MAP SOURCE-CONTROLLED CHANNEL DECODING FOR
IMAGE TRANSMISSION SYSTEM USING CPFSK AND RING
CONVOLUTIONAL CODES

A Thesis in
Electrical Engineering
by
Srijidtra Mahapakulchai

© 2004 Srijidtra Mahapakulchai

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

May 2004
The thesis of Srijidtra Mahapakulchai was reviewed and approved* by the following:

Robert E. Van Dyck  
Assistant Professor of Electrical Engineering  
Thesis Adviser  
Co-Chair of Committee

John J. Metzner  
Professor of Computer Science and Engineering  
Co-Chair of Committee

David J. Miller  
Associate Professor of Electrical Engineering  
Co-Chair of Committee

John F. Doherty  
Associate Professor of Electrical Engineering

Frank R. Deutsch  
Professor of Mathematics

W. Kenneth Jenkins  
Professor of Electrical Engineering  
Head of the Department of Electrical Engineering

*Signatures are on file with the Graduate School.
Abstract

Recently, many novel information technologies involve the transmission of imagery over noisy channels such as satellite and wireless mobile channels. In general, a low-bit-rate image transmission system requires an outstanding image encoder that provides both an excellent quality for the reconstructed image and a high compression ratio. However, the resulting compressed bit stream becomes highly sensitive to channel noise. There have been several approaches to add error resiliency to an image coder. In this work we concentrate on the use of joint source-channel (JSC) methods. In particular, source-controlled channel decoding, based on the residual redundancy in MPEG-4 compressed imagery, is considered. Here an embedded zerotree wavelet (EZW) algorithm is used to generate a compressed bit stream, which is then passed through a ring convolutional encoder (CE) and a CPFSK modulation system. The overall polynomial encoder is the combination of the CE and the continuous phase encoder (CPE). The source-controlled channel decoder exploits the source transition matrix (STM) of the zerotree symbols in computing the combined trellis branch metrics, giving MAP decoding. Simulation results for both the AWGN and flat Rayleigh fading channels show the performance improvement compared to conventional ML decoding.

Moreover, we investigate the design of trellis codes using ring convolutional codes and CPFSK for MAP decoding. The goal is to further improve the performance of the image transmission system when MAP decoding is used. Conventionally a ring convolutional encoder was designed for maximum likelihood (ML) decoding over the
AWGN channel. The criteria is to find a code that has the maximum of the minimum squared Euclidean distance. Without considering the source information, this criteria may not be suitable for the case of using MAP decoding. In this work the STM is used in the design of trellis codes for a particular source and value of noise power $N_0$. The “Lena” and “Barbara” images for both single quantization and multi-quantization mode are used.
Table of Contents

List of Tables ................................................................. x

List of Figures ................................................................. xix

List of Acronyms .............................................................. xxvi

Acknowledgments .............................................................. xxix

Chapter 1. Introduction ......................................................... 1

1.1 Image Transmission Over Noisy Channels ......................... 1

1.2 Trellis-Coded CPM and Ring Convolutional Codes ............... 4

1.3 Organization of the Dissertation .................................. 6

1.4 List of Contributions .................................................. 7

Chapter 2. Image Compression and Transmission ....................... 9

2.1 Image Compression ...................................................... 9

2.1.1 The SPIHT Algorithm ............................................ 14

2.1.2 MPEG-4 Embedded Zerotree Wavelet Coder .................. 15

2.1.2.1 Lowest Frequency Subband Coding and Decoding ..... 16

2.1.2.2 Higher Frequency Subband Coding and Decoding ..... 18

2.1.2.3 Packet Structures of HFS ................................. 21

2.2 Image Transmission Over a Noisy Channel ....................... 24

2.2.1 Residual Redundancy ............................................ 25
# Table of Contents

Chapter 3. Convolutional Codes ................................................. 31

3.1 Convolutional Encoder .................................................. 32
   3.1.1 Distance structure of Convolutional Codes ................. 36
   3.1.2 Equivalence of Convolutional Encoders ..................... 38
   3.1.3 Catastrophic Error Propagation of Convolutional Codes ... 40

3.2 Convolutional Decoder .................................................. 43

3.3 Design of Convolutional Codes ....................................... 49

3.4 Ring Convolutional Codes ............................................. 50

Chapter 4. Continuous Phase Modulation (CPM) and Trellis-Coded Modulation (TCM) .............................................. 53

4.1 The General Description of CPM ...................................... 53
   4.1.1 General Definition ............................................. 54
   4.1.2 The Representation of CPM .................................. 55
   4.1.3 Spectral Characteristics of CPM .............................. 56

4.2 The Alternative Representation of CPM .............................. 58

4.3 Trellis-Coded Modulation for CPM ................................... 62

Chapter 5. MPEG-4 Imagery Transmission System and Source-Controlled Channel Decoding ............................................. 68

5.1 System Description ..................................................... 68

5.2 Channel Models ....................................................... 70
5.2.1 Additive White Gaussian Noise (AWGN) Channel 70
5.2.2 Fading Channel 72
5.3 MAP Decoding for the AWGN Channel 73
5.3.1 MAP Decoding Derivation for the CPFSK System 74
5.3.2 MAP Decoding Derivation for the CPFSK system with an external convolutional encoder 78
5.4 MAP Decoding for Fading Channel 86

Chapter 6. Design of Ring Convolutional Trellis Codes for MAP Decoding 88
6.1 Trellis Ring Convolutional Coded CPFSK 88
6.2 Code Design Description for MAP Decoding 90
6.3 Suboptimal Code Design Description 94
6.4 Code Search Results 96

Chapter 7. Symbol Error Rate Simulation Results 106
7.1 Channel Coding Simulation Results 106
7.1.1 Examples and Simulation Results for Uncoded CPFSK 107
7.1.2 Examples and Simulation Results for Trellis-coded CPFSK 110
7.1.3 Examples and Simulation Results for Fading Channel 116
7.2 Mismatched Channel Simulation Result 118

Chapter 8. Image Transmission Results Part I: Performance Comparison Between ML and MAP Decoding 121
8.1 The Simulation Results from using Single and Multi-quantization Modes for the “Lena” Image ........................................ 122
  8.1.1 The Results from using Single Quantization Mode .......... 123
  8.1.2 The Results from using Multi-quantization Mode .......... 133
  8.1.3 Mismatched channel ............................................. 137

8.2 Simulation Results from using Single and Multi-quantization Modes for the “Barbara” Image ................................. 140
  8.2.1 The Results from using Single Quantization Mode .......... 140
  8.2.2 The Results from using Multi-quantization Mode .......... 149
  8.2.3 Mismatched channel ............................................. 153

Chapter 9. Image Transmission Results Part II: Performance Comparison Between ML and MAP codes ................................. 156
  9.1 Simulation Results from Single Quantization Mode ............ 158
    9.1.1 Simulation Results for the “Lena” Image .................... 158
    9.1.2 Simulation Results for the “Barbara” Image .............. 161
  9.2 Simulation Results from Multi-Quantization Mode ............. 164
    9.2.1 Simulation Results for the “Lena” Image ................. 164
    9.2.2 Simulation Results for the “Barbara” Image ............ 167

Chapter 10. Conclusions and Future Work .............................. 170
  10.1 Conclusions ...................................................... 170
  10.2 Future Work .................................................... 171
References
List of Tables

3.1 All possible codewords generated by the convolutional encoder block diagram shown in 3.2. .......................................................... 34

3.2 The maximum free distance of systematic encoders and nonsystematic encoders ................................................................. 40

3.3 The branch metrics computed at each time interval \((k - 1)T \leq t \leq kT\). .......................................................... 46

3.4 The best rate 1/2 binary convolutional encoders and their free (Hamming) distance. ................................................................. 49

6.1 \((A)(B)\) Initial image coding results for 4 state codes. (a) Lena and (b) Barbara. Multi-quantization mode. 200 image trials per data point. ... 96

6.2 The best rate 1/2 nonsystematic ring convolutional encoders over \(Z_4\) for 4-ary CPFSK with \(h=1/4\). \(S_t\) denotes the total number of states in the overall encoder. YT is the best systematic code from [15], while ML is our best code. * indicates the search was incomplete. ......................... 97

6.3 The rate 1/2 nonsystematic ring convolutional encoders over \(Z_4\) for 4-ary CPFSK with \(h=1/4\) for MAP decoding. The STM in Eq. (6.13) is used. \(S_t\) denotes the total number of states in the overall encoder. .................. 99

6.4 The rate 1/2 nonsystematic ring convolutional encoders over \(Z_4\) for 4-ary CPFSK with \(h=1/4\) for MAP decoding. STM in Eq. (6.14) is used. \(S_t\) denotes the total number of states in the overall encoder. .................. 101
6.5 The rate 1/2 nonsystematic ring convolutional encoders over $\mathbb{Z}_4$ for 4-ary CPFSK with $h=1/4$ for MAP decoding. STM in Eq. (6.15) is used. $S_t$ denotes the total number of states in the overall encoder. . . . . . . . . 103

