even}}$, where $\left\lceil . \right\rceil_{\text{even}}$ rounds down to the nearest even integer. The source rate $r_s$ in bpp is given by $r_s = \frac{\sum l b_s^{(l)}}{L_x \times L_y}$, where $b_s^{(l)} = \nu r_l - 16 - m$.

For a given $r$, first $d$ is determined and used in our optimization algorithm to construct the optimal hierarchical parameters ($\alpha^* := [\alpha_1^*, \alpha_2^*, \ldots, \alpha_{N/2}^*]$). Then, we optimize over all possible $(R_1, R_2)$ to find $r^* = [R_1^*, R_2^*]$. The optimal parameters $\{\alpha^*, r^*\}$ are chosen to minimize the reconstructed source distortion at the receiver, as will be discussed later.

### 2.3 Bits-to-Symbol Assignment Methodologies

Different bits-to-symbols assignment strategies will be called Packetization [31]. We consider two combining strategies: Folded Packetization ($FP$) and Sequen-

\(^1\text{Floor function rounds the argument of the function down to the nearest integer}\)
tial Packetization (SP). As shown in Fig. 2.2a, for FP, \( \alpha_i \) with \( i = 1, 2, \ldots, N/2 \) is used to combine bits in \( P_i \) and \( P_{N-i+1} \). This means that for \( z = 1, \ldots, \nu \), the \( z \)th bit of \( P_i \) and the \( z \)th bit of \( P_{N-i+1} \) are encoded together using, say, the H-4PAM(\( \alpha_i \)) constellation to produce the modulated symbols.

![Diagram of Packetization](image)

**Figure 2.2:** Two different assignments of packetized bits to symbols (out of \( \prod_{i=1}^{N/2} \binom{N-2i}{2} \) different possible ways) in a progressively encoded source. \( P_i \): \( i \)th chunk of information.

We use hierarchical parameters \( \{ \alpha_i \in \mathbb{R} : \alpha_i \in [t_i, u_i] \} \), and \( t_i, u_i \) are lower and upper bounds, respectively, for \( \alpha_i \). For SP, as shown in Fig 2.2b, \( \alpha_{(k+1)/2} \) is used to combine bits in \( P_k \) and \( P_{k+1} \), where \( k = 1, 3, 5, \ldots, N-1 \). The assignment strategy for FP is originally used in [32] and is called *Packet Reversed Packet Combining* in [33], where the bits are encoded using XOR operation (bit combining). The idea was to correct bit errors in Automatic Repeat reQuest (ARQ) through multiple transmissions of the same message. The same assignment strategy is utilized in conjunction with hierarchical modulation in [34]. It is used with channel coding and hierarchical modulation in [35] for AWGN and flat Rayleigh fading channels.

An error in a progressive coded bit stream leaves the remaining part of the stream undecodable [9], [10], [5]. Therefore, in case of an error, the encoded stream
is truncated, and the decoded packets are used to reconstruct the source. If we consider fixed length packets, the number of information bits \((b_l^{(l)})\) within each packet varies based on the channel code rate used. The optimized system assigns the set of hierarchical parameters \(\alpha^*\) to determine relative reliability among the packets. Previous UEP JSCC techniques protect the progressive stream using a discrete set of channel code rates. If the optimal protection can be provided with a code that falls in between two available code rates in the set, the closest one is chosen to protect the packet [9]. In our system, UEP is flexibly provided using both a discrete code set and continuous-valued hierarchical parameters throughout the progressive bit stream.

2.4 Performance Analysis

In this section, we describe the distortion-minimization framework for our system and give Packet Error Rate (PER) expressions for RCPC codes. A lower bound for the system performance improvement over Equal Error Protection (EEP) is analytically derived, and numerical results are given to support the argument. We assume ideal coherent detection and perfect CRC error detection.

Let \(\rho_l(\gamma)\) be the average bit error probability for the \(l\)th packet as a function of \(\gamma = \frac{E_b}{N_0}\), where \(E_b\) is the average bit energy and \(N_0\) is the power spectral density of the noise, and assume the all-zero codeword is transmitted. For a given code rate \(\beta \in C_r\), let \(\delta^{(\beta)}\) represent the distance to the all-zero codeword of the path
being compared with the all-zero path at some node in the trellis. For a BSC with 
crossover probability $\rho_l(\gamma)$, the probability of selecting the incorrect path is 
given by [1]

$$P_{\delta(\beta)} = \begin{cases} 
\sum_{j=\delta(\beta)+1}^{\delta(\beta)} \binom{\delta(\beta)}{j} (1 - \rho_l(\gamma))^{\delta(\beta) - j} \rho_l^j(\gamma) & \text{if } \delta(\beta) \text{ odd} \\
\sum_{j=\delta(\beta)/2 + 1}^{\delta(\beta)} \binom{\delta(\beta)}{j} (1 - \rho_l(\gamma))^{\delta(\beta) - j} \rho_l^j(\gamma) + \frac{1}{2} \binom{\delta(\beta)/2}{\delta(\beta)/2} (1 - \rho_l(\gamma))^{\delta(\beta)/2} \rho_l^{\delta(\beta)/2}(\gamma) & \text{if } \delta(\beta) \text{ even}
\end{cases} \tag{2.2}$$

The number of different values of $P_{\delta(\beta)}$ is equal to the number of different 
BERs that can be provided by the hierarchical $M$-ary constellation. For example, 
in 4-PAM, we need to compute (2.2) for two BERs, which are given for AWGN 
($\rho_{HP}(\alpha, \gamma), \rho_{LP}(\alpha, \gamma)$) in Equations (2.3), (2.4) and for flat Rayleigh fading channels 
($\rho_{Ray}^{HP}(\alpha, \gamma), \rho_{Ray}^{LP}(\alpha, \gamma)$) in Equations (2.5), (2.6) [36].

$$\rho_{HP}(\alpha, \gamma) = \frac{1}{2} \left[ Q\left(\sqrt{\frac{8\gamma}{1 + \alpha^2}}\right) + Q\left(\sqrt{\frac{8\gamma\alpha^2}{1 + \alpha^2}}\right) \right] \tag{2.3}$$

$$\rho_{LP}(\alpha, \gamma) = \frac{1}{2} + \frac{1}{2} \sum_{s=0}^{1} \sum_{m=0}^{1} (-1)^{s+m} Q\left(\frac{(-1)^s(1 + \alpha)}{2} + \alpha^m \sqrt{\frac{8\gamma}{1 + \alpha^2}}\right),$$

where $Q(x) = \frac{1}{\sqrt{2\pi}} \int_{x}^{\infty} e^{-\frac{z^2}{2}} dz$. \tag{2.4}

$$\rho_{Ray}^{HP}(\alpha, \gamma, \sigma) = \frac{1}{2} - \frac{1}{4} \sqrt{\frac{\lambda}{1 + \lambda}} - \frac{1}{4} \sqrt{\frac{\alpha^2 \lambda}{1 + \alpha^2 \lambda}} \text{ where } \lambda = \frac{8\sigma^2 \gamma}{(1 + \alpha^2)} \tag{2.5}$$

$$\rho_{Ray}^{LP}(\alpha, \gamma, \sigma) = \frac{1}{2} + \frac{1}{2} \sum_{s=0}^{1} \sum_{m=0}^{1} (-1)^{s+m} \left(\frac{1}{2} - \frac{\text{sgn}(\tilde{\Omega})}{2\sqrt{1 + \frac{1}{\gamma}}}\right),$$

where $\tilde{\Omega} = \frac{(-1)^s(1 + \alpha)}{2} + \alpha^m, \gamma = \tilde{\Omega}^2 \lambda \tag{2.6}$
Depending on the symbol sent, BERs for a particular bit location can be quite different. For example, using H-4PAM, we can have different BERs for HP bits depending on the values of $\alpha$ and of LP bits. We use $\epsilon_1$ to denote the BER for the HP bit if the LP bit is 0, and $\epsilon_2$ denotes the BER for the HP bit if the LP bit is 1 where $(\epsilon_1, \epsilon_2)$ can be given for AWGN and flat independent Rayleigh channels as [36]

$$
(\epsilon_1, \epsilon_2) = \begin{cases} 
\text{AWGN:} & \left( Q \left( \sqrt{\frac{8\gamma}{1+\alpha^2}} \right), Q \left( \sqrt{\frac{8\gamma\alpha^2}{1+\alpha^2}} \right) \right) \\
\text{Rayleigh:} & \left( \frac{1}{2} - \frac{1}{2} \sqrt{\frac{\lambda}{1+\lambda}}, \frac{1}{2} - \frac{1}{2} \sqrt{\frac{\alpha^2\lambda}{1+\alpha^2\lambda}} \right)
\end{cases}
$$

where $\lambda = \frac{8\sigma^2\gamma}{1+\alpha^2}$, $\gamma(=\frac{E_b}{N_0})$ is the average SNR per bit, $\alpha$ is the hierarchical parameter, $Q(z) = \frac{1}{\sqrt{2\pi}} \int_z^{\infty} e^{-\frac{x^2}{2}} dx$ and $\sigma$ is the parameter of the Rayleigh probability density function (pdf) which is given by $f(x; \sigma) = \frac{e^{-\frac{x^2}{2\sigma^2}}}{\sigma \sqrt{2\pi}}$.

Note that since LP bits are equally likely to be 0 or 1, the average BERs of the HP bits given in (2.3) and (2.5) for AWGN and flat Rayleigh, are simply given by $\frac{1}{2}(\epsilon_1 + \epsilon_2)$. Thus, the channel to which the bits of the H-4PAM symbol are exposed can be modeled as a 2-state channel with transition probabilities 0.5, 0.5, where the channel in one state is a BSC with crossover probability $\epsilon_1$, and the channel in the other state is a BSC with crossover probability $\epsilon_2$. We can also generate a similar set of error probabilities for LP bits. We show in Appendix A that (2.2) is applicable for $M$-ary hierarchical modulation where the channel for each bit location can be modeled as a two state channel [37].
2.4.1 Packet Error Rate Approximation

Our objective is to minimize a cost function that contains Packet Error Rates (PER). Let us define \((\overline{Q}_b^l)\) to be the union bound for the average bit error probability for the bits in packet \(l\) [1]. Using the formulation in [38], the PER for packet \(l\) \((PER_l)\) can be upper bounded by

\[
PER_l \leq 1 - \left(1 - \overline{Q}_b^l\right)^{b_l^*} = 1 - \left(1 - \frac{1}{p} \sum_{\beta_1 = \delta_{\text{free}}}^{\infty} c_{\beta} P_{\beta}^l \right)^{b_l^*},
\]

where \(p\) is the puncturing period, \(\delta_{\text{free}}\) is the free distance of the code, and \(c_{\beta}\) is the coefficient of the bit Input Weight Enumeration Function (IWEF) of a given code \(\beta \in C_r\) [30].

In this formulation, the bounds derived can be very loose, especially at low SNR values. Therefore, we use a Nonlinear Least Squares Regression (NLSR) technique to approximate \(PER_l\) in (2.7). For packet \(l\), the approximation is \(\widehat{PER}_l = 1 - A^* e^{B^* \times \overline{Q}_b^l}\) where \(A^*, B^*\) are parameters chosen according to the following criterion:

\[
(A^*, B^*) = \arg \min_{A, B \in \mathbb{R}} \left\{ \sum_{j=1}^{s} \left| \overline{PER}_l^{(\gamma_j)} - \left(1 - A \times 10^{B \times \overline{Q}_b^{(\gamma_j)}}\right) \right|^2 \right\}
\]

where \(s\) is the number of SNR values used in the approximation and \(\overline{PER}_l^{(\gamma_j)}\) is the average PER for packet \(l\), i.e., the expected value of the random variable that is the outcome of a Monte Carlo simulation at each average SNR \(\gamma_j\). Note that the actual PER is a random variable, since, for each channel realization, the packet error rate is defined to be the number of packets in error divided by the total number of packets received. Also, \(\overline{Q}_b^{(\gamma_j)}\) is the hard decision upper bound for the bit error
Table 2.1: Some sample values of $A^*, B^*$ that are used in our simulations for a set of RCPC code rates

<table>
<thead>
<tr>
<th>Code Rate</th>
<th>$A^*$</th>
<th>$B^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>8/9</td>
<td>2.252</td>
<td>3.121</td>
</tr>
<tr>
<td>4/5</td>
<td>2.737</td>
<td>3.169</td>
</tr>
<tr>
<td>2/3</td>
<td>3.879</td>
<td>3.759</td>
</tr>
<tr>
<td>4/7</td>
<td>4.838</td>
<td>4.272</td>
</tr>
<tr>
<td>1/2</td>
<td>5.021</td>
<td>3.978</td>
</tr>
<tr>
<td>4/9</td>
<td>5.849</td>
<td>4.342</td>
</tr>
<tr>
<td>2/5</td>
<td>6.359</td>
<td>4.474</td>
</tr>
<tr>
<td>4/11</td>
<td>7.516</td>
<td>5.201</td>
</tr>
<tr>
<td>1/3</td>
<td>8.134</td>
<td>5.472</td>
</tr>
<tr>
<td>4/13</td>
<td>8.438</td>
<td>5.392</td>
</tr>
<tr>
<td>2/7</td>
<td>8.561</td>
<td>5.262</td>
</tr>
<tr>
<td>4/15</td>
<td>9.099</td>
<td>5.440</td>
</tr>
<tr>
<td>1/4</td>
<td>9.655</td>
<td>5.671</td>
</tr>
</tbody>
</table>

probability evaluated at each $\gamma_j$. Finally, $\{\gamma_j\}_{j=1}^s$ is the set of average SNR values we use to apply the NLSR for a good functional approximation to the simulated average PER values ($\{\overline{PER}_i^{(\gamma_j)}\}_{j=1}^s$) over the range $[\min\{\gamma_j\}, \max\{\gamma_j\}]$. Based on our experimental observation, $s = 10$ for the SNRs of interest yields an accurate estimate (i.e., $A^*, B^*$ of the functional approximation in Equation (A.1)). Some sample values of $A^*, B^*$ that are used in our simulations for a set of RCPC code rates are given in Table 2.1.

For example, we simulated coded conventional 4PAM and 16QAM (i.e., $\alpha = 3$) using $FP$ and a packet size of 450 bits with the code rate $\beta = 1/2$ and we plot
PER versus SNR per bit (Fig. 2.3 and Fig. 2.4) for both HP and LP packets, assuming the same average power per constellation. The NLSR is a better match to simulation results for the range of SNRs of interest. This functional approximation will be helpful later in formulating the cost function which needs to be minimized.

![Graph](image)

**Figure 2.3**: Proposed NLSR versus simulation results and upper bounds derived in this paper for both HP and LP priority classes using 4PAM and an RCPC code rate $\beta = 1/2$.

### 2.4.2 Construction of the Optimization Problem

For a given $\gamma$ and $r$, we want to select vector $\alpha^*$ so as to minimize the expected distortion $\overline{D}_\alpha$. Assuming independent packet losses, it can be shown that
Figure 2.4: Proposed NLSR versus simulation results and upper bounds derived in this paper for both HP and LP priority classes using 16QAM and an RCPC code rate $\beta = 1/2$.

$D_{\alpha}$ can be expressed (for both $FP$ and $SP$) as [39]:

$$D_{\alpha} = \sum_{l=0}^{N-1} \hat{PER}_{l+1} \prod_{i=0}^{l} (1 - \hat{PER}_i) d_l = d_0 - \sum_{l=1}^{N} \prod_{i=1}^{l} (1 - \hat{PER}_i) \Delta_l$$  \hspace{1cm} (2.9)

where $\hat{PER}_{l+1} \prod_{i=0}^{l} (1 - \hat{PER}_i)$ is the probability of having the first $l$ packets correct and an error in the $(l+1)$th packet, and $\Delta_l = d_{l-1} - d_l \geq 0$ is the amount that the distortion is reduced by having the $l$–th packet received error-free given that all previous $l-1$ packets were correctly received. We define $\hat{PER}_0 = 0$ and $\hat{PER}_{N+1} = 1$. Note that $\hat{PER}_l$ depends on the average bit error rate bound ($Q_b$), which is a function of the channel code rate and the hierarchical modulation parameters. Our optimization problem is given by

$$\min_{\omega_l \in C_r, \omega_{(l+1)} \in \tilde{C}_r} \left\{ \min_{\alpha_{l+1}} \left\{ \sum_{i=1}^{N/2} \min_{\alpha_i} \left\{ D_{\alpha} \right\} \right\} \right\} \text{ subject to } t_i \leq \alpha_i \leq u_i$$  \hspace{1cm} (2.10)
where \( t_i \in \mathbb{R} \) and \( u_i \in \mathbb{R} \) are the lower and upper bounds for the hierarchical parameter \( \alpha_i \), respectively. Since \( d_0 \geq 0 \), we can rewrite the previous expression using (2.9) as

\[
\min_{r, \omega_l \in C_r, l=1, \ldots, N} \left\{ \min_{\alpha_i} \left\{ \sum_{l=1}^{N} \xi_{\omega_l, l} \Delta_l \right\}, \text{ such that } t_i \leq \alpha_i \leq u_i, \right\}
\]

(2.11)

where \( \xi_{\omega_l, l} = - \prod_{i=1}^{l} (1 - \hat{P}E\hat{R}_i) \).

### 2.4.3 Optimization of Hierarchical Parameter Set: \( \alpha \)

Starting with the discrete code set, for each choice of \( r \), we optimize the hierarchical parameters for that value of \( r \). After exhausting all values of \( r \) in the set, we obtain the optimal code rate vector \( r^* \) with the corresponding optimal hierarchical parameters that give the minimum distortion. More specifically, for each \( r \), we determine \( \alpha^* \) by solving Equation (10). Note that it would be possible to approach this iteratively: for a given \( r \), find the corresponding optimal \( \alpha^* \), and then given \( \alpha^* \), determine \( r^* \) by solving Equation (10) for \( r \), etc. We avoid this iterative approach however, because it can result in a local minimum and be complex\(^2\). Instead, for each element of a constrained set of code rates, we solve for \( \alpha^* \), and in the end obtain the global optimum.

Using Equation (2.11), our optimization problem is given by

\[
\min_{r, \omega_l \in C_r, l=1, \ldots, N} \left\{ \min_{\alpha_i} \left\{ \sum_{l=1}^{N} \xi_{\omega_l, l} \Delta_l \right\}, \text{ such that } g_l \leq x_l \right\}
\]

(2.12)

\(^2\)Since the number of code rates in \( C_r \) is discrete, the latter optimization procedure involves an integer programming which is computationally complex (NP-hard).
where
\[ g_l = \begin{cases} 
\alpha_l & 1 \leq l \leq N/2 \\
-\alpha_{l-N/2} & \frac{N}{2} + 1 \leq l \leq N 
\end{cases} \quad \text{and} \quad x_l = \begin{cases} 
u_l & 1 \leq l \leq N/2 \\
-t_{l-N/2} & \frac{N}{2} + 1 \leq l \leq N 
\end{cases} \]

The Lagrangian function of (2.12) can be written as [40]:
\[ \Lambda_D(\alpha) = \Lambda(\alpha_1, \alpha_2, \ldots, \alpha_{N/2}, \lambda_1, \ldots, \lambda_N) = \sum_{l=1}^{N} \xi_{\omega_l} \Delta_l - \lambda_l(g_l - x_l) \] (2.13)

where the parameters $\lambda_1, \lambda_2, \ldots, \lambda_N$ are the Lagrange multipliers. The unconstrained minimization problem is $\min_r \min_\alpha \{ \Lambda_D(\alpha) \}$. The necessary conditions for our optimization problem are given by the Karush-Kuhn-Tucker (KKT) conditions:
\[ \nabla \Lambda_D(\alpha^*) = \left[ \frac{\partial \Lambda_D}{\partial \alpha_1^*}, \ldots, \frac{\partial \Lambda_D}{\partial \alpha_{N/2}^*}, \frac{\partial \Lambda_D}{\partial \lambda_1^*}, \ldots, \frac{\partial \Lambda_D}{\partial \lambda_N^*} \right] = 0 \] (2.14)

with $\lambda_l^*(g_l^* - x_l) = 0$, $\lambda_i^* \geq 0$ and $t_i \leq \alpha_i^* \leq u_i$. We use numerical methods to solve the set of nonlinear equations in (2.14) for the $\alpha^*$ that minimizes $\overline{D}_{\alpha}$ [41].

Also in [42], the KKT conditions are derived similar to the way we derive our KKT conditions for a convex cost function and convex constraints. Descriptions of the system components as well as the optimization procedure are given in Sections II, III and IV and are summarized in a diagram in Fig. 2.5.

2.4.4 A Lower Bound on the Performance Improvement of the Proposed System

We denote the general hierarchical parameter set $\alpha := \{ \alpha_i \in \mathbb{R} \}_{i=1}^{N/2}$ and use $\alpha_a := \{ \alpha_i = a \}_{i=1}^{N/2}$ when all the hierarchical parameters have the same real value.
Figure 2.5: Functional flow diagram of the proposed optimization scheme.
For example, \( \alpha_3 \) means that all hierarchical parameters have the value 3; this corresponds to conventional modulation. As a baseline for comparison, we consider an EEP scheme that uses \( SP \) and conventional modulation, i.e., \( \alpha_3 \).

**Proposition 2.1:** We can order the expected distortion of the systems as follows:

\[
\mathbb{E}[D^{SP}(R^*, R^*, \alpha_3)] \geq \mathbb{E}[D^{FP}(R^*, R^*, \alpha_3)] \geq \mathbb{E}[D^{FP}(R^*_1, R^*_2, \alpha^*)]
\]

where \( D^j(R_1, R_2, \alpha) \) is the distortion using packetization \( j \in \{SP, FP\} \), channel code rates \( R_1 \) and \( R_2 \) for the first and second halves of the packet stream, respectively, and hierarchical parameter set \( \alpha \). The EEP system uses \( R^* \) as the single optimal code rate. For the system that uses \( FP \) and a single optimal hierarchical parameter \( \alpha^* \), \( (R^*_1, R^*_2) \) is the optimal rate pair and \( \alpha_{\alpha^*} \) is the vector of identical hierarchical parameters \( \alpha^* \).