6.6 The rate 1/2 nonsystematic ring convolutional encoders over $\mathbb{Z}_4$ for 4-ary CPFSK with $h=1/4$ for MAP decoding. STM in Eq. (6.16) is used. $S_t$ denotes the total number of states in the overall encoder. . . . . . . . . 105

7.1 Symbol error rate result on mismatch channel. * means symbol error rate for match channel. . . . . . . . . . . . . . . . . . . . . . . . . . . . 119

7.2 Symbol error rate results on a mismatched channel. * Symbol error rate for matched channel. . . . . . . . . . . . . . . . . . . . . . . . . . . . 119

7.3 Symbol error rate results on a mismatched channel. *Symbol error rate for a matched channel. . . . . . . . . . . . . . . . . . . . . . . . . . . . 119

7.4 Symbol error rate results on a mismatched channel. *Symbol error rate for a matched channel. . . . . . . . . . . . . . . . . . . . . . . . . . . . 120

8.1 The average PSNR of the reconstructed “Lena” image using ML and MAP decoding for the AWGN channel. Strategy 1 (Top ten rows) and strategy 2 (Bottom ten rows) are used. . . . . . . . . . . . . . . . . . . . . . . . . . . . 126

8.2 The average PSNR of the reconstructed “Lena” image using ML and MAP decoding for the Rayleigh channel. Strategies 1 (Top eight rows) and 2 (Bottom eight rows) are used. . . . . . . . . . . . . . . . . . . . . . . . . . . . 130
8.3 The average PSNR of the reconstructed “Lena” image using ML and MAP decoding for AWGN channel. Strategies 1 (Top ten rows) and 2 (Bottom ten rows) are used. .............................. 136

8.4 The average PSNR of the reconstructed “Lena” image using ML and MAP decoding for a Rayleigh channel. Strategies 1 and 2 are used. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 136

8.5 The average PSNR of the reconstructed “Lena” image using MAP decoding with mismatched noise power $N_o$ for the Gaussian channel. The asterisk indicates the matched case. Strategy 1 (Top three rows) and 2 (Bottom three rows) are used. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 138

8.6 The average PSNR of the reconstructed “Lena” image using MAP decoding with mismatched noise power $N_o$ for the Rayleigh fading channel. The asterisk indicates the matched case. Strategy 1 (Top three rows) and 2 (Bottom three rows) are used. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 139

8.7 The average PSNR of the reconstructed “Lena” image using MAP decoding with mismatched noise power $N_o$ for the Gaussian channel. The asterisk indicates the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 139

8.8 The average PSNR of the reconstructed “Lena” image using MAP decoding with mismatched noise power $N_o$ for the Rayleigh channel. The asterisk indicates the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 140

8.9 The average PSNR of the reconstructed “Barbara” image using ML and MAP decoding for the AWGN channel. Strategies 1 and 2 are used. . . 143
8.10 The average PSNR of the reconstructed “Barbara” image using ML and MAP decoding for a Rayleigh channel. Strategies 1 and 2 are used.

8.11 The average PSNR of the reconstructed “Barbara” image using ML and MAP decoding for AWGN channel. Strategies 1 (Top eight rows) and 2 (Bottom eight rows) are used.

8.12 The average PSNR of the reconstructed “Barbara” image using ML and MAP decoding for the Rayleigh channel. Strategies 1 (Top seven rows) and 2 (Bottom seven rows) are used.

8.13 The average PSNR of the reconstructed “Barbara” image using MAP decoding with mismatched noise power $N_o$ for a Gaussian channel. The asterisk indicates the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used.

8.14 The average PSNR of the reconstructed “Barbara” image using MAP decoding with mismatched noise power $N_o$ for the Rayleigh fading channel. The asterisk indicates the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used.

8.15 The average PSNR of the reconstructed “Barbara” image using MAP decoding with mismatched noise power $N_o$ for the Gaussian channel. The asterisk indicates the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used.
8.16 The average PSNR of the reconstructed “Barbara” image using MAP decoding with mismatched noise power $N_o$ for the Rayleigh channel. The asterisk indicates the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used. . . . . . . . . . . . . . . . . . . . . 155

9.1 The comparison of reconstructed image average PSNR between our sub-optimal code (SC) and Yang and Taylor’s equivalent code (YTE) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. AWGN channel. . 156

9.2 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) $[D^3 + 3D^2 + 2D + 3, 2D^2 + 3D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. AWGN channel. . 158

9.3 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) $[D^3 + 3D^2 + 2D + 3, 2D^2 + 3D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. AWGN channel. . . . . . . . . . . . . . . . . . . . . 159
9.4 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([D^3 + 3D^2 + 2D + 3, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. Rayleigh channel.

9.5 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([D^3 + 3D^2 + 2D + 3, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. Rayleigh channel.

9.6 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([D^3 + 3D^2 + 2D + 1, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. AWGN channel.

9.7 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([D^3 + 3D^2 + 2D + 1, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. AWGN channel.
9.8 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([D^3 + 3D^2 + 2D + 1, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. Rayleigh channel.

9.9 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([D^3 + 3D^2 + 2D + 1, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. Rayleigh channel.

9.10 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([D^3 + 2D^2 + D + 3, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. AWGN channel.

9.11 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([D^3 + 2D^2 + D + 3, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. AWGN channel.
9.12 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([D^3 + 2D^2 + D + 3, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. Rayleigh channel.

9.13 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([D^3 + 2D^2 + D + 3, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. Rayleigh channel.

9.14 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([3D^3 + 3D^2 + 2D + 3, 2D^2 + D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. AWGN channel.

9.15 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([3D^3 + 3D^2 + 2D + 3, 2D^2 + D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. AWGN channel.
9.16 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([3D^3 + 3D^2 + 2D + 3, 2D^2 + D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. Rayleigh channel.

9.17 The comparison of reconstructed image average PSNR between our sub-optimal MAP code (MAP-MAP) \([3D^3 + 3D^2 + 2D + 3, 2D^2 + D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. Rayleigh channel.
# List of Figures

2.1 First stage decomposition of an image using a separable technique. ........................................ 10  
2.2 A spatial orientation tree (SOT) structure. ................................................................. 12  
2.3 The scanning pattern. ................................................................................................. 13  
2.4 The block diagrams of the MPEG-4 EZW encoder (a) and decoder (b). .................. 17  
2.5 The position of quantized coefficient X and its neighborhood. .............................. 17  
2.6 The basic packet structure for the HFS ................................................................. 22  
2.7 The packet structure for single quantization mode; MAP decoding of  
    the zerotree symbols is possible because of the separation of the zerotree  
    symbols from the nonzero values. .............................................................................. 23  
2.8 The packet structure for multi-quantization mode. Note that we have  
    only zerotree symbols in this case. .............................................................................. 23  
3.1 Convolutional encoder block diagram ......................................................................... 32  
3.2 Convolutional encoder block diagram ......................................................................... 34  
3.3 State diagram of convolutional encoder in Example 3.1. ........................................ 35  
3.4 Trellis diagram of the convolutional encoder in Example 3.1. ............................... 36  
3.5 Trellis diagram of convolutional encoder in Example 3.1. ........................................ 38  
3.6 The state diagram of the rate 1/2 catastrophic encoder, \( G(D) = [1 + 
    D^3, 1 + D + D^2] \). .................................................................................................. 41  
3.7 The block diagram of the convolutional encoder \( G(D) = [D + 1, D^2 + 
    D + 1] \). .................................................................................................................... 46
3.8 The trellis diagram labeled with branch metrics at $k = 1$ and the survivor path after the first iteration.

3.9 The trellis diagram labeled with branch metrics at $k = 2$ and the survivor paths after the second iteration.

3.10 The survivor paths after the fifth iteration and the decision path (solid line).

3.11 The block diagram of the encoder $G(D) = \left[ D, 1 + 3D \right]$.

4.1 (a) the MSK phase tree and (b) the MSK phase trellis

4.2 The MSK state diagram.

4.3 Spectral density for MSK and 4-ary CPFSK with $h=1/4$.

4.4 The MSK tilted phase tree (a) and the MSK tilted phase trellis (b).

4.5 The block diagram of a CPE and MM system.

4.6 The block diagram of a TCM system.

4.7 Decomposition model of $M$-ary full response CPM with $h=K/M$. Feedback-free CPE (left) and conventional CPE (right).

5.1 System block diagram for the image transmission system using a ring convolutional encoder and continuous phase modulation scheme.

5.2 Decomposition model of $M$-ary full response CPM with $h=K/M$.

5.3 The phase trellis diagram of 4-ary CPFSK, $h=1/4$.

5.4 The block diagram of an equivalent encoder of rate $2/4$.

5.5 The phase trellis diagram of the block diagram shown in Figure 5.4.

5.6 The block diagram of overall encoder.
5.7 The trellis diagram and the phase trellis diagram of the block diagram shown in Figure 5.6. .................................................. 82

5.8 The block diagram of the overall encoder, $G_{all}$. .................... 84

6.1 Symbol error rate (SER) vs. signal-to-noise ratio $E_s/N_0$ for the STM given by Eq. (6.13). The best ML codes use ML decoding, while the suboptimal MAP codes use MAP decoding. The parameter $m$ is the overall number of states in the trellis. ...................... 98

6.2 Symbol error rate vs. $E_s/N_0$ for the source transition matrix (STM) given by Eq. (6.14) for using the best ML and suboptimal MAP codes. The parameter $m$ is the overall number of states in the trellis. .............. 100

6.3 Symbol error rate vs. $E_s/N_0$ for the source transition matrix (STM) given by Eq. (6.15) for using the best ML and suboptimal MAP codes. The parameter $m$ is the overall number of states in the trellis. .............. 102

6.4 Symbol error rate vs. $E_s/N_0$ for the source transition matrix (STM) given by Eq. (6.16) for using the best ML and suboptimal MAP codes. The parameter $m$ is the overall number of states in the trellis. .............. 104

7.1 System block diagram for a Markov source consisting of a quaternary input sequence using a ring convolutional encoder and a continuous phase modulation scheme. ............................................. 106

7.2 Symbol Error Rate (SER) vs. $E_s/N_0$. Not using source transition probability matrix (solid line), Using source transition probability matrix (dashed line). ............................................. 108
7.3 Symbol Error Rate (SER) vs. $E_s/N_0$. No source transition probability matrix (solid line). Source transition probability matrix (dashed line). 109

7.4 The block diagram of overall encoder for Example 7.1 111

7.5 The trellis diagram of rate 1/2 encoder $[2D + 1, 1]$ with quaternary CPFSK, $h=1/4$ 111

7.6 The block diagram of overall encoder for Example 7.2 113

7.7 The trellis diagram of rate 1/2 encoder $[D + 1, 2D + 1]$ with quaternary CPFSK, $h=1/4$ 114

7.8 Symbol error rate vs. $E_s/N_0$ for the source transition matrix (STM) given by Eq. (7.2). The parameter $m$ is the overall number of states in the trellis. 115

7.9 Symbol error rate vs. $E_s/N_0$ for the source transition matrix (STM) given by Eq. (7.2). The parameter $m$ is the overall number of states in the trellis. 115

7.10 Symbol error rate vs. $\bar{\gamma}_b = (1 + \bar{\gamma})E_s/(\gamma N_0)$ for the source transition matrix (STM) given by Eq. (7.2) The parameter $m$ is the overall number of states in the trellis. 117

7.11 Symbol error rate vs. $\bar{\gamma}_b = (1 + \bar{\gamma})E_s/(\gamma N_0)$ for the source transition matrix (STM) given by Eq. (7.2) The parameter $m$ is the overall number of states in the trellis. 117

8.1 The original “Lena” image. 124
8.2 The reconstructed “Lena” image for a noiseless channel. PSNR is 31.83 dB at a CR of 30.80:1.

8.3 The reconstructed “Lena” image using ML decoding with strategy 1.
Note that PSNR = 26.22 dB at SNR = 3.5 dB. The Gaussian channel is used.

8.4 The reconstructed “Lena” image using MAP decoding with strategy 1.
Note that PSNR = 26.55 dB at SNR = 3.5 dB. The Gaussian channel is used.

8.5 The reconstructed “Lena” image using ML decoding with strategy 2.
Note that PSNR = 26.35 dB at SNR = 4 dB. The Gaussian channel is used.

8.6 The reconstructed “Lena” image using MAP decoding with strategy 2.
Note that PSNR = 26.96 dB at SNR = 4 dB. The Gaussian channel is used.

8.7 The reconstructed “Lena” image using ML decoding with strategy 1.
Note that PSNR = 25.08 dB at $\gamma_b = 7.5$ dB. The Rayleigh channel is used.

8.8 The reconstructed “Lena” image using MAP decoding with strategy 1.
Note that PSNR = 25.35 dB at $\gamma_b = 7.5$ dB. The Rayleigh channel is used.

8.9 The histogram of the improvements (dB) of 500 trials of the “Lena” image. Strategy 1 and the Rayleigh channel at $\gamma_b = 7.5$ dB are used.

8.10 The histogram of the improvements (dB) of 500 trials of the “Lena” image. Strategy 2 and the Rayleigh channel at $\gamma_b = 7.5$ dB are used.

8.11 The original “Lena” image.

8.12 The reconstructed “Lena” image for a noiseless channel. The PSNR is about 29.33 dB at a CR of 30.87:1.
8.13 The original “Barbara” image. ........................................ 142
8.14 The reconstructed “Barbara” image for a noiseless channel. PSNR is about 27.52 dB at a CR of 20:1. ........................................ 142
8.15 The reconstructed “Barbara” image using ML decoding with strategy 1. Note that PSNR = 21.23 at SNR = 3.5 dB. The Gaussian channel is used. 144
8.16 The reconstructed “Barbara” image using MAP decoding with strategy 1. Note that PSNR = 21.50 at SNR = 3.5 dB. The Gaussian channel is used. ........................................ 144
8.17 The reconstructed “Barbara” image using ML decoding with strategy 2. Note that PSNR = 20.28 at SNR = 3.5 dB. The Gaussian channel is used. 145
8.18 The reconstructed “Barbara” image using MAP decoding with strategy 2. Note that PSNR = 20.59 at SNR = 3.5 dB. The Gaussian channel is used. ........................................ 145
8.19 The reconstructed “Barbara” image using ML decoding with strategy 1. Note that PSNR = 21.69 at \( \gamma_b = 7.5 \) dB. The Rayleigh channel is used. 147
8.20 The reconstructed “Barbara” image using MAP decoding with strategy 1. Note that PSNR = 21.93 at \( \gamma_b = 7.5 \) dB. The Rayleigh channel is used. 147
8.21 The histogram of the improvements (dB) of 500 trials of the “Barbara” image. Strategy 1 and the Rayleigh channel at \( \gamma_b = 7.5 \) dB are used. 148
8.22 The histogram of the improvements (dB) of 500 trials of the “Barbara” image. Strategy 2 and the Rayleigh channel at \( \gamma_b = 7.5 \) dB are used. 148
8.23 The original “Barbara” image. ........................................ 150
8.24 The reconstructed “Barbara” image for a noiseless channel. PSNR is about 26.81 dB at a CR of 19.96:1. 150

10.1 Serial concatenated coding system using a ring convolutional code and trellis-coded CPFSK with iterative decoding. 172
List of Acronyms

ARQ      Automatic Repeat Request
AWGN     Additive White Gaussian Noise
BER      Bit Error Rate
BSMS     Binary Symmetric Markov Source
CE       Convolutional Encoder
CPE      Continuous Phase Encoder
CPM      Continuous Phase Modulation
CPFSK    Continuous Phase Frequency Shift Keying
CRC      Cyclic Redundancy Check
DPCM     Differential Pulse Code Modulation
DWT      Discrete Wavelet Transform
EZW      Embedded Zerotree Wavelet
FEC      Forward Error Correction
FIR      Finite Impulse Response
GCD      Greatest Common Divisor
HFS      Higher Frequency Subband
HMM      Hidden Markov Model
IDWT     Inverse Discrete Wavelet Transform
IZ       Isolated Zerotree
JSC      Joint Source Channel
LFS      Lowest Frequency Subband
<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>LIP</td>
<td>List of Insignificant Pixels</td>
</tr>
<tr>
<td>LIS</td>
<td>List of Insignificant Sets</td>
</tr>
<tr>
<td>LSP</td>
<td>List of Significant Pixels</td>
</tr>
<tr>
<td>MAP</td>
<td>Maximum <em>a Posteriori</em></td>
</tr>
<tr>
<td>ML</td>
<td>Maximum Likelihood</td>
</tr>
<tr>
<td>MPEG</td>
<td>Moving Picture Experts Group</td>
</tr>
<tr>
<td>MSE</td>
<td>Mean Square Error</td>
</tr>
<tr>
<td>NEG</td>
<td>Negative Significance</td>
</tr>
<tr>
<td>POS</td>
<td>Positive Significance</td>
</tr>
<tr>
<td>PSD</td>
<td>Power Spectral Density</td>
</tr>
<tr>
<td>PSK</td>
<td>Phase Shift Keying</td>
</tr>
<tr>
<td>PSNR</td>
<td>Peak Signal-to-Noise Ratio</td>
</tr>
<tr>
<td>QAM</td>
<td>Quadrature Amplitude Modulation</td>
</tr>
<tr>
<td>QMF</td>
<td>Quadrature Mirror Filter</td>
</tr>
<tr>
<td>RC</td>
<td>Raised Cosine Frequency Pulse</td>
</tr>
<tr>
<td>RCPC</td>
<td>Rate Compatible Punctured Convolutional</td>
</tr>
<tr>
<td>REC</td>
<td>RE Ctangular Frequency Pulse</td>
</tr>
<tr>
<td>SAMMSE</td>
<td>Sequence-based Approximate Minimum Mean Square Error</td>
</tr>
<tr>
<td>SED</td>
<td>Squared Euclidean Distance</td>
</tr>
<tr>
<td>SER</td>
<td>Symbol Error Rate</td>
</tr>
<tr>
<td>SNR</td>
<td>Signal-to-Noise Ratio</td>
</tr>
<tr>
<td>SOT</td>
<td>Spatial Orientation Tree</td>
</tr>
<tr>
<td>Abbr.</td>
<td>Description</td>
</tr>
<tr>
<td>-------</td>
<td>---------------------------------</td>
</tr>
<tr>
<td>SOVA</td>
<td>Soft Output Viterbi Algorithm</td>
</tr>
<tr>
<td>SPIHT</td>
<td>Set Partitioning In Hierarchical Trees</td>
</tr>
<tr>
<td>STM</td>
<td>Source Transition Matrix</td>
</tr>
<tr>
<td>TCM</td>
<td>Trellis-Coded Modulation</td>
</tr>
<tr>
<td>VAL</td>
<td>Value</td>
</tr>
<tr>
<td>VZTR</td>
<td>Valued Zerotree Root</td>
</tr>
<tr>
<td>VQ</td>
<td>Vector Quantization</td>
</tr>
<tr>
<td>ZTR</td>
<td>Zerotree Root</td>
</tr>
</tbody>
</table>
Acknowledgments