**Proof:** The second inequality in (2.15) follows because optimal parameters \( (R^*_1, R^*_2, \alpha_{\alpha^*}) \) minimize expected distortion. The third inequality arises because we increase the parameter space from 3 to \( \frac{N}{2} + 2 \) and optimize each parameter value. Therefore we need to show only the first inequality in (2.15). In Appendix B, we show that

\[
\Delta \mathcal{D}_N(\alpha_3; \Omega) \triangleq \mathbb{E}[D^{SP}(R^*, R^*, \alpha_3)] - \mathbb{E}[D^{FP}(R^*, R^*, \alpha_3)] \geq 0. \quad \blacksquare \ (2.16)
\]

We define the total distortion measure \( \mathcal{D}_N^j(\alpha; \mathcal{A}) := -\sum_{i \in \mathcal{A}} \prod_{i=1}^{l_i} P_i \Delta_i \) using only the bits in a given set \( \mathcal{A} \) and using packetization \( j \in \{SP, FP\} \) in an \( N \)-packet
error protection scheme where \( \alpha = [\alpha_1 \alpha_2 \ldots \alpha_{N/2}] \). Also, \( P_i = 1 - \widehat{PER}_i \) is the probability of receiving packet \( i \) reliably. From Equation (2.9), for packetization \( j \) we have \( D_{\alpha} = d_0 + \mathcal{D}_N^j(\alpha; \Omega) \), where \( \Omega = \mathcal{P}_U' \cup \mathcal{P}_U'' \). Finally, the PSNR gap \((\Delta PSNR)\) between systems using \( FP \) and \( SP \) is given by

\[
\Delta PSNR \triangleq 10 \log \frac{255^2}{\mathbb{E}[D_{FP}(R^*, R^*, \alpha_3)]} - 10 \log \frac{255^2}{\mathbb{E}[D_{SP}(R^*, R^*, \alpha_3)]}
\]

\[
= 10 \log \frac{d_0 + D_{FP}^N(\alpha_3; \Omega)}{d_0 + D_{SP}^N(\alpha_3; \Omega)} - 10 \log \frac{d_0 + D_{SP}^N(\alpha_3; \Omega)}{d_0 + D_{FP}^N(\alpha_3; \Omega)}
\]

\[
= 10 \log \frac{d_0 + D_{SP}^N(\alpha_3; \Omega)}{d_0 + D_{SP}^N(\alpha_3; \Omega) - \Delta \mathcal{D}_N(\alpha_3; \Omega)} \geq 0 \quad (2.17)
\]

where

\[
\Delta \mathcal{D}_N(\alpha_3; \Omega) = D_{SP}^N(\alpha_3; \Omega) - D_{FP}^N(\alpha_3; \Omega)
\]

\[
= \mathbb{E}[D_{SP}(R^*, R^*, \alpha_3)] - \mathbb{E}[D_{FP}(R^*, R^*, \alpha_3)] \geq 0 \quad (2.18)
\]

by Proposition 2.1. From (2.15), we have

\[
\mathbb{E}[D_{SP}(R^*, R^*, \alpha_3)] - \mathbb{E}[D_{FP}(R_1^*, R_2^*, \alpha^*)] \geq 0 \quad (2.19)
\]

\[
\geq \mathbb{E}[D_{SP}(R^*, R^*, \alpha_3)] - \mathbb{E}[D_{FP}(R^*, R^*, \alpha_3)]
\]

\[
= \Delta \mathcal{D}_N(\alpha_3; \Omega) \geq 0. \quad (2.20)
\]

Thus, \( \Delta PSNR \) is a lower bound for the performance improvement over the EEP scheme of the proposed system.
2.4.5 Numerical Results for Memoryless Channels

Our first simulation demonstrates the average BER performance of different packetized bits-to-symbols assignments. We present simulation results for the coded cases and numerical computation of BER expressions for the uncoded case. We consider both H-4PAM and H-16QAM and assume the same average power per constellation. Fig. 2.6 and Fig. 2.7 show the performance of different packetization schemes using conventional modulation. There is a BER gap between uncoded HP bits and LP bits which is inherent to PAM signalling, and therefore to higher QAM constellations. This gap widens when we use lower channel code rates because parity bits are also unequally protected. There is no gap in SP, because the average HP BER equals the average LP BER.

Next, we apply code rates $R_1$ and $R_2$ for the first and second halves of the total packet stream i.e. $\mathbf{r} = [R_1 \ R_2]$. We initially use $\nu_m = 450$ bits (we will later look at different values of $\nu_m$) and the RCPC code set with constraint length $K = 7$ from [30]: the code rate set is $C_r = \{\frac{8}{9}, \frac{4}{5}, \frac{2}{3}, \frac{4}{7}, \frac{1}{2}, \frac{4}{9}, \frac{2}{5}, \frac{1}{3}, \frac{4}{11}, \frac{1}{7}, \frac{2}{9}, \frac{4}{15}, \frac{1}{4}\}$. A CRC from [10] is used for error detection. Standard grayscale (8 bpp) images $Lena(512 \times 512)$, $Barbara(512 \times 512)$, $Goldhill(512 \times 512)$, $Peppers(512 \times 512)$ and $Baboon(512 \times 512)$ are encoded using the SPIHT and JPEG2000 [43] algorithms. The header information of the encoded bit stream at almost all practical transmission rates is negligibly small and therefore assumed to be error-free throughout. Because of space limitations, we will only show the results for $Lena$ and $Barbara$ using SPIHT.
Figure 2.6: HP and LP BER performances using uncoded and coded \((r = 1/2)\) conventional 4PAM/16QAM for AWGN channel.

Figure 2.7: HP and LP BER performances using uncoded and coded \((r = 1/3)\) conventional 4PAM/16QAM for AWGN channel.
For the other images, the proposed scheme shows similar performance gains. The transmission rate ($r_{tr}$) is 0.25bpp. For around $10^4$ different channel realizations, we simulate the systems to obtain MSE values. We then average the MSE values before converting them to average PSNR. In the proposed system, hierarchical parameters and channel code rates are found by solving the optimization problem. We introduce the following systems:

- **seqConv1**: $SP$, conventional modulation. One optimal code rate chosen from $C_r$.

- **foldConv1**: $FP$, conventional modulation. One optimal code rate chosen from $C_r$.

- **foldHier1**: $FP$, Hierarchical modulation ($\alpha^*$). One optimal code rate chosen from $C_r$.

- **foldHier2**: $FP$, Hierarchical modulation ($\alpha^*$). Two optimal code rates chosen from $C_r$.

We ignore the other four combinations: *seqConv2, seqHier1, seqHier2* and *foldConv2*. First of all, combining $SP$ and hierarchical modulation is not useful because consecutive packets bear almost the same significance in terms of end-to-end distortion. Also, by switching from $SP$ to $FP$, *foldConv2* gives lower distortion than *seqConv2*. Finally, note that *foldConv2* is a special case of *foldHier2* when $\alpha = \alpha_3$. Thus, the performance of those systems is not shown.
Note that $seqConv1$ is an EEP scheme, because it assigns a fixed average BER (corresponding to $\alpha_3$) for every pair of packets. Fig. 2.8 shows the performance of various systems. The UEP schemes always perform better than the EEP scheme. The non-concave behavior of these curves is a consequence, at least in part, of $C_r$ being discrete. In addition to this constraint, our design constraint is that the $\alpha$ values determine the BER for each packet in the first and second halves of the packet stream simultaneously. However, the $foldHier2$ system somewhat alleviates both constraints by employing two different code rates for the two halves of the stream. We also observe a slight performance improvement when we use more than two channel code rates at the expense of greater complexity.

The optimal hierarchical values, $\alpha^*$, as a function of packet index, are plotted in Fig. 2.9 for the first half of the stream. The figure shows that $\alpha_1 \leq \alpha_2 \leq \cdots \leq \alpha_{N/2}$, which means that earlier packets in the stream are more heavily protected by the hierarchical modulation. The discrete nature of the code set is the cause of nonuniform gains going from one UEP scheme to another, and the cause of different gains at different SNRs. At low SNRs, the gap between the curves becomes more pronounced, as UEP is more effective when the channel degrades.

Another interesting observation is that $R_1 \geq R_2$ for $foldHier2$. However, in a JSCC-only UEP scheme [9], we would expect $R_1 \leq R_2$ [39], i.e., we would expect to protect the first part more heavily than the second part. This is not the case when JSCC is used with hierarchical modulation simply because the hierarchical parameters adjust themselves to protect the bits of the first half more than the
Figure 2.8: Performance of different systems for an independent flat Rayleigh fading channel. JSCC EEP is also drawn for comparison. This figure uses the approximate PER expressions by nonlinear regression for both Lena and Barbara images.

Figure 2.9: Optimal hierarchical parameters ($\alpha^*$) for different channel code rate pairs at SNR=7dB for an AWGN channel. It also shows the corresponding calculated values of MSE. (2/3,2/5) is the optimal code rate pair.
bits of the remaining half. As long as these parameters are able to compensate for the decreased protection due to the FEC, \( R_1 \geq R_2 \) can improve the system performance by allocating more information bits to the first half of the packets, where the favorable hierarchical modulation parameters ensure their reliable transfer. This leads to better reconstruction quality.

In Fig. 2.10, PER as a function of Packet Index (PI) number is shown at SNR=10dB with both the optimal pair of codes and one with reverse order. The case where \( R_1 \geq R_2 \) is seen to protect almost all the packets better than the \( R_2 \geq R_1 \) case. Lastly, the system can provide as many UEP levels as the number of packets, although the system uses only two code rates. The step jump for \( \text{foldConv1} \) is due to the natural BER gap between HP and LP bits when we use \( \alpha_3 \), and the channel coding which increases this gap. On the other hand, the step jump in the system using code rate pair (2/3, 8/9) is due to the different channel code rates with different protection capabilities. Also for this system, adaptive hierarchical parameters cannot compensate for the performance gap due solely to channel coding. In \( \text{foldHier2} \), however, this gap due to different channel code rates is compensated with adaptive hierarchical parameters and we observe no step jump in the PER performance.

We also have considered the H-16QAM constellation. To begin, we choose \( r_{tr} = 0.15 \text{bpp} \) for H-4PAM and let the two constellations have the same average power. Since H-16QAM transmits twice the number of bits of H-4PAM in a given unit time, the effective transmission rate is doubled i.e., \( r_{tr} = 0.3 \text{bpp} \) for the H-16QAM system. Fig. 2.11 shows the \( \text{foldHier1} \) system using different hierarchical
modulations for a slowly varying flat Rayleigh fading channel.

Finally, we present numerical calculation results of the performance improvement lower bound ($\Delta PSNR$) given in (2.16) to find the effect of transmission rate ($r_{tr}$) in bpp and the packet size in bits (which previously in this paper were taken to be 0.25bpp and $\sim$ 450 bits, respectively) of the proposed system. For a flat Rayleigh fading channel using H-4PAM at SNR = 7dB, $\Delta PSNR$ is calculated and plotted in Fig. 2.12 as a function of packet size and $r_{tr}$. The jagged curves are contours along which $\Delta PSNR$ is the same up to four digit precision. The reason that the contours are jagged is because the code set is discrete, so the system is unable to find the real number optimal rate and chooses the closest available code rate. Clearly, as the packet size decreases, the performance improvement increases because the PER decreases. However, the PSNR performances of all compared systems become lower
Figure 2.11: Proposed system using H-4PAM, H-16QAM constellations and the corresponding upper bounds for $PER_l$ given in (2.7) in formulation of the cost function instead of using the approximation $\hat{PER}_l, l = 1, 2, \ldots, N$ and the optimal values in (A.1).

Figure 2.12: Performance upper bound is calculated for different transmission rates (in bpp) and packet sizes, illustrating the effect of transmission rate and packet size on the proposed system.
because of increased overhead and redundancy introduced by the CRC and channel coding. For example, when \( \nu \approx 200 \) and \( r_{tr} \approx 1 \), \( \Delta PSNR \approx 4.27 \)dB, yet the EEP system has PSNR=25.5dB, which is very low quality, and the UEP due only to packetization has PSNR=30dB, which is significantly better but still not high quality. Also, at small enough packet sizes, the performance improvement is small because \( N \) becomes large and both systems use the same optimal channel code rate. In addition, the performance improvement increases with the transmission rate for the range of rates considered.

### 2.4.6 Computational Complexity

We experimentally observed the following: for a fixed \( r \), bit budget constraint \( B \) and packet size \( \nu \), the computational complexity of the optimization procedure (i.e., solving Equation (2.10)) grows approximately linearly with the number of optimized hierarchical parameters (\( |a| \)). Note that the most complex optimization procedure belongs to \texttt{foldHier2}. After running an exhaustive search for the best code rate, the complexity of the optimization procedure for \texttt{foldHier2} grows at most quadratically in the number of elements in \( C_r \) and linearly in the number of hierarchical parameters (and therefore in the number of packets). The growth is at most quadratic because, for some code rates, the numerical optimization tools we use terminate early in their iterations simply because those code rates are either too weak or too powerful for a given channel state.
2.5 Proposed Scheme under Frequency Selective Channels

In this section we will extend our results from AWGN and flat Rayleigh fading channels to frequency selective channels. We consider a single carrier system for transmitting a progressive bit stream through a frequency selective channel. We use equalization to combat ISI due to the frequency selectivity of the channel. We will show that for this channel, the combining/packetization strategy yields performance improvement compared to other UEP schemes and the EEP scheme introduced in the previous section.

We consider short distance high data rate communications where the Line of Sight (LOS) component of the channel can be lost through random link breaks. Examples include wireless internet connections in malls, airports and hotels. In capacious spaces such as these, the multipath delay spread is large, so the coherence bandwidth is fairly small. At reasonably high data rates, the required bandwidth would typically exceed the coherence bandwidth and the channel will be characterized as being frequency selective. The random link breaks are commonly due to mobile objects obstructing the transmitter-receiver direct communication. In malls, airports and hotels, link breakages are common because of people walking around. Therefore, we are interested in considering such scenarios involving both frequency selective channels and link breaks.

The symbol duration is assumed to be small enough that the fading coeffi-
cients are constant during the transmission of a symbol. Since the cost function is difficult to formulate in closed form, and exhaustive search is not a plausible option, it is infeasible to optimize all the parameters of the system for the given frequency selective channel. Instead, we use the optimized hierarchical parameters found for the flat Rician/Rayleigh fading case at a given average SNR and then optimize the channel code rates. In other words, an optimal rate schedule is found based on the suboptimal hierarchical parameter set for the given channel model. The performance results will be shown to give gains around 1dB over the EEP scheme, and can be thought of as lower bounds on the performance improvement of the fully optimized system over the EEP scheme.

2.5.1 Channel Model

In multipath channels, paths often arrive in clusters. Our channel model includes this clustering phenomenon and random fading gains with deterministic multipath delays, very similar to the clustering phenomenon found in IEEE 802.15.3c [44]. The first tap gain of the channel is given as a mixture of Rician and Rayleigh distributions to model an abrupt link breakage. The parameter $\kappa$ is used to denote the percentage of time the LOS link is available. NLOS components are Rayleigh distributed.

The general fading process is a two-component complex stationary random process for the $i$th multipath, described as $h_i(t) = \delta_{i,0} \Gamma_i(t)e^{j\Psi_i(t)} + \tilde{a}_i(t), i = 0, 1, \ldots$ [44], where $\Gamma_i(t)$ is the amplitude of the specular component of the $i$th multipath of
the fading process, $\Psi_i(t)$ is the uniformly distributed random phase of the specular component of the $i$th multipath, and $\tilde{a}_i(t)$ is the diffuse fading component of the $i$th multipath which is usually assumed to be a complex zero-mean Gaussian process with independent in-phase and quadrature components, each with variance $\sigma^2_{\tilde{a}_i}$. We drop the time dependence hereafter because channel coefficients are assumed to be the same during the transmission of any particular symbol. However, channel coefficients are allowed to vary from one symbol to another. The current LOS component is statistically dependent on the previous state of the LOS component of the channel. We assume that when there is no LOS component ($\Gamma_0 = 0$), the probability of having a LOS component in the next transmission is $p_1$. Similarly, when there is an LOS component ($\Gamma_0 \neq 0$), the probability of having no LOS component in the next transmission is $p_2$.

For the first subpath, we have a Rician distribution, conditioned on $\Gamma_0$, given by $f_R(r|\Gamma_0) = \frac{r}{\sigma^2_{\tilde{a}_0}} \exp \left\{ -\frac{r^2 + \Gamma_0^2}{2\sigma^2_{\tilde{a}_0}} \right\} I_0 \left( \frac{\Gamma_0 r}{\sigma^2_{\tilde{a}_0}} \right)$ where $I_0(.)$ is the modified Bessel function of the first kind with order zero. Also, for other NLOS components ($\Gamma_j = 0, j = 1, 2, \ldots$), the Rayleigh distribution for the $i$th multipath ($i = 1, \ldots$) is given by $f_x(x) = \frac{x}{\sigma^2_{a_i}} \exp \left\{ -\frac{x^2}{2\sigma^2_{a_i}} \right\}$. The ratio of specular-to-diffuse energy of the first subpath fading component is defined to be ($K$-factor) $K = \frac{\Gamma_0^2}{2\sigma^2_{\tilde{a}_0}}$, and is usually expressed in dB. Since an AWGN channel has a tap gain with power one, the same power constraint on this channel should satisfy $\mathbb{E} \left[ \sum_i |h_i|^2 \right] = \sum_i \mathbb{E}[|h_i|^2] = 1$, where $\mathbb{E}[.]$ denotes the ensemble average [1].

Our frequency selective channel is assumed to consist of $M$ complex chan-
nel coefficients $\mathbf{h} = [h_0, h_1, \ldots, h_{M-1}]$. Let $\phi_i = E[|h_i|^2] = \delta_{i,0} \Gamma_i^2 + 2\sigma_i^2 \in \mathbb{R}$ be the power coefficient for each subpath. Thus, the power vector is given by $\mathbf{\phi} = [\phi_0, \phi_1, \ldots, \phi_{M-1}]$, the entries of which satisfy the power constraint. We use a simplified discrete-time domain representation for the impulse response of the channel. The time axis is divided into $M$ equal time intervals, i.e., each single multipath component is separated by an integer multiple of $1/B$ compared to other multipaths such that $M/B \geq \sigma_T$, where $\sigma_T$ is the rms delay spread of the channel and $B$ is the bandwidth of the baseband equivalent signal. Thus, we write the impulse response as follows:

$$
\Theta(\tau) = \sum_{i=0}^{M-1} h_i \delta(\tau - i/B) 
$$

(2.21)

Typical values for the channel parameters can be found in various specifications for channel models, such as COST 207 [45] or SUI [46]. In our model, similar to [44] and [45], we assume exponential decay of multipath energy. In modeling this energy, we assume two clusters, each having their own decay rate. The cluster decay is modeled using an exponential function. One sample profile based on [45] is shown in Fig. 2.13, where each subpath is placed at the beginning of a subinterval. The multipath delay spread is assumed to be $9/B$ for this particular case. Finally, the parameter $\kappa$ represents the percentage of time the LOS component is available. Since we have a two-state Markov chain, it can be shown that $\kappa = \frac{p_1}{p_2 + p_1} \times 100$. 
2.5.2 Numerical Results for a Frequency Selective Channel

We use an equalizer with nine taps at the receiver based on the MMSE criterion to mitigate the effects of ISI on the overall system performance. We assume perfect CSI at the receiver i.e., the channel tap gains are estimated accurately at the receiver. Based on the current SNR conditions, the MMSE equalizer taps are calculated after each symbol transmission and used to equalize the system. We tested three sample channels with the following power decay profiles [45]:

\[
\phi_1 = [0.9861, 0.0011, 0.0, 0.0, 0.0, 0.0118, 0.0, 0.0010] \quad (2.22)
\]

\[
\phi_2 = [0.9211, 0.0654, 0.0, 0.0, 0.0, 0.0117, 0.0018, 0.0] \quad (2.23)
\]

\[
\phi_3 = [0.7799, 0.1248, 0.0468, 0.0, 0.0, 0.0321, 0.0164, 0.0] \quad (2.24)
\]
We assume that the channel coefficients stay the same during the transmission period of nine symbols and then change based on their corresponding energy delay profiles. In other words, the coherence time is assumed to be approximately equal to the total delay spread of the channel. Simulation parameters (parameter values of Equation (2.21)) for each frequency selective channel are listed in Table 2.2.

**Table 2.2**: Simulation parameters (T:Symbol duration)

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>$h_1$</th>
<th>$h_2$</th>
<th>$h_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Cluster decay</td>
<td>$e^{-0.74T}$</td>
<td>$e^{-0.72T}$</td>
<td>$e^{-0.53T}$</td>
</tr>
<tr>
<td>1st Cluster Ray decay</td>
<td>$e^{-6.81T}$</td>
<td>$e^{-2.65T}$</td>
<td>$e^{-1.77T}$</td>
</tr>
<tr>
<td>2nd Cluster Ray decay</td>
<td>$e^{-2.51T}$</td>
<td>$e^{-1.89T}$</td>
<td>$e^{-0.81T}$</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>71%</td>
<td>71%</td>
<td>71%</td>
</tr>
<tr>
<td>$M$</td>
<td>8</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>$\sigma_T$</td>
<td>$0.5878T$</td>
<td>$0.6626T$</td>
<td>$1.3459T$</td>
</tr>
<tr>
<td>$p_1$</td>
<td>0.81</td>
<td>0.81</td>
<td>0.81/1</td>
</tr>
<tr>
<td>$p_2$</td>
<td>0.33</td>
<td>0.33</td>
<td>0.33/0</td>
</tr>
<tr>
<td>Coherence time</td>
<td>$9T$</td>
<td>$9T$</td>
<td>$9T$</td>
</tr>
<tr>
<td>$K$ (in dB)</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>

We show the following simulation results using RCPC channel codes in conjunction with H-16QAM and folded packetization for three different channels. First, in Fig. 2.14, decoded BER performance is shown as a function of average SNR for channels $h_1$, $h_2$ and $h_3$ with and without MMSE equalization using channel parameters $p_1 = 0.81$, $p_2 = 0.33$ and $K = 10\text{dB}$, which is equivalent to $\kappa = 71\%$ LOS availability. The simulations are run using optimal code rates that minimize the average distortion of $\text{seqConv1}$ as shown in the title of each plot. As can be seen, the equalization improves the system performance by lowering the HP and LP BERs.
while preserving the BER gap in between.

![Graphs of BER for various frequency selective channels](image)

**Figure 2.14**: Effect of the equalization for various frequency selective channels $h_1$, $h_2$ and $h_3$. 
Next, we show some of the results of an image transmission (512 × 512 grayscale Lena image encoded with SPIHT) using the channel \( h_3 \) and compare different packetizations using H-16QAM at an average received SNR = 15dB and transmission rate \( r_t = 0.25 \text{bpp} \). For better understanding of the performance, we plot the Cumulative Distribution Function (CDF) of the PSNR. We also show average PSNR values in parentheses.

Since it is infeasible to optimize hierarchical parameters of the systems \( \text{foldHier1} \) and \( \text{foldHier2} \) for the given frequency selective channel, we use the optimal hierarchical parameter values found for the flat Rayleigh fading case. Fig. 2.15 compares the CDF of PSNR performances of the optimal \( \text{seqConv1, foldConv1} \) and the suboptimal \( \text{foldHier1 (foldHier1*), foldHier2 (foldHier2*)} \) systems under no link breakage assumption i.e., \( p_1 = 1, p_2 = 0 \). The figure suggests that by simply switching the packetization scheme i.e., going from \( \text{seqConv1} \) to \( \text{foldConv1} \), we pick up around 0.3dB gain in average PSNR. When we use suboptimal hierarchical parameters with a single optimal code rate (\( \text{foldHier1*} \)), we pick up an additional 0.2dB in average PSNR over \( \text{foldConv1} \). Finally, \( \text{foldHier2*} \) gives more than 0.35dB average PSNR improvement over \( \text{foldHier1*} \). In total, \( \text{foldHier2*} \) improves the average PSNR performance upon EEP by more than 0.85dB. Another observation is that \( \text{foldHier2*} \) gives more than 1dB PSNR improvement over \( \text{FoldConv1} \) around 13% of the time.

In Fig. 2.16, we consider the link breakage by setting \( p_1 = 0.81, p_2 = 0.33 \). When we optimize the system \( \text{foldHier1} \) for a flat Rayleigh fading channel, the
Figure 2.15: CDF of PSNR for the frequency selective channel $h_3$ using equalization under frequency selective channel without link breakage.

Figure 2.16: CDF of PSNR for the frequency selective channel $h_3$ using equalization under frequency selective channel with link breakage.
optimal code rate turns out to be 4/9. Let us denote the corresponding optimal hierarchical parameter vector as $\alpha_{4/9}^*$. If we use the optimal code rate in conjunction with $\alpha_{4/9}^*$ under our frequency selective channel model, $foldHier1^*$ gives an average PSNR of 29.33dB. When we use optimal double code rates (one for the first half and one for the second half of the bit stream) i.e., the optimal code rate schedule ($r^*$) using $\alpha_{4/9}^*$ for the given frequency selective channel, the optimal channel code rates turn out to be (4/7, 4/11) and the average PSNR is 29.57dB. As can be seen using the channel $h_3$, the partially optimized system $foldHier2^*$ still gives around 1dB average PSNR performance improvement over the EEP scheme $seqConv1$, which is around the same gain seen in flat fading cases. Therefore, one can expect more than 1dB average PSNR gain over the EEP by jointly optimizing the code rates and hierarchical parameters ($foldHier2$). Also, $foldHier2^*$ gives more than 1dB PSNR improvement over $FoldConv1$ around 21% of the time. Comparing this with the no-link break scenario (13% of the time), the proposed scheme is more effective with growing probability of link breakage, as the UEP is more effective with degrading channel conditions.

Finally, Fig. 2.17 shows PSNR versus $E_b/N_0$ performance of various systems using the Lena image and the frequency selective channel $h_3$ with $p_1 = 0.81$ and $p_2 = 0.33$. Similar gains are observed for a range of average SNR values.
Figure 2.17: Performance of different systems for a frequency selective channel with link breaks using $h_3$ and $p_1 = 0.81$ and $p_2 = 0.33$. JSCC EEP is also drawn for comparison. The results correspond to the Lena image.
2.6 Conclusions

In this chapter, we presented a reliable and robust progressive source encoding scheme for fixed packet length image transmission based upon the combined use of several UEP methods. Several different transmission channels are assumed: AWGN, independent flat fading channels, and a frequency selective channel that accounts for a link breakage. Specifically, a packetization methodology that is coupled with both hierarchical modulation and FEC is considered. It is shown in this chapter that the different UEP methods can be judiciously combined to provide enhanced reliability for the transmission of the progressive source, as one of the methods usually alleviates the constraints coming from the others. A lower bound for the performance improvement of the proposed system is derived and shown at various packet sizes and transmission rates to be an indicator for usefulness of the proposed system. Consequently, the proposed scheme is shown to give improvements for wireless progressive image transmission.
Acknowledgement

Chapter 3

Concatenated Block Codes for
Unequal Error Protection of
Embedded Bit Streams

This chapter investigates a robust source transmission system for embedded bit streams through the combined use of a state of the art progressive source encoder and a concatenated block coding mechanism. The proposed scheme efficiently trades off the available total bit budget between information bits and parity bits through efficient information block size adjustment, concatenated block coding and random block interleavers. As is common, interleaving is added between concatenated codes to spread out burst errors. The main objective is to create embedded codewords such that, for a particular information block, the necessary protection is obtained via multiple channel encodings, contrary to the conventional methods which use
a single code rate per information block. In this way, a more flexible protection scheme is obtained. The information block size and concatenated coding rates are judiciously chosen to maximize system performance, subject to a total bit budget. In our system, we used various channel codes such as RCPC and semi random RC-LDPC codes with Viterbi, list-Viterbi and belief propagation decoding algorithms.

In many image coding systems, the overall system design is subject to a constraint of the channel code rate set being finite and discrete in practical communication scenarios. This finite set is usually created by puncturing a low rate mother code so that a single encoder-decoder pair is used. The main goal of creating embedded codewords is to effectively enlarge this code set by providing more protection levels than is possible using the code rate set directly. This study focuses on BSCs. In some situations, soft channel information is not available, and the only access to the data occurs after some form of hard decision decoding takes place. In this case, a BSC is a reasonable model for any memoryless channel with a binary input such as binary input Gaussian channel.

The remainder of this chapter is organized as follows: In Section 3.1, related works and the contributions of this chapter are presented. In Section 3.2, the proposed formatting is explained and the problem formulation is introduced. In Section 3.3, performance as well as the algorithm design is addressed, and the optimization framework is discussed. A low complexity descent search algorithm is given as an alternative to the overall optimization problem. We also compare the decoding complexity of the proposed design to previous coding architectures. Section 3.4 presents
some of the numerical results using different sets of channel codes. We derive two conditions that restrict the search space of possible channel code rates. These two conditions lead to significant complexity reduction of the optimization process. Finally, a brief summary and conclusions are presented in Section 3.5.

3.1 Related Work and Contributions

UEP using forward error correction (FEC) coding of progressive sources has been extensively studied in the literature. Early studies include [10] where CRC are cascaded with FEC coding to protect progressively coded images using SPIHT. This was shown to be more robust than previous image encoding results for BSCs. Error propagation is avoided by stopping the decoding when the first CRC failure is detected. Later, [9] used a similar idea at the packet level to provide UEP by using non-uniform channel coding throughout the bit stream. This permits a more flexible coding scheme to be achieved. RCPC codes are also considered by applying different channel code rates for different types of bits (e.g., sign bits, significance bits, etc) of SPIHT encoded data [16]. More powerful codes have also been used for the transmission of embedded sources. In particular, turbo codes are considered in [11] and [12], IRA codes in [15], and LDPC codes in [13] and [14]. Some of the gains reported in [11], [12], [15], [13] and [14] compared to [10], [9] and [16] can be attributed to the superiority of capacity-achieving codes over conventional coding schemes rather than strictly attributing the gains to the manner in which the FEC
is deployed. Also, those studies use larger block sizes to exploit the “asymptotically good” performance of LDPC or IRA codes compared to studies that use convolutional codes. In other studies, RS codes are utilized for graceful degradation of image quality in packet erasure channels [17]. One of the constraints of these studies is that they use a finite discrete channel code set usually in the form of a mother code and an associated puncturing pattern. Therefore, only a predetermined number of protection levels are possible with this finite set. Concatenated coding can loosen this constraint significantly and provide a more flexible system.

Concatenated coding was introduced in [50], where the total bit stream is encoded first with the inner code, and then the coded bit stream is further encoded with the outer code. This constructs a code that has an exponentially decreasing error probability with increasing block length [50]. It is also shown to be particularly effective in bursty environments. A typical use of concatenated codes can be found in space communications and usually involves a convolutional inner code and RS outer code. Concatenated coding can also be used for the UEP of progressive sources. In particular, product codes were used across the transmitted packets as a two dimensional code structure to provide UEP for progressive sources in fading channels [51]. Later, product codes were shown to be very effective in transmission of JPEG2000 [43] coded bit streams, especially in correlated Rayleigh fading using turbo codes and RS codes [53], [54]. The main objective of those studies is to use concatenated coding with RS coding to help improve system performance against packet erasures due to channel fading given a bandwidth constraint.
Besides the choice of channel coding, packet formatting, or packetization, is another important parameter in a FEC-based overall system design for progressive source transmission. Packetization is typically done either by fixing the size of the information block [10], [56] or by fixing the total number of coded bits [11], [12]. These packet formats are shown in Fig. 3.1 and named *FixedInfo* and *FixedCoded* formats hereafter. For both methods, rate-maximization and distortion-minimization problems for various channel coding choices are solved and fast algorithms are proposed for the former [57]. The performance of the overall system design can vary based on the choice of the channel code set and the associated packetization format.

The main contribution of this study is a novel packet formatting method along with a flexible system design for the transmission of embedded bit streams. By properly adjusting the channel codes, a flexible protection assignment strategy is achieved through the use of serially concatenated coding and random block interleavers. Subject to a total bit budget constraint, we obtained notable performance improvement over some of the conventional methods. Note that the proposed serial encoding scheme can be combined with RS codes similar to [51], [53] and [54] to create a two dimensional product code to help mitigate the correlation among the
random channel coefficients in correlated fading channels.

3.2 Proposed Formatting and System Model

Similar to previous studies [9], [10], in this study, error detection is achieved by appending a CRC to every information block $I_i, 1 \leq i \leq M$. Once an error is detected, all additional information bits are excluded from the reconstruction process.

*Figure 3.2*: Proposed formatting consists of $M$ stages of encoding as shown above.

Consider the proposed $M$-codeword scheme shown in Fig. 3.2 and assume that a discrete convolutional code set $C$ is chosen. First, we take $b_1$ source information bits (source block $I_1$) and derive two bytes of CRC\(^1\) ($N_c = 16$ bits) based on $b_1$ bits to be concatenated with $I_1$ for bit error detection. Also, $z$ zero tailing bits are appended to end the trellis at the all zero state. These bits together constitute

\(^1\)Here, a CRC polynomial is chosen that gives sufficiently small undetected-error probability [55] and the same CRC polynomial is used for all information blocks. The selected CRC polynomial is $X^{16} + X^{12} + X^5 + X$. 
the payload $P_1$ of the first codeword. The number of bits in the payload is $|P_1| = b_1 + N_c + z$. These bits are encoded using the channel code rate $r_1 \in C$ to produce the codeword $c_1$. In the next encoding stage, $c_1$ is concatenated with the second information block $I_2$, $N_c$ CRC bits and $z$ bits to produce the second payload $P_2$ where $|P_2| = \left\lceil \frac{(b_1 + N_c + z)}{r_1} \right\rceil + b_2 + N_c + z$ and $|I_2| = b_2$. In the second encoding stage, $N_c$ CRC bits are derived based on those $b_2$ bits. Since the errors out of a maximum likelihood sequence estimator (MLSE) are bursty, and convolutional codes show poor performance when channel errors are not independent and identically distributed (i.i.d.) [58], we use random block interleavers to break up the long burst noise sequences. We use $\pi(x)$ to denote the random block interleaving function that chooses a permutation table randomly, with a uniform distribution, and permutes the entries of $x$ bitwise based on this table. We choose the size of the random permutation table to be equal to the length of the payload size in each encoding stage except the first. After the interleaving, the bits in $\pi(P_2)$, are encoded using code rate $r_2 \in C$ to produce codeword $c_2$. This recursive process continues until we encode the last codeword $c_M$. The codeword $c_M$ is sent over the BSC channel.

The receiver obtains the noisy version ($\hat{c}_M$) of the codeword $c_M$. At the decoder, the sequential encoding operations of the encoding stage are performed in reverse order. In other words, the noisy received version $\hat{c}_M$ is decoded first using the ordinary Viterbi decoder. Then, the deinterleaver is invoked to obtain $\pi^{-1}(\pi(\hat{P}_M)) = \hat{P}_M$. The CRC check of the $M$th information block ($I_M$) is performed to label $I_M$ as useful or not for the reconstruction process. Thus, $I_M$ is associated
with a label and peeled off from $\hat{P}_M$. In the next decoding stage, $\hat{c}_{M-1}$ is decoded and deinterleaved in the same manner to get $\hat{P}_{M-1}$. Based on the CRC check, $I_{M-1}$ is determined to be useful or not in the reconstruction. The decoding operation is finalized after decoding the codeword $c_1$ and assigning a label to $I_1$. Assuming that the first label with a CRC failure is associated with $I_l$, then only the information blocks up to but not including the block $l$ are used to reconstruct the source.

### 3.3 Algorithm Design and Optimization

We define two sets, $R = \{r_1, \ldots, r_M\}$, $B = \{b_1, \ldots, b_M\}$ where $r_i \in C$ is the code rate used to protect the $i$th payload $P_i$. Those sets are optimized to minimize the distortion, as will be explained in subsection A. The length of codeword $n$ ($|c_n|$), $1 \leq n \leq M$ can be found as

$$|c_n| = \left( \prod_{i=1}^{n} r_j \right)^{-1} \left( b_i + N_c + z \right)$$

(3.1)

The length of the codeword $c_M$ equals the total allowed bit budget $B$ i.e., $|c_M| = B$. We send the total bit stream over a BSC channel with crossover error probability $\epsilon_0$, as shown in Fig. 3.3a. Because of the embedded nature of $M$ convolutional codes, the various test statistics are dependent, so a closed-form analysis does not appear to be feasible. Therefore, for the purpose of algorithm design, we will ignore the dependence and set up the algorithm assuming statistical independence.
The net result of this is that we approximate the sequence of data bit decisions at the output of each decoding stage as arising from a BSC, whereby the crossover probability of each BSC is functionally related to the crossover probability of the previous BSC, as well as the rate of the currently used channel encoder, as shown in Fig. 3.3b. Note that since our final results on performance are obtained by simulations, the dependencies that were ignored in the algorithm design will automatically be present in the performance results.

**Figure 3.3**: Each payload in the embedded bit stream is exposed to a BSC with different crossover probabilities.

After decoding the noisy codeword $\hat{c}_M$ and deinterleaving, we assume that the decoded bit stream will have a BER $\epsilon_1 < \epsilon_0$. In light of the previous paragraph, the decoded error rate $\epsilon_1$ is a function of $\epsilon_0$ and the channel code rate $r_M$, and is denoted as $\epsilon_1(\epsilon_0, r_M)$ hereafter. Next, the noisy codeword $\hat{c}_{M-1}$ is decoded, etc. After decoding, we argue in a similar way that the channel experienced by the
bits in $P_{M-1}$ can be modeled as a BSC with crossover probability $\epsilon_2(\epsilon_1, r_{M-1})$. In general, the channel that $P_i, \{1 \leq i \leq M\}$ experiences can be approximated as a BSC with crossover probability $\epsilon_{M-i+1}(\epsilon_{M-i}, r_i)$. Since $P_i \supseteq I_i$, the BER for $I_i$ is $\epsilon_{M-i+1}(\epsilon_{M-i}, r_i)$. In this study, the $\{\epsilon_{M-i+1}\}_{i=1}^{M}$ are found by simulating the system for a given channel raw error rate $\epsilon_0$ and code rates $r_i, r_{i+1}, \ldots, r_M$. As will be seen, the packet error rate (PER) appears in the expression for the expected distortion of the system. In principle, it is possible to determine the PERs directly by simulation. In practice, however, an information block can have any integer number of bits (up to the total length of the stream), and each such block size $b_i$ corresponds to a different PER ($PER_i$). Therefore in the optimization process, it would have been computationally intractable to find all possible PERs corresponding to all possible $b_i$ values. For this reason, we instead use the simulation to find the BER, and use that and the various $b_i$ values to compute the PERs. Having approximated each channel as a BSC with some crossover probability, we approximate the PER of the $i$-th information block ($I_i$) by [38]

$$PER_i \approx 1 - (1 - \epsilon_{M-i+1})^{b_i} \quad (3.2)$$

The described channel modeling is illustrated in Fig. 3.3b. Using Equation (3.2), the expected distortion of the progressive bit stream using the proposed
formatting and the independence assumption is approximated by [39]

\[
E[D] = \sum_{l=0}^{M} PER_{l+1} \prod_{i=0}^{l} (1 - PER_i) d_l = d_0 - \sum_{l=1}^{M} \prod_{i=1}^{l} (1 - PER_i) \Delta_l \quad (3.3)
\]

\[
\approx d_0 - \sum_{l=1}^{M} \Delta_l \prod_{i=1}^{l} (1 - \epsilon_{M-i+1})^{b_l}
\]

\[
\triangleq d_0 - D_M(B, R) \geq 0 \quad (3.4)
\]

where \(d_0\) is the distortion when the source decoder reconstructs the source without using any of the transmitted information, \(d_l\) is the distortion when the decoder decodes only \(\sum_{j=1}^{l} b_j\) information bits and \(\Delta_l = d_{l-1} - d_l \geq 0\). We also assume \(PER_0 = 0\) and \(PER_{M+1} = 1\) for completeness as the zeroth and \((M+1)\)th information blocks do not exist. Minimization of \(E[D]\) is equivalent to maximizing the average distortion reduction \(\overline{D}_M(B, R)\). Note that the above expression corresponds to expected distortion when all errors are assumed to have been caused by statistically independent events, as explained above. To help ensure that the errors are as uncorrelated as possible, we have introduced interleavers at the transmitter, and corresponding deinterleavers at the receiver before each decoding stage.

### 3.3.1 Optimization

For the proposed scheme, the design objective is to find \(M\), the rate allocation \(R\) and information block size set \(B\) such that \(\overline{D}_M(B, R)\) is maximized. In other words, we intend to find the set \(\{M^*, R^*, B^*\} = \arg \max_{\{M, R, B\}} \overline{D}_M(B, R)\) subject to bit budget constraint \(B\) where \(*\) denotes the optimal values. For a given \(M\), we need to optimize the overall parameter set \(\{B, R\} = \{b_1, \ldots, b_M, r_1, \ldots, r_M\}\) which is
subject to a total bit budget $B$. The problem can be stated as

$$\max_{B, R} \overline{D}_M(B, R), \text{ subject to } \sum_{i=1}^{M} \frac{1}{\prod_{j=i}^{M} r_j} (b_i + N_c + z) = B \quad (3.5)$$

There are only either 9 RC-LDPC or 13 RCPC code rates in our set $C$, so including the uncoded case we have either 10 or 14 possible values for $r_i$, whereas $b_i$ has a vastly larger set of possible values (in theory, $b_i$ can take on any value from 1 up to the total number of source bits). First, we assume a fixed channel code rate set $\mathcal{R}$ and optimize the information block size set $\mathcal{B}$ using a descent search algorithm [41] by initializing $\mathcal{B}^* = \{b, b, \ldots, b\}$ for some $b$ satisfying $Mb < B$.

This constrained optimization problem is reduced to solving an unconstrained optimization problem by using a line search method$^2$. We employ the gradient vector $\nabla \overline{D}_M(\mathcal{B}, \mathcal{R})$ as the descent direction for fixed $M$ and $\mathcal{R}$. Finally, we note that the elements of the set $\mathcal{B}$ can only take discrete values (the unit is in bits). So the components of the gradient vector are not analytically available. We use the “gradients by forward finite differences” as our approximation, as given in [41, Section 2.3.1.5]. Then, we carry out this optimization procedure for all possible channel code rates. Finally, for a given $M$, we choose the set $(\mathcal{B}^*, \mathcal{R}^*) = \text{arg} \left\{ \max_{\mathcal{R}} \left\{ \max_{\mathcal{B}} \overline{D}_M(\mathcal{B}, \mathcal{R}) \right\} \right\}$ to be the optimal point. For ease of reference, we present a functional flow diagram of the optimization procedure in Fig. 3.4 to determine the optimum parameters $(M^*, \mathcal{B}^*, \mathcal{R}^*)$.

---

$^2$The line search algorithms are summarized in [41, Section 2.1.1]. We used the golden section method (explained in [41, Section 2.2.1]) as our one dimensional line search approach. We found that a sufficient number of iterations for convergence is almost linear in $M$. The complexity of the optimization problem will also be discussed in simulations section.
Choose \( R = \{ r_1, \ldots, r_M \} \)

Solve (6) for \( B^* \)

Compute \( \mathbb{E}[D] \) given in (4)

\begin{align*}
D_0 &= \infty \\
M &= 1 \\
D_0 &= \infty
\end{align*}

\begin{align*}
\text{BSC crossover probability } p_{\text{BSC}} \\
\text{PER estimation} \\
\text{Choose } R &\Rightarrow \text{Solve (6) for } B^* \\
\text{Compute } \mathbb{E}[D] &\text{ given in (4)}
\end{align*}

\begin{align*}
D_0 &= \text{computed distortion} \\
\text{Is } D_0 &< D_0\text{-th computed distortion} \\
\text{NO} \\
\text{Change } R, \\
\text{no longer decrease } D_0\text{.} \\
\text{YES} \\
M &= M+1 \\
\text{Is } D_n &< D_{n-1}\text{?} \\
\text{NO} \\
\text{Stop}
\end{align*}

**Figure 3.4**: Functional flow diagram of the proposed optimization and transmission scheme.
Let us fix the code rate set \( \mathcal{R} = \{ r_i \}_{i=1}^M \subset \mathcal{C} \). First, we realize that \( \Delta_1 = d_0 - d_1 \) depends on \( b_1 \) because \( d_0 \) is a fixed real number. Similarly, \( \Delta_2 = d_1 - d_2 \) depends on both \( b_1 \) and \( b_2 \). Thus, \( \Delta_l = d_{l-1} - d_l \) depends on \( b_1, b_2, \ldots, b_l \). To signify the functional dependence, they will be denoted as \( \Delta_1(b_1), \Delta_2(b_1, b_2) \) and \( \Delta_l(b_1, b_2, \ldots, b_l) \) (see Fig. 3.5). For this general case, our constraint equation given by Equation (3.5) becomes

\[
\sum_{i=1}^{M} \frac{b_i}{\prod_{j=i}^{M} r_j} = B - (N_c + z) \sum_{i=1}^{M} \left( \prod_{j=i}^{M} r_j \right)^{-1}
\]  
(3.6)

Letting \( \hat{B} \triangleq B - (N_c + z) \sum_{i=1}^{M} \left( \prod_{j=i}^{M} r_j \right)^{-1} \), we have for the \( l \)th information block

\[
b_l = \left( \hat{B} - \sum_{i=1, i \neq l}^{M} \frac{b_i}{\prod_{j=i}^{M} r_j} \right) \prod_{j=l}^{M} r_j.
\]  
(3.7)

This means that one of the parameters, \( b_l \), can be expressed in terms of the other parameters. For example, using Equation (3.7), \( b_M = r_M \hat{B} - r_M \sum_{i=1}^{M-1} \frac{b_i}{\prod_{j=i}^{M} r_j} \), and

\[
\Delta_M \left( b_1, b_2, \ldots, b_{M-1}, r_M \left( \hat{B} - \sum_{i=1}^{M-1} \frac{b_i}{\prod_{j=i}^{M} r_j} \right) \right)
\]  
(3.8)

is now only a function of \( b_1, \ldots, b_{M-1} \). Thus, using Equation (3.8) and (3.4), we
obtain
\[
\overline{D}_M \left( \left\{ b_1, \ldots, b_{M-1}, r_M \hat{B} - r_M \sum_{i=1}^{M-1} \frac{b_i}{\prod_{j=i}^{M} r_j} \right\}, R \right) \\
= \sum_{i=1}^{M-1} \Delta_i (b_1, \ldots, b_l) \prod_{i=1}^{l} (1 - \epsilon_{M-i+1})^{b_i} \\
+ \Delta_M \left( b_1, b_2, \ldots, b_{M-1}, r_M \left( \hat{B} - \sum_{i=1}^{M-1} \frac{b_i}{\prod_{j=i}^{M} r_j} \right) \right) \\
\times (1 - \epsilon_1) \left( \hat{B} - \sum_{i=1}^{M-1} \frac{b_i}{\prod_{j=i}^{M} r_j} \right) \prod_{i=1}^{M-1} (1 - \epsilon_{M-i+1})^{b_i} \tag{3.9}
\]

which is only a function of \{b_1, \ldots, b_{M-1}\}. The constrained optimization problem in Equation (6) becomes unconstrained and has now \(M-1\) parameters subject to optimization. We use the gradient descent algorithm to find a maximum \{b_1^*, \ldots, b_{M-1}^*\} for this problem [41]. Finally, we calculate \(b_M^* = r_M \hat{B} - r_M \sum_{i=1}^{M-1} \frac{b_i^*}{\prod_{j=i}^{M} r_j} \).