This dissertation could not be completed without an exceptional supervision of the author’s advisor, Professor Robert E Van Dyck. The author wishes to express her sincerest appreciation for his beneficial advice, his generous assistance, his constant patience and encouragement toward the conclusion of this work. It has been an invaluable experience working under his guidance. In addition, the author would like to show a special gratitude to her co-advisor, Professor John J. Metzner for all his constant assistance. Her great thankfulness also goes to Professors David J. Miller for his kind help and useful suggestion. The author also wants to individually thank Professor John F. Doherty and Professor Frank Deutsch for their useful suggestions and comments. All comments from all committees allow the author to immensely improve this dissertation.

More importantly, she would like to acknowledge Royal Thai Government and engineering department of Kasetsart university for the opportunity and financial support.

Her grateful feelings go to her dedicating father and caring mother. Their unconditional love and endless support give her the strength to finish this work. She wants to particularly thank her wonderful husband and son for their love, patience and support. Last but not least, she also thanks her elder brother, younger sister, and friends for their encouragement.
Chapter 1

Introduction

1.1 Image Transmission Over Noisy Channels

Many revolutionizing technologies in the communication industry these days result from research in the area of image transmission over a wireless channel. In general, an image allows one to enhance the ability to communicate more effectively. However, image transmission over a noisy channel typically requires an image encoding scheme with error resilient properties, a channel encoding scheme with noise immunity, an efficient use of bandwidth, and image and channel decoding processes with high performance.

There has been a significant amount of research on image coding techniques for many years. Recently, due to the variety of applications, new techniques aim for not only state-of-the-art performance, but also desirable properties, such as error resilience, sufficient scalability, and coding in some region of interest. The embedded zerotree wavelet (EZW) algorithm introduced by Shapiro [22] and set partitioning in hierarchical trees (SPIHT) proposed by Said and Pearlman [46] have become popular image coding techniques, since they not only achieve excellent performance with low computational complexity but also provide an embedded coding feature. This feature allows one to transmit the sequence of compressed bits in order of importance so that an encoding process can be terminated at any time to achieve the target rate. Since a compressed bit stream generated by an excellent image coder is very sensitive to a noisy channel,
some modification in the coding technique must be considered. For an EZW coder there are many different approaches to enhance error resilience (see [57], [58], [59], [60], [49], and [23]). Firstly, the compressed bit stream is segmented into a sequence of packets with a fixed or variable length. These packets can be decoded individually. Each packet includes a resynchronization marker that can be used to locate the beginning of each packet. Secondly, the spatial orientation trees of wavelet coefficients are decomposed into a number of groups. Each group is coded separately, so that a bit error will affect only a single group. Moreover, entropy coding, such as arithmetic coding, is sometimes removed to reduce the noise sensitivity of the compressed bit stream. Thirdly, the coding of the lowest frequency subband (LFS) is done separately from the coding of the higher frequency subband (HFS) in order to allow one to easily apply unequal error protection for both sets of subbands.

Instead of modifying an image coder, one can simply add error correcting techniques such as forward error correction (FEC) codes. An FEC scheme allows one to correct a certain number of errors, but at the expense of using extra bits. The number of these bits should be selected wisely by considering the system performance, computation complexity, and the compression ratio. Moreover, automatic repeat request (ARQ) protocols are useful when lost or unrecoverable packets occur. These protocols allow the encoder to retransmit such packets. However, ARQ may introduce a transmission delay that cannot be tolerated in some applications such as audio and video transmission.

An alternative approach known as joint source-channel (JSC) coding is developed for practical use in a noisy channel situation. JSC coding can be found in several
forms, such as the modification of the source encoder/decoder structure based on channel knowledge, a design of channel coding for a particular source, and a design of a source-controlled channel decoder based on the use of residual redundancy of a source. A discussion about JSC coding techniques for image and video transmission can be found in [47]. In this work we focus our interest on the latter two forms. The methods of using residual source redundancy to design both channel coding and a source-controlled channel decoder have been investigated in a number of papers [55], [56], [17], [18], [19], [20], [21], and [23]. Hagenauer [55] proposed a source-controlled decoder named APRISOVA, which is the modification of the soft output Viterbi decoding algorithm (SOVA) by exploiting the \textit{a priori} information of a source. Ruf and Hagenauer [56] illustrate the performance improvement for an image transmission system by using APRISOVA. Sayood and Borkenhagen [17] showed that a significant improvement can be obtained by using the residual redundancy in differential pulse code modulation (DPCM) as error protection. Extension of this work can be found in [18]. Here the image transmission system consisting of DPCM coding and a convolutional coding was considered.

Phamdo and Farvardin [19] considered two types of optimal detection designed for a discrete Markov source. Firstly, sequence maximum \textit{a posteriori} (MAP) detection is used to determine the most probable transmitted sequence given the observed sequence. Another method is known as an instantaneous MAP detection, which is used to determine the most probable symbol at time $n$ given the previous observations up to time $n$. A discrete memoryless channel was assumed. For a binary Markov channel, the design of joint source-channel coding for memoryless and Gauss-Markov sources can be found in
Miller and Park [21] developed an optimal source decoder known as a sequence-based approximate minimum mean-squared error decoder (SAMMSE). The framework of discrete hidden Markov model (HMM) was used to characterize a system consisting of a source encoder and a memoryless channel. Van Dyck [23] employed this method of using the discrete HMM [21] to estimate the LFS and HFS coefficients generated by an MPEG-4 encoder for both a binary symmetric channel and Gilbert-Elliot [44, 45] channel.

1.2 Trellis-Coded CPM and Ring Convolutional Codes

Unlike conventional channel coding, trellis-coded modulation (TCM) [43] can achieve a significant performance improvement without augmenting the bandwidth used. The design of a trellis-coded modulation system using continuous phase modulation (CPM) is motivated by the bandwidth efficient feature of CPM. Moreover, in a fading environment, the requirement of using a digital modulation scheme that provides constant envelope signals such as CPM becomes intensively crucial. In the 1980s, the design of trellis-coded CPM by using a binary convolutional encoder (CE) and CPM over a Gaussian channel was profoundly studied in [8], [9], and [10], where the common objective was to determine an optimal CE that maximizes the minimum squared Euclidean distance (SED). Note that this optimal code provides the minimum symbol error rate at high signal-to-noise ratios (SNR). The drawback of using a convolutional encoder over the binary field is the requirement of a mapper. This results in receiver complexity and also hinders the analysis based on a single underlying algebraic structure.
There is much research [32], [11], [12], [15], and [16] interested in using convolutional codes over a ring to design a trellis-coded modulation scheme. These works generally focus on multilevel modulation techniques such as $M$-ary phase shift keying (PSK), $M$-ary CPFSK, and $M$-ary partial response CPM. In 1989, Massey and Mittelholzer [32] pointed out that a convolutional encoder over the ring $\mathbb{Z}_M$ of integers modulo $M$, is a natural way to apply $M$-ary phase modulation without using a mapper. Baldini and Farrell [11] showed a performance improvement of a TCM system using both systematic and nonsystematic ring convolutional codes. The performance of their system was compared with an uncoded $M$-ary phase modulation system. Ugrelidze and Shavgulidze [12] designed new trellis codes based upon rate 1/2 $M$-ary ring convolutional codes and continuous phase frequency shift keying (CPFSK) signals.