A simple example that is easier to visualize (the concavity of \(\overline{D}_M(B, R)\) for a

![Figure 3.5](image)

**Figure 3.5**: Distortion versus number of source bits illustrating \(\Delta_1, \Delta_2, \ldots, \Delta_l\) as functions of \(b_1, b_2, \ldots, b_l\).
given $\mathcal{R}$) occurs when $M = 2$. In this particular case, $\Delta_2(b_1, b_2) = \Delta_2(b_1, r_2\hat{B} - \frac{b_1}{r_1})$
is now only a non-increasing function of $b_1$. After absorbing the constraint equality
into the cost function, we will have an unconstrained optimization problem with the
following cost function to maximize:

$$
\bar{D}_2 \left( \{ b_1, r_2\hat{B} - \frac{b_1}{r_1} \}, \mathcal{R} \right) = (1 - \epsilon_2(\epsilon_0, r_2), r_1))^{b_1} \Delta_1(b_1)
+ (1 - \epsilon_2(\epsilon_1(\epsilon_0, r_2), r_1))^{b_1}(1 - \epsilon_1(\epsilon_0, r_2))^{r_2 \hat{B} - \frac{b_1}{r_1}} \Delta_2 \left( b_1, r_2\hat{B} - \frac{b_1}{r_1} \right).
$$

(3.10)

Several examples are shown in Fig. 3.6 for different values of $\epsilon_0$ and $r_{tr}$, where we
take $B = 0.3 \times 512 \times 512$ bits for a $512 \times 512$ Lena image using the RCPC code set
with memory 6 from [30].

### 3.3.2 Concatenated Coding with LDPC Codes

The concatenated coding mechanism presented in this paper can be used
with any type of error correction code, i.e., we can employ a class of “asymptotically
good” codes such as LDPC codes. However, there might be some minor changes in
the proposed design when it is used with LDPC codes. For example, we would not
need to consider the CRC codes in our design because of the inherent error detection
property of LDPC codes [59]. On the other hand, using LDPC codes increases both
the computational complexity and memory requirements of the encoder. In the
proposed design, the length of the codewords increases after each encoding stage.
To enable linear encoding complexity, we resort to the semi-random LDPC encoding
Figure 3.6: For several different sets $\mathcal{R}$, $\epsilon_0$ and $r_{ir}$, $D_2 \left( \left\{ b_1, r_2 \hat{B} - \frac{b_1}{r_1} \right\}, \mathcal{R} \right)$ is plotted for a SPIHT source encoder. As can be seen, it is a concave function of $b_1$ and has a single maximum point.
Table 3.1: Decoding thresholds for the semi-random rate compatible punctured LDPC codes.

<table>
<thead>
<tr>
<th>Code Rate</th>
<th>Decoding threshold ($\epsilon^*$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4/5</td>
<td>0.0239</td>
</tr>
<tr>
<td>8/11</td>
<td>0.041</td>
</tr>
<tr>
<td>2/3</td>
<td>0.057</td>
</tr>
<tr>
<td>8/13</td>
<td>0.079</td>
</tr>
<tr>
<td>8/15</td>
<td>0.097</td>
</tr>
<tr>
<td>1/2</td>
<td>0.111</td>
</tr>
<tr>
<td>4/9</td>
<td>0.131</td>
</tr>
<tr>
<td>2/5</td>
<td>0.1399</td>
</tr>
<tr>
<td>4/11</td>
<td>0.1492</td>
</tr>
</tbody>
</table>

structure given in [60]. We optimize the system using the cost function in which the approximate PER values are found using the analysis of LDPC codes given in [61] and the algorithm presented in [62], for finding the decoding thresholds of the semi-random RC-LDPC code family. These decoding thresholds for the code set \{8/22, 8/20, 8/18, 8/16, 8/15, 8/13, 8/12, 8/11, 8/10\} are given in Table 3.1.

### 3.3.3 Complexity of the Proposed Scheme

As far as the encoding of the convolutional codes is concerned, the encoders use the same amount of memory elements both in the proposed system and in the comparison systems of [10], [9] and [56]. Although convolutional encoding has a minor effect on the overall complexity, the Viterbi decoding is a more complex operation. We focus on the decoding complexity at the receiver.
For a fixed constraint length, the decoding complexity of punctured convolutional codes increases linearly with the length of the codeword [63]. Also, the maximum trellis complexity of a random block interleaver increases linearly with the length of the interleaver [64]. On the other hand, since we use the LDPC encoding structure given in [60] and Max-product algorithm at the receiver, computation complexity of both LDPC encoding and decoding is linear with increasing block length. If we assume that the complexity of deinterleaving is minor compared to the decoding operation, the maximum trellis complexity of the decoding procedure of the proposed scheme is found to be on average at most $M$ times more complex than the decoding operations of the systems in [10], [9], [14] and [56].

3.4 Numerical Results and Discussions

We use two channel code sets: a convolutional code set that consists of the 13 RCPC codes with memory 6 found in [30] and used in [10], and an LDPC code set that consists of the 9 RC-LDPC codes used in [14]. The convolutional code is treated as a block code by using zero-tail padding with $z = 6$ zeros at the end of each information block.

3.4.1 Simulations with RCPC Codes

Throughout this subsection, the RCPC channel code set is used. First, we show some simulation results that illustrate the role of the interleavers in our system.
design. We set \( M = 2 \) and choose two pairs of codes \((r_1, r_2) : (2/3, 1/2), (4/9, 1/2)\) and information block size \((b_1, b_2) : (10^4, 10^4)\). In Fig. 3.7, the decoded BER is plotted for the bits in \( \mathcal{I}_1 \) as a function of the crossover probability \( \epsilon_0 \) with two different permutation table sizes: \(|P_1| = 1500\) and \(|P_1| \approx 10^4\). As expected, if we increase the size of the random permutation table we get better BER performance. Also shown in the same figure is the decoded BER for the bits in \( \mathcal{I}_2 \) (single code rate 1/2). The curve named non-interleaved shows that when there is no interleaver, we obtain very minor performance improvement over the single code rate scheme. This is because correlated error patterns at the output of the Viterbi decoder significantly affect the performance of the following decoding stage. At higher crossover error probabilities, the decoded BER for \( \mathcal{I}_1 \) is worse than for the single code rate scheme. To explain this, first observe in Fig. 3.7a that towards the righthand edge of the plot (roughly \( \epsilon_0 > 0.1 \)), the outer decoder is overwhelmed by the high raw channel error rate. As we move to the left, the outer decoder is not overwhelmed. However, the decoded BER of the outer decoder has to be low enough so that the inner encoder is not overwhelmed. Finally, towards the lefthand edge of the plot (roughly \( \epsilon_0 < 0.09 \)), neither decoder is overwhelmed and give lower decoded BERs than \( \epsilon_0 \).

Finally, we obtained the curve denoted i.i.d. Assump. in Fig. 3.7 in the following way: for any given raw channel error rate \( (\epsilon_0) \), we first decoded the outer code \( r_2 \) and obtained the decoded BER \( \epsilon_2 \) by simulation. After finding \( \epsilon_2 \), we sent the inner codeword (information block encoded with \( r_1 \)) through a BSC with crossover probability \( \epsilon_2 \) and obtained the decoded BER \( \epsilon_1 \) by simulation again (argument
Figure 3.7: The effect of interleaving with different size of the random permutation tables.
given in Section III). The curve \textit{i.i.d Assump.} given in Fig. 3.7 is the set of $\epsilon_1$s for different values of $\epsilon_0$ shown in the abscissa. It is verified that an interleaver depth of $|P_1|$ using a block length of $10^4$ gives a very close approximation to the i.i.d. assumption.

In the rest of the simulations, we use an embedded bit stream and we use our protection scheme to send it over a BSC. We compare the following transmission schemes based on embedded bit streams:

- \textit{ShZeg}: This is based on [10]. It uses a single optimal code with \textit{FixedInfo} formatting. The information block size is 200 bits and a single optimal code rate is chosen from the code set that minimizes the average distortion for all the transmission rates in consideration. Also, the list Viterbi decoder (LVA) of [10] is replaced with the ordinary Viterbi algorithm.

- \textit{NosLu}: This is the approach presented in [9]. It uses an optimal code rate for each information block with \textit{FixedInfo} formatting [39]. The information blocks have size of 202 bits.

- \textit{Product Code}: This system was originally proposed in [51] for a correlated fading channel, but the effectiveness of their scheme was shown for a BSC as well. It uses RS codes in concatenation with convolutional codes as a two dimensional code structure.

- \textit{Concatenated}: This is the proposed concatenated coded system with related optimizations as defined.
We produced the embedded bit stream by encoding the grayscale 512 × 512 Lena, Barbara and GoldHill images using the SPIHT with arithmetic coding and JPEG2000 Part 1 (with no error correction capability) progressive image coders. Due to space limitations, we only present the results of Lena encoded using SPIHT with arithmetic coding and Goldhill encoded with JPEG2000. Other combinations exhibited similar results. We tested two crossover error probabilities: \( \epsilon_0 = 0.01 \) and \( \epsilon_0 = 0.05 \). We define total transmission rate \( r_{tr} \) in bits per pixel (bpp) for a given \( L_x \times L_y \) image as \( r_{tr} = \frac{B}{L_x \times L_y} \). Quality assessment of the decoded images are given in terms of average PSNR. PSNR results for all the systems are shown as a function of \( r_{tr} \) in bpp, where

\[
PSNR = 10 \log_{10} \left( \frac{MAX_I^2}{\mathbb{E}[D]} \right)
\]  

(3.11)

and \( MAX_I = 255 \) is the maximum possible intensity value of the image.

For a given BSC channel with crossover probability \( \epsilon_0 \) and sets \( \mathcal{B} \) and \( \mathcal{R} \), we produce total bit stream i.e., \( c_M \), as explained in Section 3.3 and send it through the BSC. We obtain the \( \{\epsilon_i\}_{i=1}^M \) by simulating \( M \) sequential decoding stages. Note that, given the sets \( \mathcal{B} \) and \( \mathcal{R} \), the sizes of the random permutation tables are automatically determined. We should also notice that the \( \{\epsilon_i\}_{i=1}^M \) are assumed to be independent of the \( \{b_i\}_{i=1}^M \), as the decoded BER of RCPC codes changes only slightly with varying block size [65]. Finally, we calculate \( \mathbb{E}[D] \) using the approximations given in (3.2) and (3.4). After optimization, we find the optimal parameters to be used in the proposed scheme. Finally, using the optimal set of parameters \( (M^*, \mathcal{B}^*, \mathcal{R}^*) \), we simulate the
system to find the expected distortion and convert to PSNR using (3.11). As seen in Fig. 3.8, the proposed system is more effective at higher channel error rates and higher transmission rates.

Another observation is that, although NosLu optimizes channel code rates for each information block, because of the flat region of the R-D curve of the source, NosLu does not improve much over ShZeg at higher transmission rates. However, at smaller transmission rates, NosLu gives around 0.35dB gain over the ShZeg scheme as reported in [9]. Also note that [56] does not use the approximation in [9] and finds the optimal rate schedule using dynamic programming instead of Lagrangian multipliers. However, we experimentally observed that the performance loss due to the approximation in [9] is minor. Product Code for the Lena image gives around 0.47dB gain over ShZeg for all transmission rates. The advantage of Product Code is its two dimensional coding structure that provides the same level of protection of ShZeg using an overall higher code rate (RCPC + RS). This allows more information content in the bit budget and the potential for performance improvement. Concatenated provides more than 1dB gain over ShZeg and 0.5dB gain over Product Code at all transmission rates. The advantage of Concatenated can be attributed in part to packet size adjustment (and thus PER adjustment) and its flexible protection by multiple channel codes. Using multiple code rates enables protection levels that are not possible using the discrete code set. We have tested various information block sizes of 100, 200, 400, 600, 800 and 1000, and verified that choosing other than 200 bits as the information block size of conventional methods does not help their
Figure 3.8: Different systems are compared over a BSC with BER=0.01 and BER=0.05 using 512×512 Lena image.
Figure 3.9: Different systems are compared over a BSC with BER=0.01 and BER=0.05 using 512 × 512 Goldhill image.
performance much. In Fig. 3.9, we show average PSNR performances of some of those systems using Goldhill encoded with JPEG2000. The results of Concatenated exhibit similar gains compared to ShZeg i.e., they are more significant at lower $\epsilon_0$ and higher transmission rates. The jaggedness of the curves is due to the discrete code set as well as the source encoder.

Next, we provide and compare results for Concatenated using Lena at various values of $M$, as shown in Table 3.2. There are two observations regarding the proposed formatting and channel code rate allocation of Concatenated. First, when $M = m \in \{1, 2, 3, 4, 5\}$, we have $r_1^*(m) \geq r_2^*(m) \geq \cdots \geq r_m^*(m)$ (We refer to this as the first (1st) condition), where the superscript denotes the current value of $M$. This is because the baseline protection is achieved by the outermost code, and then incremental protection is added by each encoding stage. Our results show that the incremental protection is enough to obtain the necessary UEP with a less powerful outermost code. Note that this result can be used to restrict the search space of our exhaustive search in finding the optimal code rate set $\mathcal{R}^*$. It leads to complexity reductions as shown in Table 3.3. Secondly, if we consider two different values for $m$, namely $m_1$ and $m_2 = m_1 + 1$, we observe that $r_{m_2}^*(m_2) \geq r_{m_1}^*(m_1)$ (We refer to this as the second (2nd) condition). This is because as we increased $M$, we observe that the outermost code rate did not need to be more powerful than the one used in the system with one less encoding stage, since going from $m_1$ to $m_2$ encoding stages gives a certain level of additional protection for more important bits. Therefore, it is enough to choose the outermost code $r_{m_2}^*(m_2)$ to be equal to or greater than
Table 3.2: Optimal allocation: $R^*$ for $M$-packet transmission. Optimal parameter $(M^*)$ is shown in bold.

<table>
<thead>
<tr>
<th>$M$</th>
<th>$r_1^*$</th>
<th>$r_2^*$</th>
<th>$r_3^*$</th>
<th>$r_4^*$</th>
<th>$r_5^*$</th>
<th>PSNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1/4</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>14.48</td>
</tr>
<tr>
<td>2</td>
<td>4/7</td>
<td>1/4</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>26.96</td>
</tr>
<tr>
<td>3</td>
<td>8/9</td>
<td>2/3</td>
<td>1/4</td>
<td>N/A</td>
<td>N/A</td>
<td>27.32</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td>8/9</td>
<td>2/3</td>
<td>4/15</td>
<td>N/A</td>
<td>27.38</td>
</tr>
<tr>
<td>5</td>
<td>1</td>
<td>1</td>
<td>8/9</td>
<td>2/3</td>
<td>4/15</td>
<td>27.37</td>
</tr>
</tbody>
</table>

| $\epsilon_0$=0.15, $r_{tr}$ = 0.6 bpp |
|-----|--------|--------|--------|--------|--------|---------|
| 1   | 1/4    | N/A    | N/A    | N/A    | N/A    | 20.44   |
| 2   | 2/3    | 1/3    | N/A    | N/A    | N/A    | 28.45   |
| 3   | 8/9    | 4/5    | 4/13   | N/A    | N/A    | 28.71   |
| 4   | 8/9    | 8/9    | 4/5    | 1/3    | N/A    | 28.79   |
| 5   | 1      | 8/9    | 8/9    | 4/5    | 1/3    | 28.75   |

| $\epsilon_0$=0.05, $r_{tr}$ = 0.6 bpp |
|-----|--------|--------|--------|--------|--------|---------|
| 1   | 1/4    | N/A    | N/A    | N/A    | N/A    | 31.65   |
| 2   | 8/9    | 1/3    | N/A    | N/A    | N/A    | 32.72   |
| 3   | 8/9    | 4/5    | 4/9    | N/A    | N/A    | 33.10   |
| 4   | 8/9    | 8/9    | 8/9    | 4/9    | N/A    | 33.22   |
| 5   | 1      | 8/9    | 8/9    | 8/9    | 4/9    | 33.18   |

| $\epsilon_0$=0.01, $r_{tr}$ = 0.7 bpp |
|-----|--------|--------|--------|--------|--------|---------|
| 1   | 2/5    | N/A    | N/A    | N/A    | N/A    | 34.60   |
| 2   | 8/9    | 4/7    | N/A    | N/A    | N/A    | 35.77   |
| 3   | 8/9    | 8/9    | 2/3    | N/A    | N/A    | 36.07   |
| 4   | 1      | 8/9    | 8/9    | 2/3    | N/A    | 36.08   |
| 5   | 1      | 1      | 8/9    | 8/9    | 2/3    | 36.06   |

| $\epsilon_0$=0.005, $r_{tr}$ = 0.5 bpp |
|-----|--------|--------|--------|--------|--------|---------|
| 1   | 1/2    | N/A    | N/A    | N/A    | N/A    | 33.80   |
| 2   | 8/9    | 2/3    | N/A    | N/A    | N/A    | 34.90   |
| 3   | 1      | 8/9    | 2/3    | N/A    | N/A    | 34.97   |
| 4   | 1      | 1      | 8/9    | 2/3    | N/A    | 34.97   |
| 5   | 1      | 1      | 1      | 8/9    | 2/3    | 34.96   |

| $\epsilon_0$=0.001, $r_{tr}$ = 0.3 bpp |
|-----|--------|--------|--------|--------|--------|---------|
| 1   | 2/3    | N/A    | N/A    | N/A    | N/A    | 33.02   |
| 2   | 8/9    | 4/5    | N/A    | N/A    | N/A    | 33.43   |
| 3   | 1      | 8/9    | 4/5    | N/A    | N/A    | 33.49   |
| 4   | 1      | 1      | 8/9    | 4/5    | N/A    | 33.48   |
| 5   | 1      | 1      | 1      | 8/9    | 4/5    | 33.45   |
Table 3.3: Number of possibilities with and without the first condition. We obtain complexity reductions with increasing $M$. For example when $M = 5$, it is enough to search only $1/62$ of all possibilities.

<table>
<thead>
<tr>
<th>$\forall \epsilon_0, \forall r_{tr}$</th>
<th>$m = 1$</th>
<th>$m = 2$</th>
<th>$m = 3$</th>
<th>$m = 4$</th>
<th>$m = 5$</th>
<th>$m = 6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>No condition</td>
<td>14</td>
<td>196</td>
<td>2744</td>
<td>38416</td>
<td>537824</td>
<td>7529536</td>
</tr>
<tr>
<td>1$^{st}$ condition</td>
<td>14</td>
<td>105</td>
<td>560</td>
<td>2380</td>
<td>8568</td>
<td>27132</td>
</tr>
</tbody>
</table>

Table 3.4: Number of possibilities with two conditions when $\epsilon_0 = 0.01$, $r_{tr} = 0.7$bpp. We obtain more complexity reductions with increasing $M$. For example when $M = 5$, it is enough to search only $1/3055$ of all possibilities.

<table>
<thead>
<tr>
<th>$\epsilon_0 = 0.01$, $r_{tr} = 0.7$bpp</th>
<th>$m_1 = 1$</th>
<th>$m_1 = 2$</th>
<th>$m_1 = 3$</th>
<th>$m_1 = 4$</th>
<th>$m_1 = 5$</th>
<th>$m_1 = 6$</th>
</tr>
</thead>
<tbody>
<tr>
<td>No condition</td>
<td>14</td>
<td>196</td>
<td>2744</td>
<td>38416</td>
<td>537824</td>
<td>7529536</td>
</tr>
<tr>
<td>1$^{st}$ condition</td>
<td>14</td>
<td>105</td>
<td>560</td>
<td>2380</td>
<td>8568</td>
<td>27132</td>
</tr>
<tr>
<td>1$^{st}$ $\land$ 2$^{nd}$ condition</td>
<td>14</td>
<td>50</td>
<td>85</td>
<td>120</td>
<td>176</td>
<td>260</td>
</tr>
</tbody>
</table>

$r_{m_1}^*$ to provide the baseline protection for the total bit stream. Having a larger outermost code rate also allows more source bits into the bitstream and increases the potential for performance improvement. However, as we increase $M$, the CRC and tailing bits will start to overwhelm the allowed bit budget, giving rise to performance degradation due to lack of source bits. Using the condition $r_{m_2}^* (m_2) \geq r_{m_1}^* (m_1)$, we can obtain additional reductions in complexity, especially at low $\epsilon_0$. One particular example is shown in Table 3.4.

Table A.1 also shows the average PSNR performance of Concatenated as a function of $M$. For all the cases shown, we pick up more than 0.5 dB gain using the optimal value $M^*$. For $1 \leq M \leq M^*$, we get diminishing returns as we approach $M^*$. When $M > M^*$, there can be a slight degradation in the performance, because
Table 3.5: PSNR in dB for 512 × 512 images transmitted over BSC at various transmission rates. Ordinary Viterbi decoder is replaced with an LVA.

<table>
<thead>
<tr>
<th>$r_t$ (bpp)</th>
<th>Image</th>
<th>System</th>
<th>Channel raw BER ($\epsilon_0$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.1</td>
</tr>
<tr>
<td>0.252</td>
<td>Lena</td>
<td>Proposed</td>
<td>29.46</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$10\left[ \begin{smallmatrix} 9 \ 51 \end{smallmatrix} \right]$</td>
<td>28.4 (28.61, 28.88)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>31.29</td>
</tr>
<tr>
<td>0.505</td>
<td>Lena</td>
<td>Proposed</td>
<td>32.33</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$10\left[ \begin{smallmatrix} 9 \ 51 \end{smallmatrix} \right]$</td>
<td>31.1 (31.27, 31.6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>34.39</td>
</tr>
<tr>
<td>0.994</td>
<td>Lena</td>
<td>Proposed</td>
<td>35.27</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$10\left[ \begin{smallmatrix} 9 \ 51 \end{smallmatrix} \right]$</td>
<td>34.2 (34.25, 34.66)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>37.39</td>
</tr>
<tr>
<td>0.252</td>
<td>Goldhill</td>
<td>Proposed</td>
<td>27.43</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$10$</td>
<td>26.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>28.63</td>
</tr>
<tr>
<td>0.505</td>
<td>Goldhill</td>
<td>Proposed</td>
<td>29.24</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$10$</td>
<td>28.6</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>30.78</td>
</tr>
<tr>
<td>0.994</td>
<td>Goldhill</td>
<td>Proposed</td>
<td>31.41</td>
</tr>
<tr>
<td></td>
<td></td>
<td>$10$</td>
<td>30.7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>33.4</td>
</tr>
</tbody>
</table>
Table 3.6: PSNR in dB for $512 \times 512$ Lena and Goldhill images (JPEG2000). Bold denotes the winning scheme. ConRCPC uses an RCPC code family with ordinary Viterbi decoding and ConLDPC uses an RC-LDPC code family with a belief propagation algorithm.

<table>
<thead>
<tr>
<th>Image</th>
<th>$r_{tr}$ (bpp)</th>
<th>System</th>
<th>Channel raw BER ($\epsilon_0$)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>0.01</td>
</tr>
<tr>
<td>Lena</td>
<td>0.252</td>
<td>ConRCPC</td>
<td>31.39</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ConLDPC</td>
<td><strong>32.83</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RC-LDPC [14]</td>
<td>32.77</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IRA [15]</td>
<td>32.75</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RCTC [11]</td>
<td>32.56</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>33.59</td>
</tr>
<tr>
<td></td>
<td>0.505</td>
<td>ConRCPC</td>
<td>34.47</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ConLDPC</td>
<td><strong>36.12</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RC-LDPC [14]</td>
<td>36.08</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IRA [15]</td>
<td><strong>36.18</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RCTC [11]</td>
<td>35.67</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>36.82</td>
</tr>
<tr>
<td></td>
<td>0.994</td>
<td>ConRCPC</td>
<td>37.53</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ConLDPC</td>
<td>39.01</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RC-LDPC [14]</td>
<td><strong>39.03</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>IRA [15]</td>
<td>38.93</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RCTC [11]</td>
<td>38.78</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>39.84</td>
</tr>
<tr>
<td>Goldhill</td>
<td>0.252</td>
<td>ConRCPC</td>
<td>28.77</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ConLDPC</td>
<td>29.74</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IRA [15]</td>
<td>29.74</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RCTC [11]</td>
<td>29.64</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>30.30</td>
</tr>
<tr>
<td></td>
<td>0.505</td>
<td>ConRCPC</td>
<td>30.88</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ConLDPC</td>
<td><strong>32.21</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>RC-LDPC [14]</td>
<td><strong>32.28</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>32.84</td>
</tr>
<tr>
<td></td>
<td>0.994</td>
<td>ConRCPC</td>
<td>33.63</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ConLDPC</td>
<td><strong>35.30</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>IRA [15]</td>
<td>35.00</td>
</tr>
<tr>
<td></td>
<td></td>
<td>RCTC [11]</td>
<td>34.81</td>
</tr>
<tr>
<td></td>
<td></td>
<td>“error-free case”</td>
<td>36.06</td>
</tr>
</tbody>
</table>
with larger $M$, the optimization results in a code rate of unity for some of the initial information blocks. This leads to an increased number of CRC bits which, in turn, decreases the total number of source and channel coding bits in the allowed bit budget. In addition, since the size of the permutation tables are chosen to be equal to the payload size in this study, smaller information blocks mean shorter permutation table size, which significantly impact the overall performance of Concatenated.

Finally, we compare the proposed scheme with some of the reported results found in [10], [9] and [51]. As can be seen, those studies use the LVA for increased performance. Therefore, we use our proposed scheme with LVA for comparison. In other words, the ordinary Viterbi algorithm is replaced with LVA in the proposed formatting and system model. LVA finds the “best path” which has the smallest accumulated metric in the trellis and satisfies the CRC check at the same time. We constrained the path search depth to 70 candidate paths. If none of the 70 candidate paths satisfies the CRC, decoding failure is declared. Table 3.5 shows some results for Lena and Goldhill images for $\epsilon_0 = 0.1$ and $\epsilon_0 = 0.01$.

As a perspective on PSNR, if one set the information rate equal to that which corresponds to the capacity of a BSC with crossover probability $\epsilon_0$, and assume that the data is sufficiently coded such as to produce no errors, then the results corresponding to this idealized scenario are listed in the rows entitled “error-free case”. Note that the capacity of a BSC is given by $C(\epsilon_0) = 1 - h(\epsilon_0)$, where $h(\epsilon_0) = -\epsilon_0 \log_2(\epsilon_0) - (1 - \epsilon_0) \log_2(1 - \epsilon_0)$ is the binary entropy function and $\epsilon_0$ is the raw error rate of the channel. Also note that “error-free case” corresponds
to \( \{0.252, 0.505, 0.994\} \times C(\epsilon_0)|_{\epsilon_0=0.1} \approx \{0.1338, 0.2682, 0.5278\} \) bpp and \( \{0.252, 0.505, 0.994\} \times C(\epsilon_0)|_{\epsilon_0=0.01} \approx \{0.2316, 0.4642, 0.9137\} \) bpp. As can be seen, the proposed scheme not only improves the performance of [10] but also gives results that are close to “error-free case”.

### 3.4.2 Simulations with RC-LDPC Codes

Throughout this subsection, the RC-LDPC channel code set is used. An extra byte (instead of the CRC) is added in each encoding stage to inform the RC-LDPC decoder about the channel coding rate used, as in [14]. Our mother code rate is 8/13, and through puncturing and extending the mother code [52], we obtain 9 code rates in total: \( \{8/22, 8/20, 8/18, 8/16, 8/15, 8/13, 8/12, 8/11, 8/10\} \). This yields similar simulated PER performance to that of the irregular RC-LDPC construction in [14] for a BSC. In what follows, we present the performance of the proposed coding scheme using the Lena and Goldhill images (encoded with JPEG2000) with the class of RCPC codes (\( \text{ConRCPC} \)) and RC-LDPC codes (\( \text{ConLDPC} \)). The results are given in Table 3.6 using a maximum of 250\(^3\) iterations of the Max-product algorithm at the receiver. As can be seen, similar gains are obtained, except that the results are now closer to the “error-free case”. Also, note that the proposed scheme is more powerful at higher crossover error probabilities, as it virtually increases the size of the available code set and allows better protection than would be possible with the

\[^3\text{Reducing the maximum number of iterations to 100 as in [14] will only reduce the reported PSNRs in Table 3.6 by about 0.15 dB.}\]
strongest code rate in the code set. Finally, we note that at $\epsilon_0 = 0.1$, the proposed scheme is at most only $\approx 0.3$ dB away from the maximum achievable PSNR for all the simulations presented.

### 3.4.3 Discussion on the Advantages of the Proposed Transmission Scheme

Progressive image reconstruction has several advantages. One of these advantages is not preserved by our coding structure, but the others are. Progressive coding allows rapid display of a low quality version of the image at early stages of the reception of the bit stream. The receiver progressively obtains better image quality as more bits reliably arrive, and can redisplay the image multiple times at increasingly higher quality, allowing one to abort the image early in the reception if one determines, based on the low quality version, that the image is not the one desired. This might allow for faster browsing of a remote image database. This feature of allowing redisplaying multiple times is not preserved by our coding structure, as decoding of the most important information chunk cannot proceed immediately when it is received, as it is interleaved with other information chunks and their parity bits. This advantage of progressive image coding does not, in any case, extend to scalable video coding, because a single frame of video, even if it were scalably encoded, is not redisplayed multiple times at increasing quality levels.

The other advantages of progressive image transmission (which do extend to
scalable video) are supported by our encoding structure. Progressive coding allows for simple downsizing of the total bit stream in case of congestion at an intermediate router. This feature is retained. The router would need to deinterleave, but would not need to do either source decoding or channel decoding in order to strip off sections of information bits and parity bits and transmit onwards the beginning portions of the progressive stream. Finally, progressive encoding has the advantage of being very suitable for UEP, thus the initial information bits are more heavily protected for a graceful degradation of transmitted source quality at the receiver. We note that if the proposed system were used for scalable video coding, the overall delay incurred on the encoder side would be dominated by the delay of the scalable video source encoder, and not by the delay of the proposed channel encoder and interleaver. Although some scalable video coders incur significant delay (such as 3D SPIHT which might need to buffer a group of 16 frames), for a lower delay encoder such as H.264 fine grain scalability, if the $M$ codewords are all part of one frame, then the interleaving is all done within one frame. In this case, the proposed system will incur negligible processing delay.

3.5 Summary and Conclusions

In this chapter, we have considered an embedded bit stream using concatenated block encoding in conjunction with random block interleavers. A novel formatting is introduced for embedded bit streams with better reconstruction properties
at the receiver. Both the information block size and the channel code rates are subject to an optimization which makes the proposed system more flexible. A low complexity descent search for block size adjustment as well as a constrained exhaustive search in finding the optimal code rates are presented. In addition, using the proposed image coding system, we use multiple code rates for the protection of information blocks which makes additional BERs achievable beyond those possible using conventional methods with a discrete channel code rate set. The proposed scheme achieves this performance gain at the expense of loss of progressive display capability and increased complexity.
Acknowledgement

This chapter, in part, is a reprint of the material as it appears in S. S. Arslan, P. C. Cosman, and L. B. Milstein, “Concatenated Block Codes for Unequal Error Protection of Embedded Bit Streams,” IEEE Transactions on Image Processing, vol. 21, no. 3, pp. 1111–1122, March 2011. I was the primary author of this publication and the coauthors Prof. Cosman and Prof. Milstein directed and supervised the research which forms the basis for this research paper.
Chapter 4

Generalized Unequal Error Protection Fountain Codes for Progressive Data Transmission

In this chapter, we consider transmission scenarios in which no information about the channel is available while the data transmission takes place over time-varying channels with unknown parameters. DF codes have been known to be an excellent match for erasure channels with uncertain parameters. The original design of standard DF codes assumed that the coded information symbols are equally important. In many applications, some source symbols are more important than others, and they must be recovered prior to the rest. UEP and unequal recovery time (URT) designs are attractive solutions for such source transmissions. In this work, we introduce a more generalized design for the first universal DF code design, LT
codes [67], that makes it particularly suited for progressive bit stream transmissions. First we introduce a progressive transmission system and then we apply what we defined below to be a Generalized LT (GLT) code to a progressive source.

The rest of this chapter is organized as follows: Section 4.1 discussed the related research and the contributions of this chapter. Section 4.2 gives the necessary background for progressive source coding, the first DF design, LT codes and previous UEP designs. In Section 4.3, we introduce the proposed UEP GLT coding design, the encoding algorithm, the optimization problem and the Unequal Iteration Time (UIT) property. Section 4.4 introduces the proposed progressive transmission setup in detail. It also describes parameter selections of the UEP GLT coding for comparisons to some of the major UEP designs. Section 4.5 presents numerical results to show the effectiveness of the proposed idea. Finally, Section 4.6 draws some conclusions and presents future directions.

4.1 Related Work and Contributions

UEP for progressive sources is usually achieved by JSCC, in which more important contents are encoded with stronger codes (see, e.g., [10], [12], [13] and [66]). Those studies assume that CSI is available at the transmitter. When the channel erasure rate or the channel fading coefficients are unknown and non-uniform, one JSCC mechanism targets the worst-case error rate as the primary design criterion in choosing the optimal code rate/s [9]. An optimal fixed code rate for a good state
is unable to perform well when the channel is bad. On the contrary, when the code rate is optimized for the bad state, then there will be many unnecessary redundant bits. DF codes such as LT codes [67], Raptor codes [68] and Online codes [69] do not assume any information about the channel and therefore are good matches for transmitting data over time varying channels with unknown parameters. Similarly, DF codes can be very good candidates for multicast transmissions because there is no prior assumption about the channels.

The original LT codes assume each information symbol has equal significance and are EEP schemes. One of the first UEP LT code designs is presented in [70]. It is shown in a series of studies ( [70]– [73]) that UEP LT codes can be produced simply by allowing coded symbols to make more edge connections with more important parts of the bit stream with high probability. This way, the unrecovered symbol probability becomes lower for the high priority content of the original source. The ideas presented in those studies are successfully applied to various transmission scenarios such as [74] and [75].

In [72], a structured LT code graph is constructed. Then the proposed code is randomized using a Degree Assigner and a Random Selector. Although the terminology in [72] is different from [70], the unequal protection is achieved essentially the same way in [70] using a fixed degree distribution and a non-uniform selection distribution. In other studies such as [76], the authors introduced the use of overlapped windows to allow one to have a virtually extended block for superior performance compared to fixed-window encoding. Although the main objective was to increase
the virtual block size of the LT code for a better performance, it is recognized that such block size enhancement can be used to provide UEP. Later, a similar idea is used in [77], called block duplication, for the equal error protection of a two-layer source. It is also shown that under certain assumptions, the method of [77] provides slightly better performance than [70]. Yet, the gains reported in [73] over [70] are greater due to using different degree distributions for each window. This shows the combined choice of degree and selection distributions had a great impact on the final performance of the fountain code. Lastly, the algorithm presented in [78] chooses degree-one coded symbols from the high priority class of source symbols and the edge connections of degree-two coded symbols are selected non-uniformly similar to the weighted approach of [70]. The UEP is achieved by this slight modification and a fixed degree distribution, and the rest of the encoding is exactly the same as in original LT coding. Although all of these studies [70]– [78] consider fountain codes with a UEP property for a given application scenario, a joint optimization of degree and selection distributions remains untouched.

We consider the first universal DF code design, LT codes. The methods developed can be applied to other DF codes such as Raptor codes, which are obtained by precoding LT encoded information symbol streams [68]. Since the beginning part of progressive bit streams is more important than the succeeding parts, previous UEP designs can directly be used in the transmission of progressive sources. However, progressive sources do not consist of only a few layers as traditionally assumed in previous scalable video transmission scenarios [79].
In this study, we propose a generalization for two major UEP LT code designs. Specifically, inspired from [78], we introduce a systematic degree-dependent selection concept that can be applied to previous UEP LT code designs. In addition, we present a progressive source transmission scheme using rateless codes. We tailor the parameters of the proposed design for this particular scenario to minimize the expected distortion. We show that although the previous UEP schemes can be used to provide UEP for progressive bit streams, the proposed generalization of the rateless codes and its configuration for progressive bit streams give dramatic improvements in terms of end-to-end expected distortion over the UEP rateless code designs described in [70]–[73]. Furthermore, we introduce a new property called UIT where we evaluated system performance as a function of the iteration index of the decoding algorithm. This might be particularly important for portable devices which are constrained by low-complexity receiver architectures.

4.2 Background Overview

In this section, we first review progressive source coding. Then, we review DF codes and give original encoding and decoding structures for LT codes. Lastly, we briefly discuss the previous UEP designs.
4.2.1 Progressive Source Coding

In progressive source coding, prefixes of a single bitstream allow the decoder to progressively reconstruct the source. In other words, all encodings of the source at lower bit rates are embedded in the beginning of the encoded source at higher bit rates. Thus, progressive source bitstreams have the property that the beginning part of the bit stream is more important than the succeeding parts of the bit stream. Progressive source coders also have the property that bits later in the bit stream are of no use unless the bits that precede them are reliably received. In this respect, it is therefore convenient to define *useful bits* to be the set of consecutive bits that are recovered from the beginning of the bit stream up to the first unrecovered bit location. In progressive source transmission, it is of more concern to consider the decoded useful bits rather than the decoded total bits.

Progressive transmission might be very useful in multimedia communication scenarios, in particular fast browsing of high definition media in non-homogeneous networks. However, such bit streams use various forms of variable length codes making them extremely susceptible to noisy channel effects, i.e., any bit error might render the rest of the bitstream useless. Therefore, a good protection mechanism is needed for reliable transfer of such compressed data.
4.2.2 Digital Fountain Codes

DF codes are random bipartite graph codes in which the source data is recoverable from any subset of coded symbols when this subset contains enough coded symbols in a lossy packet transmission scenario. These codes have significantly less complex encoding and decoding structures than more traditional RS codes and exhibit high erasure correction performance for large block lengths. A DF encoder generates potentially an unlimited number of coded symbols. Once enough coded symbols are collected at the receiver, an iterative decoder using a belief propagation (BP) algorithm recovers the information symbols through graph pruning.

Encoding

An LT encoder takes a set of $k$ symbols of information to generate coded symbols of the same alphabet. We consider a binary alphabet and let a binary information block $\mathbf{x}^T = (x_1, x_2, \ldots, x_k) \in \mathbb{F}_2^k$ consist of $k$ bits. The $m$-th coded symbol $y_m$ is generated in the following way: First, the degree of $y_m$, denoted $d_m$, is chosen according to a suitable Degree Distribution (DD) $\Upsilon(x) = \sum_{\ell=1}^{k} \Upsilon_\ell x^\ell$ where $\Upsilon_\ell$ is the probability of choosing degree $\ell \in \{1, \ldots, k\}$. Then, after choosing the degree $d_m \in \{1, \ldots, k\}$, a $d_m$-element subset of $\mathbf{x}$ is chosen randomly according to the selection distribution (SD). For standard LT coding, the SD is the uniform distribution. This corresponds to generating a random vector $\mathbf{w}_m$ of length $k$, and $weight(\mathbf{w}_m) = d_m$ positions are selected from a uniform distribution to be logical 1, without replacement. More specifically, this means that any possible binary vector
of weight \( d_m \) is selected with probability \( \frac{1}{(d_m)} \). Finally, the coded symbol is given by \( y_m = w_m^T x \mod 2 \). Some of the coded symbols are erased by the channel, and for decoding purposes, we concern ourselves only with those \( n \) coded symbols which arrive unerased at the decoder. Hence the subscript \( m \) on \( y_m, d_m \) and \( w_m \) runs only from 1 to \( n \), and we ignore at the decoder those quantities associated with erased symbols. Note that the way we generate each coded symbol is independent of the way we generate other coded symbols. LT codes have been shown to be asymptotically optimal, i.e., as \( k \) tends to infinity, then \( n \rightarrow k \) bits will be enough to recover all the information content.

**Decoding**

The Maximum Likelihood (ML) decoding of LT codes over the BEC is the problem of recovering \( k \) information symbols from the \( n \) reliably received coded symbols. Although ML decoding is optimal, it is computationally prohibitive for long block lengths (\( k \) large). In order to allow linear decoding complexity with increasing block length, the Belief Propagation (BP) algorithm is used.

Let us denote information symbols as *variable nodes* and coded symbols as *check nodes* in the bipartite graph representation of LT coding as shown in Fig 4.1. Assuming there is at least one degree-one coded symbol received, the BP starts decoding from degree-one coded symbols. The content is immediately sent to their one neighbor (variable nodes) to decode the information bits. An example for the decoding algorithm operation with \( k = 4 \) and \( n = 5 \) is shown in Fig. 4.1. As
can be seen in Fig. 4.1 a), BP starts decoding from the degree-one coded symbol. In Fig. 4.1 b), the degree-one coded symbol has transferred its content to its one neighboring information symbol, and the corresponding edge is eliminated. This is called the variable node update step. In the next decoding step shown in Fig. 4.1 c), the recovered information symbol is added modulo 2 to each connected coded symbol to update its content. Then, the corresponding edges are eliminated. This step is called the check node update step. In the later stages, as depicted in Fig. 4.1 d), e) and f), update steps for variable nodes and check nodes are performed alternately to recover the information symbols. Finally, after all node updates are made as shown in Fig. 4.1 f), we recover all the information symbols. When there is no degree-one check node at any stage of BP, the algorithm stops and declares decoding error. Details of BP can be found in [67].

For the decoding of a whole information block to be successful, we need at least one degree-one check node at each iteration. The set of degree-one nodes is called the ripple. If the ripple does not have any elements, the BP stops iterating. Luby [67] proposed the Ideal-Soliton DD so that at each decoding iteration, the expected ripple size is one. This means in expectation, the algorithm never stops and decodes the whole information block.

**Definition 4.1:** Ideal-Soliton distribution

- $\Omega_1 = 1/k$

- For $i = 2, \ldots, k$: $\Omega_i = 1/i(i - 1)$.

However, the performance of the Ideal-Soliton distribution is poor in practice.
Figure 4.1: Belief propagation algorithm for LT decoding. k=4, n=5. Squares (□) are check nodes (coded symbols) and circles (○) represent variable nodes (information symbols).
because the ripple size can very likely be 0. Later, Luby [67] proposed the **Robust Soliton Distribution** (RSD) which overcomes this problem and gives good performance in practice. The expected ripple size of RSD is \( R = c \cdot \ln(k/\delta)\sqrt{k} \geq 1 \) for some suitable constant \( c > 0 \) and an allowable failure probability \( \delta \) of the decoder. RSD is shown to give good performance in practice.

<table>
<thead>
<tr>
<th>Definition 4.2: Robust-Soliton distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>• For ( i = 1, \ldots, k ): probability of choosing degree ( i ) is given by ( \mu_i = (\Omega_i + \tau_i)/\beta ),</td>
</tr>
<tr>
<td>where</td>
</tr>
</tbody>
</table>
| \( \tau_i = \begin{cases} 
  \frac{R}{i\kappa} & \text{for } i = 1, \ldots, k/R - 1, \\
  \frac{R \ln(R/\delta)/k}{k} & \text{for } i = k/R, \\
  0 & \text{for } i = k/R + 1, \ldots, k
\end{cases} \) |
| \( \beta = \sum_{i=1}^{k}(\Omega_i + \tau_i) \). |

### 4.2.3 UEP DF Code Designs

Previous LT code designs with UEP and URT properties can be classified into two main categories.

**Weighted Approach**

In studies such as [70], [71] and [77], the authors proposed various intuitive modifications to the SD of an LT code. These UEP schemes will be referred to as a *weighted approach* in this paper. A generic description of the weighted approach is as follows: A block of \( k \) source symbols is divided into \( r \) disjoint sets \( s_1, \ldots, s_r \),
having sizes $|s_j| = \alpha_j k$ symbols, where $0 < \alpha_j < 1$ are design parameters satisfying
\[ \sum_{j=1}^{r} \alpha_j = 1, \quad \alpha_j k \text{ is an integer and } |.| \text{ denotes the cardinality of the argument}. \]
The encoding process is the same as for an LT code except the check nodes select their adjacent variable nodes non-uniformly. More specifically, after choosing a degree $d_m$ according to some DD, $d_m$ information symbols are selected one by one: first the set $s_j$ is chosen from $\{s_1, \ldots, s_r\}$ with probability $\omega_j$. After choosing the set index, the input symbol is selected from a uniform distribution from the set $s_j$ only. This selection process is repeated $d_m$ times, without replacement, to determine $d_m$ distinct connections to the information symbols. Finally, the value of the coded symbol is given by the sum of the selected $d_m$ information symbols. The probability of selecting a particular set is designed such that the probability of recovery for more important classes of bits is higher than that of the less important classes of bits. Therefore, this approach is a generalization of LT codes in which the neighbors of a coded symbol are selected non-uniformly.