In 1988, Rimoldi [14] proposed a method to decompose CPM with rational modulation index $h = K/P$ into a continuous phase encoder (CPE) and a memoryless modulator (MM), where $K$ and $P$ are relatively prime positive integers. The CPE is in the form of a CE over the ring $\mathbb{Z}_P$ of integers modulo $P$. Thus, a convolutional encoder over $\mathbb{Z}_P$ seems to be a natural way to combine with the CPE without using a mapper. With this alternative form of CPM, Yang and Taylor [15] consider a recursive systematic CE over the ring $\mathbb{Z}_M$ of integers modulo $M$ with $M$-ary CPFSK, where $h=1/M$; they obtain an overall (CE+CPE) noncatastrophic encoder. Since the CE and CPE are in the same algebraic structure, a reduction in the numbers of trellis state of the overall system is possible. Rimoldi and Li [16] also use a recursive systematic CE over $\mathbb{Z}_M$ with partial response CPM; they employ feedback of the state of the CPE to the CE. In both
cases, the performance is improved compared to coded schemes using binary CE codes with the same decoder complexity.

1.3 Organization of the Dissertation

We categorize our task here into two main topics. Firstly, we are interested in developing a source-controlled channel decoder for a trellis-coded CPM system. In particular, this coded-modulation system consists of a convolutional code over the ring $\mathbb{Z}_M$ of integers modulo $M$ and $M$-ary continuous phase frequency shift keying (CPFSK). The MPEG-4 source encoder is selected to be our image compression technique. Moreover, we consider the case when mismatch channel occurs. Secondly, we investigate the design of ring convolutional trellis codes for MAP decoding. The source transition matrix (STM) of an image is used in the design process.

In Chapters 2, 3 and 4, we briefly review some necessary background. Image compression techniques based on the embedded zerotree wavelet (EZW) algorithm are discussed in Chapter 2. Moreover, an error resilient image transmission method using residual redundancy of a source is described. In Chapter 3, the definition and some useful properties of convolutional codes and ring convolutional codes are given. Chapter 4 is focused on the description of CPM and TCM systems. We emphasize the design of TCM system using a convolutional code and a CPM scheme.

In Chapter 5, we develop an MPEG-4 imagery transmission system using source-controlled channel decoding. The design of the decoder is described for both the additive white Gaussian noise (AWGN) and flat Rayleigh fading channels. In Chapter 6, the design of ring convolutional trellis codes for MAP decoding is described. The code search
process is done for the $512 \times 512$ gray scale “Lena” and “Barbara” images. These images can be obtained at [68]. In order to compare the performance between the receivers using ML and MAP decoding, we first consider the channel coding simulation shown in Chapter 7. Without using the real image in the system, the source bit stream is generated by using the knowledge of the source transition matrix (STM). The STM is computed from zerotree symbol sequence, which generated from a zerotree scan process. Please refer to [23] for the detail of computing the STM. The results are shown for the case of uncoded and trellis coded CPFSK over Gaussian and flat Rayleigh fading channels. In Chapter 8, we discuss the performance comparison between ML and MAP decoding in term of PSNR using both images. The compressed image bit stream is generated using two quantization modes of the MPEG-4 encoder: single quantization mode and multi-quantization mode. In Chapter 9, we present the performance comparison between ML and MAP codes and Finally, conclusions and future work are discussed in Chapter 10.

1.4 List of Contributions

- We implement a trellis-coded modulation system using a nonsystematic (polynomial) ring convolutional code and CPFSK modulation system. The development of the system is based on the design of the trellis-coded CPFSK system using a systematic ring convolutional code proposed in [15].

- We develop an MPEG-4 imagery transmission system using the implemented trellis-coded CPFSK and source-controlled channel decoding. The decoding exploits residual source redundancy from zerotree symbols in the HFS of an image to
enhance an error resilience of the system. The work here can be viewed as an extension of the work in [23].

- We investigate the performance of the proposed system over both the AWGN and a flat Rayleigh fading channel for both the “Lena” and the “Barbara” image. Moreover, we consider two strategies for decoding a received packet [23].

- We implement the noncatastrophic test module for nonsystematic ring convolutional code. The test is based on the theorems which appeared in [32] and [27].

- We investigate the performance of the proposed system when the mismatch channel occurs.

- We investigate the design of non-systematic ring convolutional codes for ML decoding. We found that $d_{free}^2$ of our best non-systematic ring codes is larger than that of codes found in [12].

- We investigate the design of ring convolutional trellis-coded CPFSK system for MAP decoding. The suboptimal ring convolutional codes are determined for a particular source transition matrix (STM). In this work, we consider four STMs computed from the “Lena” and “Barbara” image for both the single quantization and multi-quantization mode.
Chapter 2

Image Compression and Transmission

2.1 Image Compression

In this work, we are interested in one type of image compression technique known as the embedded zerotree wavelet (EZW) algorithm. To have a good understanding, let us discuss the detail of each main component of this algorithm. Firstly, the name “wavelet” comes from the use of the wavelet transformation operating on an image input. The output is the set of coefficients which can be used in representing the input in the form of wavelet basis functions. Because of its remarkable properties such as decorrelation efficiency and multiresolution capability, this transformation has become increasingly popular in many image and video compression applications. For digital images, we utilize a type of wavelet transformation called a discrete wavelet transform (DWT). The DWT can be described as a special type of hierarchical subband coding system. This coding system decomposes a signal using a set of filters known as a “filter bank” into frequency subbands by iterated filtering and downsampling techniques. The DWT can be constructed by two-channel filter banks consisting of low-pass and high-pass FIR filters cascaded by halved down sampling. Note that the downsampling process represents the word “scale” in wavelet analysis. The large scale means coarse, where small means fine. Since an image is a two-dimensional signal, its row and column components
can be transformed separately or otherwise nonseparately. Note that for the nonseparable case, one would require the use of 2-D low-pass and high-pass filters, resulting in better performance, but more computational complexity in the system. In our case, the transformation process is performed separately on the rows (horizontal) and columns (vertical) of an image. Figure 2.1 illustrates generation of the four subbands ($LL_1$, $LH_1$, $HL_1$, and $HH_1$) after the first stage decomposition. For the name of subband, the first letter indicates the type of filters used for row inputs, corresponding to the scanning of an image in horizontal direction. The second letter is for the column part. The subband $LL_1$ is then decomposed in the next stage creating the other four subbands ($LL_2$, $LH_2$, $HL_2$, and $HH_2$). The decomposition process is recursively performed until the desired

Fig. 2.1. First stage decomposition of an image using a separable technique.
final stage is reached. The representation of all subbands is usually done as the decom-
position of the image itself. For instance, if the size of the original image is $256 \times 256$, the size of each first subband is $128 \times 128$. Similarly, the size of second stage subbands are $64 \times 64$ and so on. The concept of the hierarchical structure in the wavelet domain plays the crucial part in the success of the EZW coding algorithm. To fully comprehend this concept, the following review is necessary. The tree structure shown in Figure 2.2 comprises a set of wavelet coefficients, which correspond to the same spatial orientation across the subbands at the different scales. With the exception of the lowest frequency subband (LFS) indicated as $LL_3$, the beginning of a tree structure occurs from each coefficient, named as the “root”, in subbands at the coarsest scale such as $LH_3$, $HL_3$, and $HH_3$. Each root or sometimes called “parent” in the subbands at coarser scales relates to four pixels called “children” in the similarly oriented subband at the next finer scale. Inductively, one of these four nodes can also become a parent of the next four coefficients in the subband at the next finer scale and so on. Moreover, for a given root, the set of all coefficients at finer scales is known as the “descendants.” For a given child, the set of all parents in a tree is called “ancestors.” We know that in general, the energy of the images is more concentrated in the LFS than in higher frequency subbands (HFS). It is observed that the variance of the image’s wavelet coefficients decreases as we move along a tree from the coarsest to the finest scale of subbands. It is often found that if a parent node is set to zero by comparing with some threshold, then all of its child nodes are likely to be zero as well. A tree that has this characteristic is called a zerotree. Note that images that contain many zerotree structures can be effectively compressed, since only zerotree roots are necessary, not the entire zerotree structure, which can be
predicted to be zero at the decoder. It is obvious that a high compression ratio can then be achieved.

The concept of embedded coding is similar to that of a binary finite-precision representation of the real numbers. The representation of a real number by a sequence of binary digits can be more accurate if more digits are added to the right of the binary point. For an image compression application, this coding scheme allows the transmission of a sequence of digits representing the coarser subband wavelet coefficients first, and then followed by refining information bits from the finer subbands. This means that a rough image can be obtained first, and the image is getting better and better as more bits are received. The original EZW algorithm was proposed by Shapiro [22] in 1993. In this work, an image is decorrelated by 9-tap symmetric quadrature mirror filters (QMF). The wavelet coefficients then are scanned for encoding a significant map in such a way...
that parents must be scanned before children. Figure 2.3 shows the scanning pattern for a 3-scale subband decomposition. Note that within a given subband, each coefficient is scanned before any coefficient in the next subband. There are two types of passes, dominant and subordinate. The encoding then begins with a dominant pass process, meaning that each wavelet coefficient is assigned to be one of the four symbols, positive significance (POS), negative significance (NEG), zerotree root (ZTR), or isolated zero (IZ). For a given threshold $T$, a wavelet coefficient $X$ is said to be significant if the magnitude is above $T(|X| \geq T)$, and it will be identified as POS if it is positive or NEG if it is negative. The wavelet coefficient regarding as insignificant is encoded as ZTR if all its descendants are also insignificant or IZ if at least one of its descendants is significant. If ZTR is found, its descendants do not need to be encoded further along this dominant pass, since from the decoder point of view, they will be predicted as zero.
After a dominant pass is completed, a significant coefficient is moved to the subordinate list for refinement encoding. It is assigned one of two symbols, “1” if $|X| > T + T/2$ or “0” if $|X| < T + T/2$. Note that the reconstructed value of a significant wavelet coefficient is the mid-point of the interval $[T + T/2, 2T]$ if “1” is received, while the interval $[T, T + T/2]$ is used for symbol “0”.