**Expanding Window Fountain Codes**

Another approach, called Expanding Window Fountain (EWF) codes, was developed in [73]. An information block of $k$ bits was divided into $r$ successively larger windows. This can be thought of as first dividing into $r$ disjoint sets $s_1, \ldots, s_r$, and then defining $r$ embedded windows $\{W_j\}_{j=1}^{r}$ such that $W_j = \bigcup_{i=1}^{j} s_i$. To generate a new EWF coded symbol, one of the windows is randomly selected by the coded symbol according to a window selection distribution given by
Definition 4.3: Window SD

- \( \Gamma(x) = \sum_{j=1}^{r} \Gamma_j x^i \)

where \( \Gamma_j \) is the probability that window \( W_j \) is chosen.

Upon selection of the window, standard LT coding is applied only to the bits contained in that window using a suitably chosen degree distribution given by

Definition 4.4: \( j \)-th window DD

- For \( j = 1, \ldots, r \): \( \Phi^{(j)}(x) = \sum_{i=1}^{||W_j||} \Phi^{(j)}_i x^i \)

where \( \Phi^{(j)}_i \) is the conditional probability of choosing degree \( i \), given that \( W_j \) is selected by the coded symbol.

The same procedure is invoked for each encoded symbol. Thus, a UEP EWF code is a DF code which provides unequal protection by choosing the appropriate set \( \{ \Gamma(x), \Phi^{(1)}(x), \Phi^{(2)}(x), \ldots, \Phi^{(r)}(x) \} \). UEP EWF coding modifies not only the SD but also the DD, which makes the code a more flexible UEP scheme compared to the weighted approach. However, the superiority of this scheme over the weighted approach depends on whether the DD for each window is judiciously selected.

4.3 UEP Generalized LT (UEP GLT) Coding

In this section, first we apply the degree-dependent selection idea to the weighted approach to provide increased UEP, URT and UIT properties. Note that the same idea can be applied to UEP EWF codes through the use of degree-dependent window selection distribution. More specifically, we propose to use a
(window) SD that depends on the degree number of the particular check node to give priority decoding to earlier bits of a progressive bit stream.

### 4.3.1 Generalization of “weighted approach”

Similar to previous studies, let us partition the information block into \( r \) variable size disjoint sets \( s_1, s_2, \ldots, s_r \) (\( s_j \) has size \( \alpha_j k; j = 1, \ldots, r \) such that \( \sum_{j=1}^{r} \alpha_j = 1 \) and the \( \alpha_j k \) values are integers). In the encoding process, after choosing the degree number for each coded symbol, we select the edge connections according to a Generalized SD given by

**Definition 4.5: Generalized SD**

- For \( i = 1, \ldots, k \): \( P_i(x) = \sum_{j=1}^{r} p_{j,i} x^j \)

where \( p_{j,i} \geq 0 \) is the conditional probability of choosing the information set \( s_j \), given that the degree of the coded symbol is \( i \) and \( \sum_{j=1}^{r} p_{j,i} = 1 \).

Note that \( p_{j,i} \) are design parameters of the system, subject to optimization.

For convenience, we denote the proposed SD in a matrix form as follows:

\[
P_{r \times k} = \begin{bmatrix}
p_{1,1} & p_{1,2} & \cdots & p_{1,k} \\
p_{2,1} & p_{2,2} & \cdots & p_{2,k} \\
\vdots & \vdots & \ddots & \vdots \\
p_{r-1,1} & p_{r-1,2} & \cdots & p_{r-1,k} \\
p_{r,1} & p_{r,2} & \cdots & p_{r,k}
\end{bmatrix}
\]

Since the set of probabilities in each column sums to unity, the number of design parameters of \( P_{r \times k} \) is \((r - 1) \times k\). Similarly, for the proposed GLT, the DD
Algorithm 4.1: UEP GLT Encoding (for “weighted approach”)

for \( m = 1, \ldots, n \),

- Choose a degree \( d_m \in \{1, \ldots, k\} \) according to some appropriate degree distribution \( \Upsilon_w(x) \).

- Initialize \( \text{count}_{\text{deg}} = 1, \text{count}_{\text{edge}[r]} = 0 \)

- while \( \text{count}_{\text{deg}} \leq d_m \)
  \( \implies \) Choose a set index \( j \in \{1, \ldots, r\} \) according to the Generalized SD \( \{p_{1,d_m}, p_{2,d_m}, \ldots, p_{r,d_m}\} \)
  \( \implies \) if \( \text{count}_{\text{edge}[j]} < \alpha_j k \)
    \( \diamond \) Choose an information symbol from \( s_j \) uniform randomly. \( \text{count}_{\text{edge}[j]} \) previously chosen information symbols are excluded from this selection process (selection without replacement).
    \( \diamond \) \( \text{count}_{\text{edge}[j]} = \text{count}_{\text{edge}[j]} + 1. \)
  \( \implies \) else
    \( \diamond \) Choose an information symbol from all the sets except \( s_j \) uniform randomly. Again, the selection uses only information symbols not previously chosen.
    \( \diamond \) if selected information symbol belongs to \( s_t, t \neq j \)
      \( \diamond \) \( \text{count}_{\text{edge}[t]} = \text{count}_{\text{edge}[t]} + 1. \)
    \( \diamond \) end if
  \( \implies \) end if
  \( \implies \) \( \text{count}_{\text{deg}} = \text{count}_{\text{deg}} + 1. \)
- end while

- XOR all the selected information symbols to find the value of \( y_m \).

end for
can be expressed in a vector form as $\lambda_k$, where the $i$th vector entry is the probability that a coded symbol chooses degree $i$. Note $\lambda_k$ and $P_{r \times k}$ completely determine the performance of the proposed GLT code. More specifically, the design steps taken to generate each coded symbol are summarized in Algorithm 4.1. Note that $\text{count}_{\text{edge}}[r]$ denotes a vector of length $r$. Also, $\text{count}_{\text{edge}}[r] = \mathbf{0}$ denotes that each entry of the vector is initialized to 0.

**Figure 4.2**: Non-uniform selection of variable nodes. Black squares represent coded symbols (check nodes) and circles represent information symbols (variable nodes). White and gray circles represent two different priority classes.

In the BP algorithm, we observe that not all the check nodes decode information symbols at each iteration. For example, degree one check nodes immediately decode neighboring information symbols at the very first iteration [67]. Then, degree two and three check nodes recover some of the information bits later in the sequence of iterations. In general, at the later update steps of iterations, low degree check nodes will already be released from the decoding process, and higher degree check nodes start decoding the information symbols (due to edge eliminations). So the coded symbols take part in different stages of the BP decoding process depending
on their degree numbers.

As is clear from the previous UEP DF code designs, UEP is achieved by allowing coded symbols to make more edge connections with more important information sets. This increases the probability of decoding the more important symbols. However, coded symbols are able to decode information symbols in different iterations of the BP depending on their degree numbers. For example, at the second iteration of the BP algorithm, the probability that degree-2 coded symbols decode information symbols is higher than that of coded symbols with degrees > 2. If the BP algorithm stops unexpectedly at early iterations, it is essential that the more important information symbols are recovered. This suggests that it is beneficial to have low degree check nodes generally make edge connections with important information sets. This can be achieved by the proposed scheme. An example realization for \( r = 2 \) is shown in Fig. 4.2. A check node with degree 6 makes \( t_2 = 1 \) connection with \( s_1 \) whereas a check node with degree 4 makes \( t_1 = 3 \) connections with \( s_1 \) satisfying \( t_1 > t_2 \). The main advantage of the proposed scheme over the previous designs is to allow a greater flexibility in the LT encoding process. We will see later that this flexibility will allow us to tailor the parameters of the system to a progressive source transmission scenario.

4.3.2 Generalization of EWF Codes

In the encoding process of this generalization, after choosing the degree number for each coded symbol, we select the edge connections according to a Generalized
window SD given by

**Definition 4.6: Generalized window SD**

- For \( i = 1, \ldots, k \): 
  \[
  L_i(x) = \sum_{j=1}^{r} \gamma_{j,i} x^j
  \]

where \( \gamma_{j,i} \geq 0 \) is the conditional probability of choosing the information set \( s_j \), given that the degree of the coded symbol is \( i \) and \( \sum_{j=1}^{r} \gamma_{j,i} = 1 \).

Similar to the previous generalization, \( \gamma_{j,i} \) are design parameters of the system, subject to optimization. For convenience, we denote the proposed window SD in a matrix form as follows:

\[
L_{r \times k} = \begin{bmatrix}
\gamma_{1,1} & \gamma_{1,2} & \cdots & \gamma_{1,k} \\
\gamma_{2,1} & \gamma_{2,2} & \cdots & \gamma_{2,k} \\
\vdots & \vdots & \ddots & \vdots \\
\gamma_{r-1,1} & \gamma_{r-1,2} & \cdots & \gamma_{r-1,k} \\
\gamma_{r,1} & \gamma_{r,2} & \cdots & \gamma_{r,k}
\end{bmatrix}
\]

The set of probabilities in each column sums to unity, and the number of design parameters of \( L_{r \times k} \) is again \( (r - 1) \times k \). Similarly, we observe that \( \lambda_k \) and \( L_{r \times k} \) completely determine the performance of the proposed generalization of EWF codes. More specifically, the design steps taken to generate each coded symbol are summarized in Algorithm 4.2.

### 4.3.3 Unequal Iteration Time Property

The URT definition given in [71] or [73] is with respect to the reception overhead (\( \epsilon \)), i.e., it examines the fraction of the source message that can be recovered
Algorithm 4.2: UEP GLT Encoding (for EWF codes)

\textbf{for} \( m = 1, \ldots, n \),

- \textit{Choose a degree} \( d_m \in \{1, \ldots, k\} \) according to some appropriate degree distribution \( \Upsilon_{ewf}(x) \).

- \textit{Initialize} \( \text{count\_deg} = 1, \text{count\_edge}[r] = 0 \).

- \textit{while} \( \text{count\_deg} \leq d_m \)
  \hspace{1em} \textit{Choose a window index} \( j \in \{1, \ldots, r\} \) i.e., \( W_j \) according to the Generalized window SD \( \{ \gamma_{1,d_m}, \gamma_{2,d_m}, \ldots, \gamma_{r,d_m} \} \)
  \hspace{1em} \textit{if} \( \text{count\_edge}[j] < k \sum_{i=1}^{j} \alpha_i \)
    \hspace{1em} \textit{Choose} an information symbol from \( W_j \) uniform randomly.
    \hspace{1em} \textit{count\_edge}[j] \) previously chosen information symbols are excluded from this selection process (selection without replacement).
    \hspace{1em} \textit{count\_edge}[j] = \text{count\_edge}[j] + 1.
  \hspace{1em} \textit{else}
    \hspace{1em} \textit{Choose} an information symbol from the set \( \bigcup_{i=j+1}^{r} s_i \) uniform randomly.
  \hspace{1em} \textit{Again, the selection uses only information symbols not previously chosen by the same coded symbol.}
  \hspace{1em} \textit{end if}
  \hspace{1em} \textit{count\_deg} = \text{count\_deg} + 1.

- \textit{end while}

- \textit{XOR} all the selected information symbols to find the value of \( y_m \).

\textit{end for}
for different overhead values. This means that given a target bit error rate, increasing numbers of information bits can be decoded after receiving increasing numbers of encoded bits, so that information bits can be recovered in a progressive manner.

So URT is concerned with performance as a function of the number of received symbols. In contrast to this definition, UIT is concerned with the performance as a function of the number of iterations of the decoding algorithm. Usually it is assumed that the decoding algorithm iterates as much as needed. Since rateless codes are most effective with increasing source block sizes, this requires more iterations in the BP algorithm. However, in many portable wireless applications, low-complexity designs are desired for less battery consumption. In that scenario, UIT is relevant and provides insights about the performance of the compared schemes.

4.3.4 Optimization Problem

The BP algorithm can terminate at any iteration with some non-zero probability. Let $Pr(Y = s)$ be the probability that the BP algorithm terminates at the $s$th iteration. In this type of algorithm, one typically chooses the maximum number of iterations of the BP algorithm ($M_{\text{max}}$) such that $Pr(Y > M_{\text{max}})$ is negligible. $M_{\text{max}}$ is usually chosen based on the value of $k$, the number of information symbols being encoded in all the simulations. In our case, we chose $M_{\text{max}} = 70$ because the algorithm always terminated (either by correct decoding or by having a decoding
failure) prior to 70 iterations being reached. The optimization problem is given by:

$$\min_{\lambda, P(\text{or } L)} D_M \text{ s.t. } n \text{ coded symbols are received unerased},$$

(4.1)

where $n \geq k$, and $D_M$ is the average mean square distortion at the $M$th iteration of the BP algorithm. Note that the optimization problem for Algorithm 4.2 is exactly the same except that we replace $P$ with $L$. The minimization can be done at any specific iteration $M$, $1 \leq M \leq M_{\text{max}}$. This could be useful if different UEP, URT and UIT characteristics are desired for a specific application. The minimization is over the entries of $\lambda$ and $P$ (or $L$). Note that the proposed code is specified by the entries of two matrices, $\lambda$ and $P$ (or $L$), and the total number of entries is $(r + 1)k$. However, the columns of $P$ (or $L$) as well as the entries of $\lambda$ should sum up to one. Therefore, we have $(r - 1)k + (k - 1) = rk - 1$ parameters subject to optimization. For large $k$, as a practical matter, it is infeasible to jointly optimize all design parameters. The following section uses comparisons with other UEP DF code designs to reduce the number of parameters subject to optimization for a progressive source transmission scenario.
4.4 Progressive Source Transmission System and Design Parameter Selection

4.4.1 Progressive Source Transmission System Description

Our previous descriptions are based on nodes containing bits. In most practical multimedia transmission systems, the basic unit of information is usually a fixed or variable length packet or a bit-segment [80]. In this section, we describe the way we transmit a progressive bit stream using rateless codes. The system block diagram of our proposed setup is shown in Fig. 4.3.

A bit stream is produced using a $L_x \times L_y$ grayscale image coded with SPIHT and arithmetic coding. The progressive bit stream is assumed to have a total of $B$ bits, i.e., a source rate of $B/(L_x \times L_y)$ bits per pixel (bpp). First, the source bit stream is divided into equal size blocks of bits. Then, the bits in those blocks are rearranged as shown in Fig. 4.4 to produce equal size segments. Using such a
Figure 4.4: The demultiplexing of the source bit stream.

configuration, for instance, the initial part of each block constitutes the contents of the first segment. Thus, the total source bit stream is demultiplexed to produce a set of equal size segments. Those segments are the output of the DEMUX block in Fig. 4.3. To be more specific, the progressive bit stream of $B$ bits is divided into $k$ equal size blocks (each with $\lfloor B/k \rfloor$ information bits) as shown in Fig. 4.5. Those information blocks are demultiplexed into $\lfloor B/k \rfloor$ equal size segments (each with $k$ bits) in the following way: The $z$th $k$-bit segment is generated by collecting the $z$th bit of each information block. The reason for using such a demultiplexing methodology is that the proposed coding scheme is most powerful when the source bits within each segment have unequal importance. Using demultiplexing, for example, the information bits in the first block, the most important information block, are equally shared by the segments. In contrast, we could have skipped the demultiplexing, i.e., we could have treated the source blocks directly as our segments (without the rearrangement shown in Fig. 4.4) before LT encoding takes place. However, in
Figure 4.5: Each information block is of size $\lfloor B/k \rfloor$ information bits. The $z$th $k$-bit segment is generated by collecting the $z$th bit of each information block. After forming each $k$-bit segment, a realization of the LT coding is applied to each $k$-bit segment.

In that case, each $k$-bit information segment would include almost equally-significant content.

After demultiplexing, we generate a particular realization of the proposed DF code and apply it to each $k$-bit segment to produce coded symbols. Assuming that each coded symbol stream goes through the same erasure channel, we collect $n$ coded symbols at the receiver for each $k$-bit segment decoding. Note that since we apply the same realization of the random code to each $k$-bit segment and use independent decoding, if $m$ information bits are useful in each segment after each BP decoding (as shown in Fig. 4.3), then because of the rearrangement/demultiplexing operation, we will have $\lfloor B/k \rfloor m$ total number of useful bits in the progressive bit stream for source decoding.
4.4.2 Comparison with the “weighted approach”

It is easy to see that the “weighted approach” is a special case of the proposed UEP GLT coding given in Algorithm 4.1. Since encoding-decoding is done according to two interrelated distributions (SD and DD), the design criterion in our case is to select both distributions judiciously to minimize the average distortion as given by Equation (4.1). To reduce the number of optimization parameters, let us choose \( p_{j,i} \) to be an exponential function of the degree number \( i \) for \( j = 1, 2, \ldots, r-1 \) as follows:

**Definition 4.7:** Exponential SD

\[
\begin{align*}
    p_{j,i} &= A_j + B_j \times \exp \left\{ -\frac{i-1}{C_j} \right\} \quad \text{for } i = 1, 2, \ldots, k \\
\end{align*}
\]

where \( \{A_j \geq 0, B_j \geq 0, C_j \geq 0\}_{j=1}^{r-1} \) are design parameters satisfying \( \sum_{j=1}^{r} p_{j,i} = 1 \) for all \( i \).

The exponential SD is an intuitive choice because the low degree check nodes make more edge connections on average with the more important information sets.

Note that using the exponential SD, we reduce the parameter space size from \((r-1)k + k - 1\) to \(3(r-1) + k - 1\). Let us use standard DDs (for example RSD) and a predetermined partitioning set \( \{\alpha_1, \ldots, \alpha_r\} \) in conjunction with an exponential SD, so that we will have only \(3(r-1)\) parameters subject to optimization. Note that an additional optimization can be run over the partitioning set \( \{\alpha_1, \ldots, \alpha_r\} \).

In that case, the parameter space size increases to \(3(r-1) + r = 4r - 3\). As will be shown in the numerical results section, this leads to slightly better performance at the expense of increased complexity.
4.4.3 Comparison with UEP EWF Codes

In EWF codes [73], one of the expanding windows is first selected by a coded symbol before the selection of its edge connections. After choosing a specific window, all the edge connections of that coded symbol are constrained to be chosen from the selected window. In our UEP GLT coding process (Algorithm 4.1), the edge connections are not constrained in that way. Thus, although the EWF code is not a special case of the proposed UEP GLT code given in Algorithm 4.1, it is a special case of Algorithm 4.2.

Let us use a DD called the Compound degree distribution \( \Lambda^c(x) \), given by

**Definition 4.8:** Compound DD

- \( \Lambda^c(x) = \sum_{i=1}^{k} \Lambda_i x^i \) where \( \Lambda_i \triangleq \sum_{j=1}^{r} \rho_j \Phi^{(j)}_i \)

where \( \Lambda_i \) is the probability of choosing degree \( i \), \( \Phi^{(j)}_i \triangleq 0 \) if \( i > |W_j| \) and \( 0 \leq \{\rho_j\}_{j=1}^{r} \leq 1 \) such that \( \sum_{j=1}^{r} \rho_j = 1 \).

It can be shown that the Compound DD is a valid probability mass function by realizing that it is a convex combination of probability mass functions.

The main motivation behind using such a distribution is the following. In a progressive transmission, recovery of the whole source block is not usually the key objective. The goal is to maximize the number of useful source symbols. Thus, the outcome of the optimization is the source block size that can be communicated to the receiver with small error probability. Using a Compound DD enables us to tailor a set of DDs (specifically good for a set of source block sizes) for our progressive source transmission scenario. Using appropriate weights \( \rho_j \), we can make the Compound
DD a good DD for a specific information block size and a source whose rate-distortion (R-D) characteristic is known. Finally, the parameters of the Compound DD are fed to the optimization to minimize the distortion.

Similar to previous reasoning, in order to reduce the number of optimization parameters, we choose $\gamma_{j,i}$ to be an exponential function of degree number $i$ for $j = 1, 2, \ldots, r − 1$ as follows:

**Definition 4.9:** Exponential window SD for $i=1,2,\ldots,k$

$$\gamma_{j,i} = \begin{cases} A_j + B_j \times \exp \left\{ \frac{-i}{C_j} \right\} & \text{if } i \leq k \sum_{t=1}^{j} \alpha_t \\ 0 & \text{if } i > k \sum_{t=1}^{j} \alpha_t \end{cases} \quad (4.3)$$

where $\{A_j, B_j, C_j\}_{j=1}^{r-1}$ are design parameters satisfying $\sum_{j=1}^{r} \gamma_{j,i} = 1$ for all $i$.

### 4.5 Numerical Results

In this section, we will compare the performance of our proposed scheme with that of two other major classes of UEP LT schemes. We use the minimum distortion criterion throughout our simulations. Quality assessment of the decoded images is given in terms of the average Peak Signal to Noise Ratio (PSNR), expressed in dB, a performance measure inversely related to the average mean square distortion by the $M$th iteration of the BP algorithm, $\overline{D}_M$. We use standard 512 × 512 Lena and 512 × 512 Goldhill images.

We initially set $B = 50000$ bits ($\approx 0.1907$bpp source rate) and run all realizations (encoding & decoding using $M_{max} = 70$) $10^4$ times and compute the total
number of useful bits (the number of recovered consecutive information symbols) in each realization. Since a specific image with a particular source encoder determines the R-D characteristics of the source, we can find the distortion\(^1\) corresponding to the useful number of bits. Then, we average these computed distortion values computed for each of the \(10^4\) different realizations. We optimize the parameters of each system to give the minimum average distortion (the solution to the optimization problem). Lastly, minimum average distortion is converted to average PSNR.

4.5.1 Comparisons with the “weighted approach”

In our first simulation, we compare our UEP GLT scheme using Algorithm 4.1 with the weighted approach. We set \(r = 2\) and \(B = 5 \times 10^4\). Both schemes use the RSD with \(\gamma = c = 0.01\) i.e., \(\Upsilon_w(x)\) is an RSD in Algorithm 4.1. We define the following version of the proposed scheme:

- **GLTexp**: This scheme uses the Exponential SD with \(A_1 + B_1 = 1\) and optimizes the set \(\{A_1, C_1\}\) so that the proposed scheme achieves minimum distortion.