Before the second dominant and subordinate passes begin, the prior significant wavelet coefficients assigned during first dominant pass are set to zero. The update threshold usually set to be $T/2$. The process is then repeated until the target rate or distortion measure is met.

2.1.1 The SPIHT Algorithm

Set partitioning in hierarchical trees (SPIHT) image compression algorithm was proposed by Said and Pearlman [46] in 1996. At the same compression ratio, this algorithm not only allows one to achieve better performance in term of PSNR, compared to the original EZW algorithm, but also preserves the embedded coding structure. The concept of the SPIHT algorithm is based on the progressive transmission method introduced by DeVore, Jawerth and Lucier [50]. Since a coefficient with larger magnitude leads to more reduction of the mean squared error (MSE), it should be transmitted first. Note that if the value of a coefficient is represented in its binary form, the most significant bits should be conveyed first. The tree structure described in the original EZW algorithm is called a spatial orientation tree (SOT) here.

In the encoding process, all wavelet coefficients are partitioned into subsets corresponding to the zerotree structure. Please refer to the detail of each subset in [46].
These subsets are then passed through the significance test process defined as

\[
S[T(i,j)] = \begin{cases} 
1, & \max_{(k,l) \in T(i,j)} |c_{i,j}| \leq 2^n \\
0, & \text{otherwise,}
\end{cases}
\] (2.1)

where \(T(i,j)\) is either a set of nodes or a single node. A significance test is crucial, and it is done during each coding pass by the process called sorting pass. In this process, the subsets and some single nodes are moved into three lists, named the list of insignificant sets (LIS), the list of insignificant pixels (LIP), and the list of significant pixels (LSP), corresponding to their significant test results. After the sorting pass, all significant coefficients in the LSP are refined later in a process called a refinement pass.

### 2.1.2 MPEG-4 Embedded Zerotree Wavelet Coder

MPEG-4 EZW is a version of the EZW algorithm proposed by the MPEG-4 committee in [48] and [49]. Like the original EZW and SPIHT algorithms, the encoding process begins with a transformation of an image by a DWT. In this case, we decompose an input image with a 5-level DWT using Daubechies (9,15) tap biorthogonal filters. Here, the wavelet coefficients of the lowest (coarsest) frequency subband (LFS) are coded separately from those of the higher (finer) frequency subbands (HFSs). This allows unequal error protection. Also, the number of spatial and signal-to-noise ratio (SNR) scalability levels is more flexible. The transformed coefficients in the LFS are encoded by using a scalar quantizer followed by differential pulse code modulation (DPCM). We will discuss this in detail in the next section. The encoding of the wavelet coefficients in the HFS is done by using successive approximation quantization process combined
with the zerotree scanning. The basic block diagram of the MPEG-4 EZW encoder and decoder are shown in Figure 2.4. At the decoder, the received bits corresponding to the LFS and HFSs are then decoded separately. The inverse DPCM followed by inverse scalar quantization are applied to those bits corresponding to the LFS. The inverse successive approximation quantization and zerotree scanning are performed on those bits corresponding to the HFSs.

2.1.2.1 Lowest Frequency Subband Coding and Decoding

In the LFS, the wavelet coefficients are quantized using scalar quantization, and then this is followed by a differential pulse code modulation (DPCM) process. The method performs the prediction of a quantized coefficient $X$ from three other quantized coefficients in its neighborhood, $A$, $B$ and $C$ as shown in Figure 2.5. Then the predicted value $\hat{X}$ is subtracted from the coefficient $X$. The residual coefficient is then transmitted. The 2-D DPCM algorithm can be described as follows [49].

\[
\begin{align*}
\text{If } ( |A - B| < |A - C| ), \quad &
\text{then } \hat{X} = C \\
\text{else } \hat{X} = A.
\end{align*}
\]

The residual $= X - \hat{X}$.

Note that in the MPEG-4 standard, all residual coefficients are offset by their minimum value, which results in only positive values; they are further coded by an adaptive arithmetic encoder. In this work, instead, we shift all residual coefficients by the number of
Fig. 2.4. The block diagrams of the MPEG-4 EZW encoder (a) and decoder (b).

Fig. 2.5. The position of quantized coefficient X and its neighborhood.
quantization levels. This change allows us to eliminate the need for sending the offset value since it is known a priori. Moreover, we do not perform arithmetic coding in the LFS. For a 512 × 512 image with five levels of decomposition, we obtain the size of the LFS of 16 × 16 coefficients. This subband requires only 1,792 bits for fixed length codeword with seven bit indices. Removing the arithmetic coding does not lead to a significant increase in bit rate. In fact, the difference is about 100-200 bits/image [23]. These extra bits also allow the encoded bit stream to be less sensitive to channel noise.

At the decoder, all received coefficients are shifted back by the number of quantization levels. Then the inverse DPCM algorithm [48] is used to retrieve the actual coefficients from their corresponding residual coefficients as described below.

\[
\text{If } ( |A - B| < |A - C| ),
\]
\[
\text{then } \hat{X} = C
\]
\[
\text{else } \hat{X} = A.
\]
\[
X = \hat{X} + \text{The residual.}
\]

In conclusion, the inverse quantization is performed on those actual coefficients, thereby reconstructing the wavelet coefficients in the LFS.

### 2.1.2.2 Higher Frequency Subband Coding and Decoding

In the MPEG-4 standard, the coding of the HFS can be categorized into two major modes, single quantization, and multi-quantization. In the single quantization mode, all wavelet coefficients are quantized only once with a multi-level quantizer. The
bit allocation of the HFSs depends on the wavelet decomposition level. Note that the sequence of bits produced from this mode does not have the embedded code property, since the most significant bits of all wavelet coefficients are not transmitted first. A tree structure begins with each coefficient (root) in the three subband at the coarsest scale \((HL_3, LH_3, \text{ and } HH_3\) in Figure 2.2) with its descendants corresponding to the same spatial orientation (SOT). These SOTs are then scanned in order and the coefficients are classified into one of four groups: zerotree root (ZTR), isolated zerotree (IZ), valued zerotree root (VZTR) and value (VAL). When a coefficient is smaller than the given threshold, it is a ZTR if all its descendants are also smaller than the threshold, and it is an IZ if at least one of its descendants is bigger than the threshold. Otherwise, it is assigned to be either VZTR or VAL. A VZTR represents a coefficient that is bigger than the given threshold, but all its descendants are smaller than the threshold. If at least one of its descendants is bigger than the threshold, it is identified as VAL. When the coefficient is a VAL or IZ, the corresponding four coefficients form four new trees, and the process is recursively applied for each new tree until an original tree is completed. The encoding process is then applied to the next tree.

Unlike the single quantization mode, the multi-quantization mode is the version of the original EZW algorithm [46]. In this mode, all wavelet coefficients are compared to a sequence of descending threshold values. In each dominant pass, each wavelet coefficient, as well as all of its descendants, are compared to a given threshold value and assigned one of four symbols, zerotree root (ZTR), isolated zero (IZ), valued zerotree root (VZTR) or value (VAL). This is done using the same rule explained in the single quantization mode. Note that in the case of ZTR and VZTR, all descendants will not be processed.
further in the current scanning pass. During scanning, the values of the coefficients mapped to a VZTR and VAL symbol must be reduced by the compared threshold value, while the value of those mapped to a ZTR or IZ symbol remain the same for the next scanning pass. Before the next scanning pass, the threshold value must be updated, typically decreased by a half of the current threshold. The process is recursively applied until some target rate or required distortion is achieved. Since many coefficients are not coded in each pass, a high compression ratio can be obtained. Typically, the transmitted bit stream, consisting of an initial threshold value and a sequence of those four symbols, is entropy coded using an adaptive arithmetic encoder. In our case, they may not be arithmetically encoded. This topic is discussed later.

At the decoder, the initial threshold value and the zerotree symbols are extracted from the incoming bit stream. Initially all coefficients are set to zero. If the VZTR or VAL symbol is received, the corresponding coefficient is increased by the corresponding threshold value (initial threshold value in the first decoding pass). If the ZTR or IZ symbol is received, the corresponding coefficients remains the same value (zero for the first decoding pass). Note that for the ZTR and VZTR symbols, all of their descendants will be skipped for this decoding pass. The threshold value must be updated for the next decoding pass according to the scheme used in the encoding process. Finally, the inverse DWT (IDWT) is applied to the reconstructed wavelet coefficients, resulting in the reconstructed image.
2.1.2.3 Packet Structures of HFS

When source coding with a high compression ratio such as EZW coding is employed, this coding becomes very sensitive to a noisy channel. In fact, a single bit error can lead to the destruction of the entire decoding process. This problem leads to a significant amount of research with the goal to increase the error resilience of source coding. Sherwood and Zeger [57] consider a concatenated coding system comprising an inner convolutional code and an outer cyclic redundancy check (CRC). Here, the bit stream produced by EZW encoder is segmented into 200-bit packets. Each packet is then passed through this concatenated coding system. Man, Kossentini and Smith [58] modify the EZW algorithm developed by Shapiro [22] and Said [46] to improve the error resilience with some cost of compression ratio. The concept is to classify the coding bit sequence into subsequences. Each subsequence can be then protected differently using rate compatible punctured convolutional (RCPC) codes. Creusere [59] introduces a new method by partitioning the wavelet coefficients into a number of groups and then independently encoding each group by using EZW encoder. In this approach, a bit error will affect only a single group, improving the visual quality of a reconstructed image. Roger and Cosman [60] extend this approach by segmenting the output of a wavelet zerotree coder into fixed-length packets that can be individually decoded. The reason is to eliminate the error propagation from one packet to another. The arithmetic coding also is removed to enhance the error resilience. Van Dyck [23] develops the use of residual redundancy in the compressed bit stream to achieve the error resilience of an MPEG-4 EZW coder. Here, variable length packets are created from the higher frequency subband, and they
are independently decodable. Moreover, he removes the arithmetic coding used in the lowest frequency subband. The packet structures taken from [23] are reviewed. The basic structure of each packet illustrated in Figure 2.6 consists of compressed image bits; zerotree symbols and non-zero values, the cyclic redundancy check (CRC) and sufficient header information for independent packet decoding. The header contains twenty bits of resynchronization marker, ten bits of the location of the first spatial orientation tree (SOT) in an image, and eight bits for the number of the SOTs in a packet. Note that this configuration [23] is used for single quantization mode. For the multi-quantization mode, ten bits are required for the number of the SOTs. The cyclic redundancy check (CRC) is four bits long. The resynchronization marker can be used to locate the starting point of the next packet when synchronization is lost. The starting SOT use to locate the position of the first coefficient in each packet. By decomposing $512 \times 512$ image five times, we end up with four $16 \times 16$ subbands at coarsest scale. One of them is the lowest frequency subband (LFS). Thus we need ten bits to represent the starting SOT which ranges from 1 to 768 ($3 \times 16 \times 16$). Also, we assume that the encoder and decoder have

<table>
<thead>
<tr>
<th>Resynch Marker</th>
<th>Starting SOT</th>
<th>Number of SOTs</th>
<th>Compressed Image Data</th>
<th>CRC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Zerotree Symbols &amp; Non-zero Values</td>
<td></td>
</tr>
</tbody>
</table>

Fig. 2.6. The basic packet structure for the HFS
Fig. 2.7. The packet structure for single quantization mode; MAP decoding of the zerotree symbols is possible because of the separation of the zerotree symbols from the nonzero values.

<table>
<thead>
<tr>
<th>Resynch Marker</th>
<th>Starting SOT</th>
<th>Number of SOTs</th>
<th>Number of Zerotrees</th>
<th>Zerotree Symbols</th>
<th>Non-zero Values</th>
<th>CRC</th>
</tr>
</thead>
</table>

Fig. 2.8. The packet structure for multi-quantization mode. Note that we have only zerotree symbols in this case.

the knowledge about the scanning order. In our case, we use scan pattern from left to right and then from top to bottom.

The zerotree symbols and the non-zero values are alternated. Figures 2.7 and 2.8 illustrate the optional packet structure for single quantization and multi-quantization mode, respectively. In both cases, the extra bits are added to the header for containing the number of zerotree symbols. We use eight bits for single quantization mode and ten bits for multi-quantization mode. For single quantization mode, the zerotree symbols are separated from the non-zero values. This allow us to perform MAP decoding to the zerotree symbols. There will be no non-zero values when multi-quantization mode (bit plane encoding) is used. Please refer to [23] for more complete detail of packetization.
2.2 Image Transmission Over a Noisy Channel

As previously mentioned in section 2.1.2.3, the use of a packetization technique with the compressed image bit stream possibly reduces the effect of fading, interference and other channel noises, which are commonly found on a wireless channel. Another possible way to further improve the quality of image transmission system is to utilize the combination of source and channel coding methods. Note that the task of a channel coding is to add extra bits, or redundancy, to the bit stream for protection purposes, while a source coding attempts to remove all the source redundancy to achieve a desirable bit rate. According to Shannon’s separation theorem [4], the design of the source and channel coding processes can be done separately while still achieving an optimal system. However, this theorem assumes that the source coder must remove all the source redundancy and the channel coder must correct all errors generated by a channel. Moreover, it is also assumed that the transmission rate is below the channel capacity. The implementation of such coders becomes impractical due to the need to limit the complexity and delay. In practice, a source coding process produces a compressed bit stream which probably contains a significant amount of redundancy, and a channel coding method may not succeed in eliminating all channel errors. Under these circumstances, the design of a joint source-channel (JSC) coding system becomes more attractive. As mentioned in [47], there are many approaches to a JSC design problem. In this work, we are interested in techniques based on residual redundancy of the source.
2.2.1 Residual Redundancy

In this section, the measurement of residual redundancy is discussed. This value can be used to approximately indicate the performance improvements. Sayood and Borkenhagen [17] studied redundancy in differential pulse code modulation (DPCM) system. The error correcting capability index $I$ [17] was used as the indicator of performance improvement. Alajaji, Phamdo, Farvardin, and Fuja [53] pointed that the total residual redundancy $\rho_T$ can be written as the combination between the redundancy in the form of a nonuniform distribution $\rho_D$ and in the form of memory $\rho_M$. The total redundancy $\rho_T$ is defined as the difference between a fixed rate $R$ bits/sample and the minimum rate (entropy rate of a stochastic process $H(I)$) of the output indexed sequence $I$. Note that $H(I)$, representing the minimum number of bit/sample, is defined [4] by

$$H(I) = \lim_{n \to \infty} \frac{1}{n} H(I_1, I_2, \ldots, I_n),$$  \hspace{1cm} (2.2)

where $H(I_1, I_2, \ldots, I_n)$ is the joint entropy of the stochastic process \{I_i\}. If the random variables $I_1, I_2, \ldots, I_n$ of the process are independent but not identically distributed, we obtain

$$H(I_1, I_2, \ldots, I_n) = \sum_{i=1}^{n} H(I_i),$$  \hspace{1cm} (2.3)

where $H(I_i)$ is the entropy of a discrete random variable $I_i$. It is obvious that if these random variables are independent and identically distributed (i.i.d.), then the entropy rate $H(I)$ is $H(I_1)$. By using chain rule, if $H(I)$ is a stationary stochastic process
(containing memory), we can express the joint entropy $H(I_1, I_2, \ldots, I_n)$ as \[4\]

$$H(I_1, I_2, \ldots, I_n) = \sum_{i=1}^{n} H(I_i | I_{i-1}, \ldots, I_1), \quad (2.4)$$

where $H(I_i | I_{i-1}, \ldots, I_1)$ is the conditional entropy. Thus, in this case, the entropy rate $H(I)$ is $\lim_{n \to \infty} H(I_n | I_{n-1}, \ldots, I_1)$. The proof is given by theorem 4.2.3 in \[4\]. For a stationary (first order) Markov chain with stationary distribution $\mu$ whose components are the stationary probabilities of each state, and a transition probability matrix $P$ whose elements are denoted by $p_{ij}$, we obtain

$$H(I) = \lim_{n \to \infty} H(I_n | I_{n-1}) = H(I_2 | I_1) = \sum_{ij} \mu_i p_{ij} \log \frac{1}{p_{ij}}. \quad (2.5)$$

**Example 2.1.** Consider a four state stationary (first order) Markov source characterized by the source transition probability matrix $P$ shown in Eq. (2.6).

$$P = \begin{bmatrix}
0.8 & 0.1 & 0.05 & 0.05 \\
0.1 & 0.7 & 0.1 & 0.1 \\
0.03 & 0.07 & 0.8 & 0.1 \\
0.04 & 0.06 & 0.1 & 0.8
\end{bmatrix}. \quad (2.6)$$