For a fair comparison, we also optimize the weighting parameter \(\omega_1\) for the weighted approach for minimum distortion. Average PSNR in dB versus the number of reliably received coded symbols \(n\) is plotted for both systems using a fixed \(\alpha_1 = 0.3\) and the optimal \(\alpha_1\) value for each system. We also included the performance

\(^1\)We use MSE as our distortion metric throughout the paper.
of the standard LT coding (EEP scheme) in the same figure for comparison. In Figs. 4.6 and 4.7, performance results are shown for \( k = 100 \) as a function of \( n \), and observed to be increasing with growing \( n \). Although the proposed scheme does not show a major performance improvement over the weighted approach up to \( n = 130 \) coded symbols, it is shown to provide over 1dB of improvement for a substantial range of \( n \) (from 140 to 210) for both fixed or optimal \( \alpha_1 \), and a huge improvement over standard LT coding. After collecting \( n > 240 \) coded symbols, all the systems perform almost a complete decoding of the whole source block and hence they exhibit a similar performance. Also note that for the weighted approach, the optimal \( \alpha_1 \)

![Figure 4.6](image-url)

**Figure 4.6**: Performance comparisons using *Lena* and \( k = 100 \).
Figure 4.7: Performance comparisons using Goldhill and $k = 100$.

$(\alpha_i^*)$ turns out to be 0.3 for $n \geq 160$. For comparison, the PSNR performance of a source with a rate of $\sim 0.19$bpp, operating under error-free conditions, is included in the figures.

Table 4.1: Number of coded symbols which need to be sent for a PSNR of 31dB using Lena.

<table>
<thead>
<tr>
<th></th>
<th>$k = 100$</th>
<th>$k = 1000$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$n$</td>
<td>Savings</td>
</tr>
<tr>
<td><strong>GLTexp</strong></td>
<td>171</td>
<td>43</td>
</tr>
<tr>
<td><em>weighted approach</em></td>
<td>209</td>
<td>5</td>
</tr>
<tr>
<td><strong>standard LT</strong></td>
<td>214</td>
<td>–</td>
</tr>
</tbody>
</table>
Figure 4.8: Performance comparisons using Lena and $k = 1000$.

Table 4.2: Optimal parameters of various UEP designs shown in Figs. 4.6, 4.8.

<table>
<thead>
<tr>
<th></th>
<th>$A_1$</th>
<th>$B_1$</th>
<th>$C_1$</th>
<th>$\alpha_1$</th>
<th>PSNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>k=100, $\epsilon = 0.5$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>“weighted approach”</td>
<td>0.55</td>
<td>0.0</td>
<td>N/A</td>
<td>0.3</td>
<td>27.61</td>
</tr>
<tr>
<td>“weighted approach” ($\alpha_1^*$)</td>
<td>0.95</td>
<td>0.0</td>
<td>N/A</td>
<td>0.5</td>
<td>28.05</td>
</tr>
<tr>
<td>GLTexp</td>
<td>0.15</td>
<td>0.85</td>
<td>2.1</td>
<td>0.3</td>
<td>29.18</td>
</tr>
<tr>
<td>GLTexp ($\alpha_1^*$)</td>
<td>0.35</td>
<td>0.7</td>
<td>1.7</td>
<td>0.4</td>
<td>29.73</td>
</tr>
<tr>
<td>k=1000, $\epsilon = 0.3$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>“weighted approach”</td>
<td>0.45</td>
<td>0.0</td>
<td>N/A</td>
<td>0.3</td>
<td>29.63</td>
</tr>
<tr>
<td>“weighted approach” ($\alpha_1^*$)</td>
<td>0.76</td>
<td>0.0</td>
<td>N/A</td>
<td>0.6</td>
<td>30.39</td>
</tr>
<tr>
<td>GLTexp</td>
<td>0.25</td>
<td>0.75</td>
<td>1.9</td>
<td>0.3</td>
<td>30.94</td>
</tr>
<tr>
<td>GLTexp ($\alpha_1^*$)</td>
<td>0.25</td>
<td>0.71</td>
<td>1.6</td>
<td>0.29</td>
<td>32.39</td>
</tr>
</tbody>
</table>
In Fig. 4.8, we also show the performance of these systems for $k = 1000$. As can be seen, when $k$ gets large, the performance of each system increases, as does the gain of the proposed scheme over the weighted approach. It is quite common to define the coding overhead of a rateless code as $\epsilon \triangleq (n/k) - 1$. Using the proposed scheme with Lena, for example, an overhead of 0.3 when $k = 1000$ gives around 32.5dB average PSNR, whereas the same overhead with $k = 100$ gives around 27.6dB average PSNR. Another way of looking at these performance curves is to consider the percentage of savings of the coded symbols reliably received for a given image quality. Table 4.1 shows the number of coded symbols which need to be received to obtain a PSNR of 31dB using various rateless code schemes. As can be seen, a substantial savings (relative to standard LT coding) in terms of the received unerased coded symbols can be obtained. Also, Table 4.2 shows some of the optimal parameters used to obtain some of the data points shown in Fig. 4.6 and Fig. 4.8.

Note that Figs. 4.6, 4.7, and 4.8 can all be considered to be depictions of both the URT and UEP performances of the various schemes. These figures all show quality as a function of the number of received symbols. Therefore, for a fixed quality, the horizontal distance between two curves is a measure of how much earlier in the received bit stream one system can recover that fixed quality compared to another system (URT property). For a fixed number of received symbols, the vertical distance between two curves is a measure of the PSNR gain (UEP property).

Fig. 4.9 shows performance comparisons for a range of $B$ bits and $\epsilon = 0.4$. As a perspective on PSNR, assume that all the source bits are recovered. We call this
Figure 4.9: Performance comparisons using *Lena* and a overhead of $\epsilon = 0.4$ for a range of $B$. 
idealized scenario the “error-free” case, (the solid blue curve in Fig. 4.9). We observe that the proposed scheme not only improves the performance over the weighted approach, but also gives results that are close to the “error-free” case. One other observation is that for \( k = 100 \), an improvement of almost 1dB is possible for a range of \( B \) (from \( B \approx 2 \times 10^4 \) to \( B \approx 6 \times 10^4 \)). For the range of values shown on the abscissa, increasing \( B \) yields larger gains when \( k = 1000 \), but much less when \( k = 100 \). The reason is related to the system description and the assumption made in Subsection A. A closer look at Fig. 4.5 reveals that \( k \) also equals the number of blocks, and for a fixed \( k \), the size of the information blocks increases with growing \( B \). For large \( B \) and small \( k \), the contents of each segment might be quite different. For example, if \( k = 2 \), the first segment will have one bit from the beginning of the total bit stream and one bit from the middle. The last segment will have one bit from the middle of the bit stream and one bit from the end. Since \( B \) is large and the progressive coder has a non-increasing and convex R-D characteristic, these two segments are substantially different from each other. Considering the progressive source transmission system shown in Fig. 4.3, we note that the same code is applied for each segment. Therefore, a set of optimal values tailored to a particular segment might not be the best set of parameters for other segments. This degrades the performance of the system. If \( k \) gets large, the R-D characteristics of each segment become similar, and thus the parameters of the proposed code fit a larger set of segments. We note that the progressive source transmission system shown in Fig. 4.3 assumes that one segment contains almost equally important
content compared to another segment, although the importance varies considerably within each segment. In Fig. 4.9, for the parameters of this scenario, $k = 100$ turns out to be too small for this assumption to hold; the $k$-bit segments differ from each other in the importance of their information content. However, if we increase $k$ from 100 to 1000, the assumption of Fig. 4.3 will approximately hold and segments become almost equally important and have similar R-D characteristics. Thus, the proposed scheme gives increased gains at larger $B$ values, as shown in Fig. 4.9.

Finally, we show the UIT performance of the proposed scheme, that is, how the PSNR varies if the BP algorithm is not allowed to iterate until its natural
Table 4.3: Parameters of the UEP GLT and Comparison of UEP DF designs.

<table>
<thead>
<tr>
<th>DF code</th>
<th>r</th>
<th>Parameters</th>
<th>DD</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>LT</td>
<td>1</td>
<td>N/A</td>
<td>$\Omega_{rs}(k, \gamma, c)$</td>
<td>Uniform</td>
</tr>
<tr>
<td>UEP EWF</td>
<td>2</td>
<td>$\alpha_1, \Gamma_1$</td>
<td>$\Omega_{rs}(\alpha_1 k, \gamma, c), \Omega_{rs}(k, \gamma, c)$</td>
<td>Uniform for the window</td>
</tr>
<tr>
<td>UEP GLT</td>
<td>2</td>
<td>$\alpha_1, \rho_1, A_1, B_1, C_1$</td>
<td>$\Lambda^{(c)}(x)$</td>
<td>Exponential window</td>
</tr>
</tbody>
</table>

termination\(^2\), but is instead cut off by design at some early iteration $M$. We note that the previous results are based on solving the minimization problem in Equation 4.1 for $M = 70$. However, the parameters that give the best performance for $M = 70$ do not necessarily give the best performance at early iterations. A simulation result is shown in Fig. 4.10, in which the proposed scheme and the weighted approach are compared using parameter values optimized for $M = 70$ as a function of the number of iterations in the BP algorithm. A 1.7dB PSNR gain when the number of iterations is greater than 16 is shown over the weighted approach. Fig. 4.10 also shows a performance curve that results by solving the minimization problem in Equation (4.1) for $M = 6$. This curve gives a large gain over the weighted approach at iteration 6 at the expense of some performance loss (compared to the weighted approach) at later iterations. The performance curves optimized for $M = 6$ and $M = 70$ suggest that there is a family of performance curves that give better PSNR gains at early iterations if we allow some performance loss at later iterations. For example, in Fig. 9 we show a family of curves obtained by varying parameter $C_1$ from 1.2 to 2.2 with fixed $\{A_1 = 0.55, B_1 = 0.45\}$. From this family of curves, we show

\(^2\)By natural termination, we mean the algorithm either decodes all the information bits, or declares failure, i.e., there remains no degree-one coded symbol after edge eliminations and node updates even if the decoding of the whole source block is not complete.
one of them with \( C_1 = 1.6 \). It can be observed that, although this sample system performs worse than the system optimized for \( M = 70 \), it performs better than the weighted approach for all iterations, and especially for some early iterations. For example, at iteration 6, it provides a 3.32dB gain over the weighted approach. This result shows that we can tailor the parameters of the proposed scheme to achieve better UIT properties at the expense of some loss in performance at later iterations.

4.5.2 Comparisons with UEP EWF Codes

We now compare the proposed generalization i.e., Algorithm 4.2 with UEP EWF codes. We also compare the proposed scheme with increased parameter sizes. The EWF code has the parameters \( \alpha_1 \) and \( \Gamma_1 \) subject to optimization. The DDs for the UEP EWF code, \( \{ \Phi^{(1)}(x), \Phi^{(2)}(x) \} \) are the Truncated Robust Soliton distribution \([73]\) \( \Omega_{rs}(k_{rs}, \gamma, c) \) using \( \gamma = c = 0.01 \), in which \( k_{rs} \) is the maximum degree of each DD and is constrained not to exceed the size of the corresponding window i.e., the size of the first window \( W_1 = \alpha_1 k \) or the size of the second window \( W_2 = k \) \([75]\). Details of the design parameters of the comparison systems are summarized in Table 4.3. The UEP GLT scheme given in Algorithm 4.2 uses the Compound DD i.e. \( \Upsilon_{ewf}(x) \) is \( \Lambda^c(x) \). In other words, for any coded symbol, the probability of choosing degree \( i \) is given by

\[
\Lambda_i^c = \rho_1 \phi_i^{(1)} + (1 - \rho_1) \phi_i^{(2)}
\]  
(4.4)
Figure 4.11: Performance comparisons with EWF codes using Lena with $k = 100$.  

where $\phi_i^{(1)} = 0$ for $i > \alpha_k$. Let $\alpha_1^{ewf}$ and $\Gamma_1^{ewf}$ be the optimum parameters of the EWF code in the minimum-distortion sense for our progressive transmission scenario. For GLTexp, for all $n$, we set $\alpha_1 = \alpha_1^{ewf}$ and $\rho_1 = \Gamma_1^{ewf}$. In addition to GLTexp, we define two other versions of the proposed UEP GLT with a larger size of the parameter set subject to optimization.

- **GLTexpOpt**: This scheme uses the Exponential SD with $\overline{A}_1 + \overline{B}_1 = 1$. It optimizes the set $\{\alpha_1, \rho_1, \overline{A}_1, \overline{C}_1\}$ so that the proposed scheme achieves minimum distortion.

- **GLTexpFullOpt**: This scheme uses the Exponential SD and optimizes the whole set of parameters, i.e., $(\alpha_1, \rho_1, \overline{A}_1, \overline{B}_1, \overline{C}_1)$ so that the proposed scheme achieves minimum distortion.
Figs. 4.11 and 4.12 compare these UEP schemes with $k = 100$ in terms of the average PSNR as a function of $n$, the number of reliably received unerased symbols. As we increase the parameter space of the proposed UEP GLT scheme as described, we observe dramatic improvements in a progressive transmission scenario. For example, GLTexp optimizes only two parameters of the Exponential SD and gives some improvement over the UEP EWF code. Note that both GLTexp and the UEP EWF code optimize two parameters. If we increase the parameter space subject to optimization, the relative gains over the UEP EWF code performance increase. For example, in Fig. 4.11, GLTexpOpt uses four parameters, and GLTexpFullOpt uses five parameters to minimize the source reconstruction distortion, giving 1.3dB
Table 4.4: Optimal parameters of various UEP designs for various $k$ and $\epsilon$.

<table>
<thead>
<tr>
<th>$k=100$</th>
<th>$\epsilon$</th>
<th>$A_1$</th>
<th>$B_1$</th>
<th>$C_1$</th>
<th>$\alpha_1$</th>
<th>$\Gamma_1$</th>
<th>PSNR(dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>UEPEWF</td>
<td>0.4</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>0.5</td>
<td>0.97</td>
<td>29.63</td>
</tr>
<tr>
<td>GLTexpOpt</td>
<td>0.4</td>
<td>-0.62</td>
<td>1.62</td>
<td>1.1</td>
<td>0.24</td>
<td>0.55</td>
<td>30.94</td>
</tr>
<tr>
<td>$k=1000$</td>
<td>$\epsilon$</td>
<td>$A_1$</td>
<td>$B_1$</td>
<td>$C_1'$</td>
<td>$\alpha_1$</td>
<td>$\Gamma_1$</td>
<td>PSNR(dB)</td>
</tr>
<tr>
<td>UEPEWF</td>
<td>0.1</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>0.65</td>
<td>0.95</td>
<td>30.27</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>0.68</td>
<td>0.92</td>
<td>30.78</td>
</tr>
<tr>
<td></td>
<td>0.3</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>0.2</td>
<td>0.15</td>
<td>31.63</td>
</tr>
<tr>
<td>GLTexpOpt</td>
<td>0.1</td>
<td>-0.94</td>
<td>1.94</td>
<td>1.1</td>
<td>0.36</td>
<td>0.52</td>
<td>31.23</td>
</tr>
<tr>
<td></td>
<td>0.2</td>
<td>-0.7</td>
<td>1.7</td>
<td>0.8</td>
<td>0.16</td>
<td>0.55</td>
<td>31.71</td>
</tr>
<tr>
<td></td>
<td>0.3</td>
<td>-0.7</td>
<td>1.7</td>
<td>0.9</td>
<td>0.15</td>
<td>0.55</td>
<td>32.39</td>
</tr>
</tbody>
</table>

and 1.53dB average PSNR gains, respectively, compared to the UEP EWF code when $\epsilon = 0.4$. Table 4.4 shows some of the optimal parameters used to obtain the performance curves in Fig. 4.11 ($k = 100$) as well as some of the results for $k = 1000$.

4.6 Conclusions

DF codes are a type of erasure-correcting code with simple encoding and decoding structures used both for point-to-point communications and for multicasting various types of information that needs UEP. The initial design of such codes is aimed at recovering the entire information block, and therefore might not be the best choice when different parts of the data have different levels of importance, such as image or video files compressed in a progressive or scalable fashion. In this chapter, we introduced a generalization of two major types of UEP LT codes that has a larger set of parameters and hence a more flexible rateless coding scheme. We
also introduced a progressive transmission scheme using this generalized version. We compared the proposed scheme with two other major UEP LT code designs in the literature for the proposed progressive transmission scenario. Simulations show that the proposed coding scheme outperforms these previous schemes by providing improved UEP, URT and UIT properties. As a future work, we plan to investigate the asymptotic analysis of the proposed generalization. Since fountain codes are asymptotically optimal codes, this might open new perspectives and lead us to conclude some important results about the ultimate performance of our progressive transmission system.
Acknowledgement

This chapter, in full, is a reprint of the material that has been submitted for publication in S. S. Arslan, P. C. Cosman, and L. B. Milstein, “Generalized Unequal Error Protection LT codes for Progressive Transmission,” in IEEE Transactions on Image Processing, Jan. 2012. I was the primary author of this publication. The coauthor, Prof. Cosman and Prof. Milstein, directed and supervised the research which forms the basis for this research paper.
Appendix A

Hard Decision Upper Bounds for Coded $M$-Ary Hierarchical Modulation

In this appendix, we start with Equation (A.1) from [30] below, and derive an expression for $P_{d(\beta)}$ for coded hierarchical modulations. We show that the BER ($P_e$) for the HP bits (or LP bits) can be upper bounded by

$$P_e \leq \frac{1}{\delta} \sum_{d^{(\beta)} = d_f}^{\infty} c_{d^{(\beta)}} P_{d^{(\beta)}},$$  

(A.1)

where $\delta$ is the puncturing period, $c_{d^{(\beta)}}$ is the coefficient of the bit Input Weight Enumeration Function (IWEF) of a given code $\beta$, $d_f$ is the free distance of the code, and $P_{d^{(\beta)}}$ is the probability of selecting an incorrect path in the trellis $T$. Note $\delta = 1$ for unpunctured convolutional codes. Throughout this section, BER refers to the...
decoded bit error rate rather than the raw bit error rate.

### A.1 Hard Decision Upper Bound

Assume that the all-zero binary sequence is transmitted and all the bits are going through $BSC(\epsilon)$. Now suppose that the path in $T$ being compared with the all-zero path at some node has distance $n$ from the all-zero path. The probability of selecting the incorrect path is given by

$$P_n = \sum_{k=\frac{n}{2}}^{n} \binom{n}{k} \epsilon^k(1-\epsilon)^{n-k}$$  \hspace{1cm} (A.2)

Now, for simplicity, assume that we have two different symbol groups that corresponds to raw bit error rates of $\epsilon_1$ and $\epsilon_2$. From these $n$ bit positions, we assume $s$ positions have a bit error probability $\epsilon_1$, and $n-s$ positions that have a bit error probability $\epsilon_2$. Then, the average probability of selecting the wrong path in $T$ is given by

$$P_n = \sum_{s=0}^{n} Pr\{s; n\} \psi_s(k \geq \frac{n+1}{2})$$  \hspace{1cm} (A.3)

where $Pr\{s; n\}$ is the probability of having exactly $s$ bit positions that have the bit error probability $\epsilon_1$, and $\psi_s(k \geq \frac{n+1}{2})$ is the probability of selecting an incorrect path when exactly $s$ positions have the bit error probability $\epsilon_1$. Note that

$$Pr\{s; n\} = \binom{n}{s} (1-p)^s p^{n-s}$$  \hspace{1cm} (A.4)
Lemma \( \psi_s(k \geq \frac{n+\kappa}{2}) \) is given by (A.5) with \( \kappa \in \{1, 2\} \).

\[
\psi_s(k \geq \frac{n + \kappa}{2}) = \sum_{k=\frac{n + \kappa}{2}}^{n} \sum_{m=[k-n+s]}^{\min\{k,s\}} \binom{s}{m} \epsilon_1^m (1 - \epsilon_1)^{s-m} \binom{n-s}{k-m} \epsilon_2^{k-m} (1 - \epsilon_2)^{n-s-k+m},
\]

where \([x]^+ = \begin{cases} x & \text{if } x \geq 0, \\ 0 & \text{if } x < 0 \end{cases} \) and \( \kappa = \begin{cases} 1 & \text{if } n \text{ odd}, \\ 2 & \text{if } n \text{ even} \end{cases} \) (A.5)

Proof For a given distance \( n \) path i.e., the distance to the all-zero path is \( n \), assume that \( s \) bit positions have the bit error probability \( \epsilon_1 \) and \( n-s \) positions have bit error probability \( \epsilon_2 \). Now assume \( k \) positions are in error. Of these \( k \) positions, assume that \( m \) positions are chosen from the bits that have a bit error probability \( \epsilon_1 \) and \( k-m \) positions are chosen from the bits that have a bit error probability \( \epsilon_2 \).

The probability of this event happening is given by \( \binom{s}{m} \epsilon_1^m (1 - \epsilon_1)^{s-m} \binom{n-s}{k-m} \epsilon_2^{k-m} (1 - \epsilon_2)^{n-s-k+m} \). We now need to sum over \( m \) and \( k \). Note that \( \binom{s}{m} \) and \( \binom{n-s}{k-m} \) are nonzero when \( 0 \leq m \leq s \) and \( 0 \leq k-m \leq n-s \) and zero otherwise. We have \( m \leq k \), because \( k \) is the total number of errors. Therefore \( m \leq s, m \leq k \Rightarrow m \leq \min\{s,k\} \) and \( k-m \leq n-s, m \geq 0 \Rightarrow \{m \geq k-n+s, m \geq 0\} \Rightarrow \{m \geq [k-n+s]^+\}. \)

Also, since we assume the all-zero path is sent, for the selected path to be in error we need to have \( k \geq \frac{n+1}{2} \) for odd \( n \) and \( k \geq \frac{n}{2} + 1 \) for even \( n \). ■

If we substitute (A.4) and (A.5) in (A.3), we can write the sequence of equations from (A.6) to (A.11) as follows:
\[ P_n = \sum_{s=0}^{n} \binom{n}{s} (1-p)^s p^{n-s} \sum_{k=\frac{s-k}{2}}^{\min\{k,s\}} \sum_{m=\lfloor k-n+s \rfloor^+}^{\min\{k,s\}} \left( \frac{s}{m} \right) \epsilon_1^m (1-\epsilon_1)^{s-m} \left( \frac{n-s}{k-m} \right) \epsilon_2^{k-m} (1-\epsilon_2)^{n-s-k+m} \quad (A.6) \]

\[ = \sum_{k=\frac{s-k}{2}}^{\min\{k,s\}} \sum_{m=\lfloor k-n+s \rfloor^+}^{\min\{k,s\}} (1-p)^s p^{n-s} \left( \frac{s}{m} \right) \epsilon_1^m (1-\epsilon_1)^{s-m} \left( \frac{n-s}{k-m} \right) \epsilon_2^{k-m} (1-\epsilon_2)^{n-s-k+m} \quad (A.7) \]

\[ = \sum_{k=\frac{s-k}{2}}^{\min\{k,s\}} \sum_{m=\lfloor k-n+s \rfloor^+}^{\min\{k,s\}} \Lambda_{k,s,m} + \sum_{k=\frac{s-k}{2}}^{\min\{k,s\}} \sum_{m=\lfloor k-n+s \rfloor^+}^{\min\{k,s\}} \Lambda_{k,s,m} \quad (A.8) \]

\[ = \sum_{k=\frac{s-k}{2}}^{\min\{k,s\}} \sum_{m=\lfloor k-n+s \rfloor^+}^{\min\{k,s\}} \Lambda_{k,s,m} + \sum_{k=\frac{s-k}{2}}^{\min\{k,s\}} \sum_{m=\lfloor k-n+s \rfloor^+}^{\min\{k,s\}} \Lambda_{k,s,m} + \sum_{k=\frac{s-k}{2}}^{\min\{k,s\}} \sum_{m=\lfloor k-n+s \rfloor^+}^{\min\{k,s\}} \Lambda_{k,s,m} \quad (A.9) \]

\[ = \sum_{k=\frac{s-k}{2}}^{\min\{k,s\}} \sum_{m=\lfloor k-n+s \rfloor^+}^{\min\{k,s\}} \Lambda_{k,s,m} + \sum_{k=\frac{s-k}{2}}^{\min\{k,s\}} \sum_{m=\lfloor k-n+s \rfloor^+}^{\min\{k,s\}} \Lambda_{k,s,m} \quad (A.10) \]

\[ = \sum_{k=\frac{s-k}{2}}^{\min\{k,s\}} \sum_{m=\lfloor k-n+s \rfloor^+}^{\min\{k,s\}} (1-p)^s p^{n-s} \left( \frac{s}{m} \right) \epsilon_1^m (1-\epsilon_1)^{s-m} \left( \frac{n-s}{k-m} \right) \epsilon_2^{k-m} (1-\epsilon_2)^{n-s-k+m} \quad (A.11) \]

where \( \binom{a}{b} = 0 \) if \( b > a \). The explanation for each step is given as follows:

- (A.7) to (A.8): The total sum is divided into two double sums. The first double sum assumes \( s \leq n-k < k \). Therefore \( \min\{s,k\} = s \). Also \( [k-n+s]^+ = 0 \) for \( s \leq n-k \). The second double sum assumes \( s \geq n-k+1 \). If \( k \leq s \), then we have \( \min\{s,k\} = k \). Otherwise, we have \( s < k \). Since \( \binom{a}{b} = 0 \) if \( b > a \), if we sum \( m \) up to \( k \) instead of up to \( s \), we will be adding \( k-s \) zeros, so the sum will not change. Also, \( [k-n+s]^+ > 0 \) for \( s \geq n-k+1 \). We show the equivalence of the double sums in the second inequality of (A.8) by plotting the two dimensional area that each double sum covers. This is shown in Fig. A.1.

- (A.8) to (A.9): We added extra 0s to complete the sums. For the first double sum, we have \( s \leq n-k \). Thus, extending the maximum value of \( m \) from \( n-k \)
Figure A.1: Double sums in (A.8) and (A.9) are equivalent.

to $k$ will not make a difference because $\max\{s\} = n - k$. The same argument applies to the second double sum. Also, $\sum_{i=a}^b g(i) = 0$ for $a > b$ where $g(i)$ is any function of $i$.

- (A.9) to (A.10): We combine the double sums.

A.1.1 $n$ odd

Theorem 1.A: $\forall p \in \mathbb{N}^+, n = 2p + 1$, we have $P_n = \tilde{P}_n$ where

$$
\tilde{P}_n = \sum_{k=\frac{n}{2} + 1}^{n} \binom{n}{k} (1-p)\epsilon_1 + p\epsilon_2)^k (1 - ((1-p)\epsilon_1 + p\epsilon_2))^{n-k} \tag{A.12}
$$

This is the same expression given in (A.2), except that we replace $\epsilon$ with $(1-p)\epsilon_1 + p\epsilon_2 = \epsilon_1 - p(\epsilon_1 - \epsilon_2)$.

Proof By expanding (A.12), we obtain

$$
\tilde{P}_n = \sum_{k=\frac{n}{2} + 1}^{n} \binom{n}{k} ((1-p)\epsilon_1 + p\epsilon_2)^k ((1-p)(1-\epsilon_1) + p(1-\epsilon_2))^{n-k} \tag{A.13}
$$
If we use the binomial expansion for (A.13), we obtain (A.14). Let us make the change of variables \( z = s - m \) i.e., \( s = z + m \). Since \( z \in [0, n - k] \), then \( s \in [m, n - k + m] \), so we will have (A.15). This equals \( P_n \) in (A.11), since

\[
\binom{n}{s} m \binom{n-k}{s-m} = \binom{n}{s} \binom{n-k}{s-m}.
\]

\( \Box \)

\[
\bar{P}_n = \sum_{k=\frac{n+1}{2}}^{n} \binom{n}{k} \sum_{m=0}^{k} \binom{k}{m} (1-p)^m \epsilon_1^m \epsilon_2^{k-m} \sum_{z=0}^{n-k} \binom{n-k}{z} (1-p)^z (1-\epsilon_2)^z p^{n-k-z} (1-\epsilon_2)^{n-k-z} \quad (A.14)
\]

\[
= \sum_{k=\frac{n+1}{2}}^{n} \sum_{m=0}^{k} \sum_{z=0}^{n-k+m} (1-p)^z p^{n-z} \binom{n}{k} \epsilon_1^m \epsilon_2^{k-m} \binom{n-k}{s-m} (1-\epsilon_1)^{s-m} (1-\epsilon_2)^{n-k-s+m} \quad (A.15)
\]

**A.1.2 \( n \) even**

**Theorem 1.B:** \( \forall p \in \mathbb{N}, n = 2p \), we have \( P_n = \bar{P}_n \).

**Proof** If \( n \) is even, we have \( k \geq \frac{n}{2} + 1 \). An incorrect path is chosen when the number of errors exceeds \( \frac{n}{2} \). If it equals \( \frac{n}{2} \), the decoder selects one of the paths randomly. Thus, with \( \kappa = 2 \), we have

\[
P_n = \sum_{s=0}^{n} Pr\{s; n\} \psi_s(k \geq \frac{n}{2} + 1) + \frac{1}{2} \mathfrak{P}(n/2) \quad (A.16)
\]

where the \( 1/2 \) comes from the fact that half of the time the decoder incurs an error and \( \mathfrak{P}(n/2) \) is the probability of selecting an incorrect path when the number of errors equals \( n/2 \). Of these \( n/2 \) bit positions, suppose \( m \) bit positions have a bit error probability \( \epsilon_1 \), and for the remaining \( n/2 \) bit positions, \( z \) bit positions have a bit error probability \( \epsilon_1 \). The probability of this event happening is given by

\[
\binom{n/2}{m} \epsilon_1^m \epsilon_2^{n/2-m} \binom{n/2}{z} (1-\epsilon_1)^z (1-\epsilon_2)^{n/2-z}
\]
Let $\phi_{m,z}$ be the conditional probability of choosing a particular $n/2$ bit locations given that $m + z$ bit positions have bit error probability $\epsilon_1$ as described above. Thus, in order to find an expression for $\mathcal{P}(n/2)$, we sum over $m$ and $z$ as follows:

$$\mathcal{P}(n/2) = \sum_{m=0}^{n/2} \sum_{z=0}^{n/2} \phi_{m,z} \binom{n/2}{m} \epsilon_1^m \epsilon_2^{n/2-m} \binom{n/2}{z} (1 - \epsilon_1)^z (1 - \epsilon_2)^{n/2-z}$$

where $\phi_{m,z}$ is given by

$$\phi_{m,z} = \binom{n}{n/2} (1 - p)^{z+m} p^{n-z-m} \quad (A.17)$$

Therefore, using the binomial expansion as shown in Theorem 1.A, we can write the second term in (A.16) as follows

$$\frac{1}{2} \mathcal{P}(n/2) = \frac{1}{2} \binom{n}{n/2} ((1 - p)\epsilon_1 + p\epsilon_2)^{n/2} (1 - ((1 - p)\epsilon_1 + p\epsilon_2))^{n/2}$$

Since the first term in (A.16) follows the same line of proof shown for $\kappa = 1$ (i.e., $n$ is odd and $k \geq \frac{n+1}{2}$), the details are omitted for the $\kappa = 2$ case. ■

Therefore, $\forall d^{(\beta)} \in \mathbb{N}^+$, a more compact expression for $P_{d^{(\beta)}}$ in Equation (A.1) is given by

$$P_{d^{(\beta)}} = \sum_{k=d^{(\beta)}+1+\lceil d^{(\beta)}+1 \rceil/2}^{d^{(\beta)}} \binom{d^{(\beta)}}{k} (\epsilon_0)^k (1 - \epsilon_0)^{d^{(\beta)}-k}$$

$$+ \frac{[d^{(\beta)} + 1]_2}{2} \binom{d^{(\beta)}}{d^{(\beta)}/2} (\epsilon_0)^{d^{(\beta)}/2} (1 - \epsilon_0)^{d^{(\beta)}/2}$$

where $[.]_2$ is modulo two equivalent of the argument and $\epsilon_0 = \epsilon_1 - p(\epsilon_1 - \epsilon_2)$. 
Table A.1: Parameters used in the evaluation of the bound for HP bits for different channel and constellation assumptions.

<table>
<thead>
<tr>
<th>Modulation</th>
<th>Channel</th>
<th>$\epsilon_1$</th>
<th>$\epsilon_2$</th>
<th>$p$</th>
</tr>
</thead>
<tbody>
<tr>
<td>H-4PAM/H-16QAM</td>
<td>AWGN</td>
<td>$Q\left(\sqrt{\frac{8}{1+\alpha^2}}\right)$</td>
<td>$Q\left(\sqrt{\frac{8\gamma\alpha^2}{1+\alpha^2}}\right)$</td>
<td>0.5</td>
</tr>
<tr>
<td>H-4PAM/H-16QAM</td>
<td>Rayleigh</td>
<td>$\frac{1}{2} - \frac{1}{2}\sqrt{\frac{\lambda}{1+\lambda}}$</td>
<td>$\frac{1}{2} - \frac{1}{2}\sqrt{\frac{\alpha^2\lambda}{1+\alpha^2\lambda}}$</td>
<td>0.5</td>
</tr>
<tr>
<td>Concentric H-2/4PSK</td>
<td>AWGN</td>
<td>$Q(\varsigma_1 \cos \theta_1 \sqrt{3\gamma})$</td>
<td>$Q(\varsigma_2 \cos \theta_2 \sqrt{3\gamma})$</td>
<td>0.75</td>
</tr>
</tbody>
</table>

A.2 Simulations

We use two types of codes: (1) RCPC code with memory $M=6$ and $M=4$ given in [30], (2) NASA standard code [47]: $(7, 1/2)$ convolutional code, used also in DVB-T [22]. We tested both AWGN and slowly varying flat Rayleigh fading channels using various hierarchical constellations: H-4PAM, H-16QAM, two concentric hierarchical 2/4 PSK [48]. In Table A.1, we show several parameters used in evaluating the proposed bound for these modulation formats. Note that $\epsilon_1$ and $\epsilon_2$ can be obtained for any channel and any priority bit level for a given constellation topology. In short, for the given simulation parameters, as long as we find the triple ($\epsilon_1$, $\epsilon_2$, $p$), we can calculate the upper bound.

For punctured convolutional codes, the upper bound is given by Inequality (A.1). As noted, using the corresponding $c_{d(\beta)}$, with $\delta = 1$, upper bounds for unpunctured convolutional codes can also be calculated using (A.1). It is enough to sum a finite number of terms in (A.1) to get close bounds [30]. IWEFs of some of the convolutional codes used in our simulations are summarized in Table A.2.
Table A.2: Parameters of the codes. GP: Generator polynomial.

<table>
<thead>
<tr>
<th>Code</th>
<th>Rate</th>
<th>M</th>
<th>GP</th>
<th>δ</th>
<th>$c_d$</th>
</tr>
</thead>
<tbody>
<tr>
<td>CC</td>
<td>1/2</td>
<td>6</td>
<td>[171, 131]</td>
<td>1</td>
<td>$c_{10} = 36, c_{11} = 0, \ldots$</td>
</tr>
<tr>
<td>RCPC</td>
<td>8/9</td>
<td>6</td>
<td>[133, 171, 65]</td>
<td>8</td>
<td>$c_3 = 24, c_4 = 740, \ldots$</td>
</tr>
<tr>
<td>RCPC</td>
<td>2/3</td>
<td>6</td>
<td>[133, 171, 65]</td>
<td>8</td>
<td>$c_6 = 12, c_7 = 280, \ldots$</td>
</tr>
<tr>
<td>RCPC</td>
<td>1/2</td>
<td>4</td>
<td>[13, 29, 17, 27]</td>
<td>8</td>
<td>$c_7 = 32, c_8 = 96, \ldots$</td>
</tr>
<tr>
<td>RCPC</td>
<td>2/3</td>
<td>4</td>
<td>[13, 29, 17, 27]</td>
<td>8</td>
<td>$c_4 = 4, c_5 = 0, \ldots$</td>
</tr>
<tr>
<td>RCPC</td>
<td>2/5</td>
<td>4</td>
<td>[13, 29, 17, 27]</td>
<td>8</td>
<td>$c_8 = 2, c_9 = 34, \ldots$</td>
</tr>
</tbody>
</table>

Figure A.2: Conventional 4PAM ($\alpha = 3$) coded with NASA standard code.
Suppose that \( L \) denotes the number of priority layers that a hierarchical modulation can support. The first simulation assumes an AWGN channel using a conventional H-4PAM \( (L = 2) \) constellation with a single hierarchical parameter \( \alpha = 3 \) (i.e., non-hierarchical modulation). A Monte Carlo simulation result using NASA’s 1/2 convolutional code and H-4PAM as well as the upper bounds suggested in this paper are illustrated in Fig. A.2. We have shown upper bounds using both the first three and the first five terms in (A.1). Next, in Fig. A.3, a rate 8/9 RCPC code with memory \( M=6 \) is used with various \( \alpha \) values using H-16QAM with \( L = 2 \). Note that inphase and quadrature components of a given H-16QAM can be thought of as two independent H-4PAM constellations. We have used only the first 4 terms in (A.1) while evaluating the bounds. Fig. A.3 suggests that the bounds give close
approximations to the simulation results for the SNRs of interest over a reasonable range of the hierarchical parameter set.

**Figure A.4**: Two concentric 2/4 PSKs coded with several RCPC codes with M=4 from [30].

Let us now consider the two concentric 2/4 PSK waveforms \((L = 3)\) coded with RCPC codes with memory M=4 from [30]. We choose \(\theta_2 = \pi/12\) and \(\varsigma_2 = 2 \times \varsigma_1\). Therefore, we have \(\theta_1 = \arccos\left(2 \sin \frac{\pi}{12}\right)\). Let us consider only the HP bit location and the associated information BER in AWGN. Fig. A.4 shows the simulation as well as the suggested upper bounds, based on the information summarized in Table A.1. It also shows the uncoded case both using simulations and the theoretical result, which are in close agreement. Finally, we consider a flat Rayleigh fading channel using a conventional 16QAM constellation. A Monte Carlo simulation result using
Figure A.5: RCPC 2/3 code used in conjunction with H-16QAM under flat Rayleigh fading channel.

A 2/3 RCPC code with M=6 and H-16 QAM, as well as the bounds, are illustrated in Fig. A.5.
Appendix B

Lower Bound on the Performance Improvement of the Proposed System in Chapter 2

Let us denote the total distortion measure $D^j_N(\alpha; A)$ using only the packets in set $A$ and packetization $j \in \{SP, FP\}$ in an $N$-packet error protection scheme. Note that from Equation (2.9) and previous discussion, we have $E[D^{SP}(R^*, R^*, \alpha_3)] := d_0 + D^{SP}_N(\alpha_3; \Omega)$, $E[D^{FP}(R^*, R^*, \alpha_3)] := d_0 + D^{FP}_N(\alpha_3; \Omega)$.

Let $p_{HP}$ and $p_{LP}$ be the probability of having HP and LP packets correct using $R^*$ and $\alpha_3$. The PER gap is defined as $p_{HP} - p_{LP} \triangleq \Delta \gamma$. It can be shown by induction that $(p_{HP})^n - (p_{LP})^n = \Delta \gamma \phi_{n-1}$ where $\phi_n$ satisfies the recursive relation $\phi_n = p_{HP} \phi_{n-1} + (p_{LP})^n$ with the initial condition $\phi_0 = 1$. Note also that $\forall n \in \mathbb{N}, \phi_n \geq 0$ and we define $\phi_n = 0$ for $n < 0$. 
We will show

$$\mathbb{E}[D^{SP}(R^*,R^*,\alpha_3)] - \mathbb{E}[D^{FP}(R^*,R^*,\alpha_3)] = D^{SP}_N(\alpha_3;\Omega) - D^{FP}_N(\alpha_3;\Omega) \geq 0.$$  

Let us find the total distortion measure gap in the first half (i.e. bits in $P'_\cup$)

$$\Delta D_N(P'_\cup) := D^{SP}_N(\alpha_3;P'_\cup) - D^{FP}_N(\alpha_3;P'_\cup)$$

$$= -\sum_{l=1}^{N/2} \prod_{i=1}^l P_i^{(SP)}\Delta_l - \left(-\sum_{l=1}^{N/2} \prod_{i=1}^l P_i^{(FP)}\Delta_l\right)$$

where the probability of having packet $i$ correct for the system using $FP$, $P_i^{(FP)} = p_{HP}$ for $1 \leq i \leq N/2$, $P_i^{(FP)} = p_{LP}$ for $N/2 + 1 \leq i \leq N$. Also for the system using $SP$ and from Fig. 2.2 b) we have $P_i^{(SP)} = p_{HP}$ if $i$ is odd, $P_i^{(SP)} = p_{LP}$ if $i$ is even.

\textbf{B.1} $N$ is divisible by four

First, assume that $N$ is divisible by four. We have

$$\Delta D_N(P'_\cup) = p_{HP}\Delta_1 - p_{HP}\Delta_1 + p_{HP}^2\Delta_2 - p_{HP}p_{LP}\Delta_2 \cdots - p_{HP}^{N/4}p_{LP}^{N/4}\Delta_{N/2}$$

$$= 0 + p_{HP}(p_{HP} - p_{LP})\Delta_2 + p_{HP}^2(p_{HP} - p_{LP})\Delta_3$$

$$+p_{HP}^3(p_{HP}^2 - p_{LP}^2)\Delta_4 + p_{HP}^4(p_{HP}^3 - p_{LP}^3)\Delta_5 \cdots$$

$$\cdots + p_{HP}^{N/4}(p_{HP}^{N/4-1} - p_{LP}^{N/4-1})\Delta_{N/2} + p_{HP}^{N/4}(p_{HP}^{N/4} - p_{LP}^{N/4})\Delta_{N/2}$$

$$= \sum_{n=1}^{N/4} p_{HP}^n\Delta_{\gamma(p_{n-2}\Delta_{2n-1} + p_{n-1}\Delta_{2n})} \quad \text{(B.1)}$$
Let us consider the second half (i.e. bits in $P'_U$).

$$\Delta \mathcal{D}_N(P''_U) = \mathcal{D}_N^{SP}(\alpha_3; P'_U) - \mathcal{D}_N^{FP}(\alpha_3; P''_U)$$

$$= - \sum_{l=\frac{N}{2}+1}^{N} \prod_{i=1}^{l} a_i^{(SP)} \Delta_l - \left( - \sum_{l=\frac{N}{2}+1}^{N} \prod_{i=1}^{l} a_i^{(FP)} \Delta_l \right)$$

$$= + \frac{N}{2} p_{HP} P_{LP} \Delta_{\frac{N}{2}+1} + \frac{N}{2} p_{HP} P_{LP} \Delta_{\frac{N}{2}+2} + \cdots + \frac{N}{2} p_{HP} P_{LP} \Delta_N$$

$$- \frac{N}{2} p_{LP} \Delta_{\frac{N}{2}+1} - \frac{N}{2} p_{LP} \Delta_{\frac{N}{2}+2} - \cdots - \frac{N}{2} p_{LP} \Delta_N$$

$$= \sum_{n=1}^{\frac{N}{2}} p_{HP}^{n-1} p_{LP}^{2n-1} \Delta \gamma \phi_{n-1} \left( \Delta_{\frac{N}{2}+2n-1} + p_{LP} \Delta_{\frac{N}{2}+2n} \right)$$

Therefore, since $P'_U \cap P''_U = \emptyset$ and $\mathcal{D}_N^{SP}(\alpha_3; .)$ is a finitely additive measure, we have

$$\mathbb{E}[\mathcal{D}^{SP}(R^*, R^*, \alpha_3)] - \mathbb{E}[\mathcal{D}^{FP}(R^*, R^*, \alpha_3)] := \Delta \mathcal{D}_N(\Omega) = \mathcal{D}_N^{SP}(\Omega) - \mathcal{D}_N^{FP}(\Omega)$$

$$= \mathcal{D}_N^{SP}(P'_U) + \mathcal{D}_N^{SP}(P''_U) - \mathcal{D}_N^{FP}(P'_U) - \mathcal{D}_N^{FP}(P''_U) = \Delta \mathcal{D}_N(P'_U) + \Delta \mathcal{D}_N(P''_U)$$

$$= \sum_{n=1}^{\frac{N}{2}} p_{HP}^{n-1} p_{LP} \Delta \gamma \phi_{n-2} \Delta_{2n-1} + p_{LP} \Delta_{2n}$$

$$+ \sum_{n=1}^{\frac{N}{2}} p_{HP}^{n-1} p_{LP}^{2n-1} \Delta \gamma \phi_{n-1} \left( \Delta_{\frac{N}{2}+2n-1} + p_{LP} \Delta_{\frac{N}{2}+2n} \right)$$

$$= \sum_{n=1}^{N} p_{HP}^{n-1} p_{LP} \Delta \gamma \left( \phi_{n-2} \Delta_{2n-1} + \phi_{n-1} \Delta_{2n} + \cdots \right)$$

$$\geq 0.$$
B.2 \( \mathbf{N} \) is not divisible by four

When \( \mathbf{N} \) is not divisible by four, a similar discussion will follow except the Eqtions (B.1) and (B.2) will be given by

\[
\Delta \mathcal{D}_N(P') = \sum_{n=1}^{N/2} p_{HP}^n \Delta \gamma (\Delta_{N/2-n} \Delta_{2n-1} + \Delta_{N/2-n} \Delta_{2n}) + p_{LP}^{N/2} \Delta \phi_{N/2-n} \Delta_{N/2} \quad \text{(B.5)}
\]

\[
\Delta \mathcal{D}_N(P'') = \sum_{n=1}^{N/2} p_{HP}^{N/2-n} p_{LP}^{2n} \Delta \gamma \phi_{N/2-n} \Delta_{N/2-n} (\Delta_{N/2+n} + \Delta_{N/2+2n+1}) + p_{LP}^{N/2} \Delta \phi_{N/2-n} \Delta_{N/2+n+1} \quad \text{(B.6)}
\]

Thus, \( \Delta \mathcal{D}_N(P') + \Delta \mathcal{D}_N(P'') \geq 0 \) since \( \{\Delta_i\}_{i=1}^N \geq 0 \). ■
Bibliography