The probabilities of being in the particular states 0, 1, 2 and 3 are denoted by $\mu_0$, $\mu_1$, $\mu_2$, and $\mu_3$, respectively. These stationary probabilities can be computed by solving the equation $\mu P = \mu$, where $\mu = [\mu_0 \ \mu_1 \ \mu_2 \ \mu_3]$. Since we also know that $\mu_0 + \mu_1 + \mu_2 + \mu_3 =$
1, we obtain $\mu_0 = 0.25$, $\mu_1 = 0.25$, $\mu_2 = 0.25$, and $\mu_3 = 0.25$. By using Eq.(2.5), we can compute the entropy rate of this Markov source; it is equal to 1.102 bits/symbol. For comparison purposes, let us consider the source transition probability matrix of “Lena image” found in [23] as

$$P = \begin{bmatrix} 0.0389 & 0.1001 & 0.5172 & 0.3438 \\ 0.0414 & 0.1774 & 0.4227 & 0.3585 \\ 0.0487 & 0.1668 & 0.4086 & 0.3759 \\ 0.0516 & 0.2327 & 0.4142 & 0.3015 \end{bmatrix}.$$  (2.7)

In this case, we obtain the stationary probabilities $\mu_0 = 0.04786$, $\mu_1 = 0.18836$, $\mu_2 = 0.41839$ and $\mu_3 = 0.34539$. The entropy rate is 1.711 bits/symbol. By assuming that these two Markov sources produce the output indexed sequences at the same fixed rate $R$, we observe that the residual redundancy for the case of Lena image Markov source is less than that of the previous case. Note that the residual redundancy defined earlier is computed as $R - H(I)$.

### 2.2.2 Source-Controlled Channel Decoding

Instead of attempting to eliminate the residual redundancy, we can exploit it as implicit channel coding to help combat channel noise. The process of using the residual redundancy begins with the analysis of source bit probability, also known as “a priori” information. Usually, the Markov model is used to characterize source bit correlation in the form of a source transition probability matrix (STM). These probabilities are then used in the decoding technique. Hagenauer [55] introduced the technique called
source-controlled channel decoding, which is a modification of the soft output Viterbi decoding algorithm (SOVA). Here the \textit{a priori} information of the sources transitions is used in the branch metrics (APRI-SOVA) to create a MAP decoder. Let the information bit and the coded bits of a convolutional code with rate $1/N$ be denoted as $u_k$ and $x_k = (x_{k,1}, x_{k,2}, \ldots, x_{k,N})$, respectively. The branch metric $\lambda_k$ can be expressed as

$$
\lambda_k = \sum_{n=1}^{N} x_{k,n} L_{c_{k,n}} y_{k,n} + u_k L(u_k),
$$

(2.8)

where

$$
L_{c_{k,n}} y_{k,n} = \log \frac{p(y_{k,n}|x_{k,n} = 0)}{p(y_{k,n}|x_{k,n} = 1)}.
$$

(2.9)

Note that the decoder relies on the channel state information (the first term of Eq. (2.8)) when the channel is in good condition, while the \textit{a priori} information (the second term of Eq. (2.8)) is dominant when the channel is bad. By using the modified branch metric, better performance at the decoder is expected for low signal to noise ratios, compared to the case of using the conventional branch metric (only first term). Ruf and Hagenauer [56] demonstrated the performance of the APRI-SOVA decoder for image transmission system consisting of both the an image subband source coder and the channel coder. Improvements up to 1.8 dB in image PSNR or 0.3 dB in $E_s/N_0$ can be achieved. Phamdo and Farvardin [19] exploited the redundancy in a binary Markov source by applying a maximum \textit{a posteriori} (MAP) detector. Miller and Park [21] utilized the redundancy in a sequence of random indexes output by a vector quantizer. Here the new source decoder
named as a sequence-based approximate minimum mean-squared error (SAMMSE) decoder was proposed. The SAMMSE decoding rule can be computed based on a discrete hidden Markov model (HMM). Van Dyck [23] used the residual redundancy in the compressed bit stream from the MPEG-4 embedded zerotree wavelet (EZW) coder. Here the performance improvement of the system comes entirely from the use of source-controlled channel decoding; there is no explicit channel coder. The method of using a discrete HMM based symbol MAP estimation discussed in [21] is applied for both cases; the lowest frequency subband (LFS) and the zerotree symbols in the higher frequency subband (HFS). The goal of the symbol MAP estimation is to find the states that are individually most likely for each time by computing the \textit{a posteriori} probability

\begin{equation}
\gamma_t(i) = P(I_t = i|\mathcal{L}, \lambda), \tag{2.10}
\end{equation}

where is the sequence of transmitted source symbols \( \mathcal{I} = (I_1, I_2, \ldots, I_T) \), and the received sequence \( \mathcal{J} = (J_1, J_2, \ldots, J_T) \). The parameter \( \lambda \) represents a discrete HMM as \( \lambda = (A, B, \pi) \). The first argument \( A \) is an \( N \times N \) matrix, whose elements are the source transition probabilities

\begin{equation}
a_{ij} = P(I_{t+1} = j | I_t = i). \tag{2.11}
\end{equation}

The second matrix \( B \) contains the observation symbol probabilities

\begin{equation}
b_i(j) = P(J_t = j | I_t = i). \tag{2.12}
\end{equation}
The third matrix \( \pi \) is the initial state distribution, with the members given by \( \pi_i = P(I_1 = i_1) \) for all \( N \) states. It is worth noting that the matrices \( A \) and \( \pi \) are actually \textit{a priori} information, and the matrix \( B \) is the channel state information. When there is no \textit{a priori} information, or in another words \( a_{ij} = a_c, \forall i, j \) and \( \pi_i = \pi_c, \forall i \), the value of the \textit{a posteriori} information shown in Eq. (2.10) depends only on the channel transition probabilities \( b_i(j) \). Here the simulation results in term of PSNR for the LFS [23] show significant improvement at bit error rates (BERs) ranging from \( 10^{-2} \) to \( 5 \times 10^{-4} \). For the case of the zerotree symbols in the HFS, only a slight improvement is achieved for the Gilbert-Elliot [44, 45] channel. The use of an explicit channel coder is suggested for further improvement.
Chapter 3

Convolutional Codes

Like block codes, convolutional codes are a type of channel codes. In the 1980s and 1990s, convolutional codes have become popular because of not only their outstanding performance, but also their simple implementation. During the 1970’s [27], the fundamentals of the algebraic structure of convolutional codes was profoundly studied by Forney. This led to better understanding of convolutional codes and useful tools for their analysis. For instance, the invariant factor theorem allows one to determine whether or not a given convolutional code has the noncatastrophic error propagation property. Johannesson and Wan [30] introduced the minimality criteria and a minimality test. Viterbi’s tutorial material [28] emphasized the use of convolutional codes in a communication system; the distance properties and an upper bound on bit error probability for communication systems over an additive white Gaussian noise (AWGN) channel were examined. The convolutional codes mentioned above are actually known as binary convolutional codes, which are the convolutional codes over a Galios finite field $F = [0, 1] (GF(2))$. In 1989, Massey and Mittelholzer [32] proposed convolutional codes over finite rings. This type of linear codes can be naturally connected to $M$-ary phase modulation. Note that this paper is the first reference to ring convolutional codes.

In this chapter, the fundamental structure of a convolutional encoder and the development of the maximum likelihood decoder are described first. Then, the important
properties of convolutional codes, such as being noncatastrophic and having a desirable minimum (free) distance, are explained in detail. The use of these properties is commonly seen in the design of convolutional codes. Finally, the structure and properties of convolutional codes over a ring are discussed.

3.1 Convolutional Encoder

A convolutional encoder \((n, k, m)\) over a finite field \(F\) can be described as a \(k\)-input, \(n\)-output, constant linear causal, finite-state sequential circuit consisting of an \(m\) shift registers [27]. Figure 3.1 illustrates the block diagram of a convolutional encoder. The input \(u_i\) consists of \(k\) symbols which are elements in \(F\). A convolutional code produces the output \(y_i\) consisting of \(n\) symbols. Mathematically, a convolutional code

![Convolutional encoder block diagram](image-url)
can be represented as a row space of a generator matrix whose elements are generally in
the field of binary rational functions in a delay variable $D$. The code rate $R$ is defined as
$k/n$, and the constraint length of the code is $(m+1)k$. In general, the input and output
sequences can be assumed to be semi-infinite, and they can be expressed in terms of a
unit time delay $D$ as

$$
\mathbf{u}(D) = \mathbf{u}_0 + \mathbf{u}_1 D + \mathbf{u}_2 D^2 + \ldots
$$

(3.1)

and

$$
\mathbf{y}(D) = \mathbf{y}_0 + \mathbf{y}_1 D + \mathbf{y}_2 D^2 + \ldots,
$$

(3.2)

respectively, where $\mathbf{u}_i = (u_{0,i}; u_{1,i}; \ldots; u_{k-1,i})$ and $\mathbf{y}_i = (y_{0,i}; y_{1,i}; \ldots; y_{n-1,i})$. The
generator matrix $G(D)$ of the convolutional code is defined as a rank-$k$, $k \times n$ matrix
of polynomial or rational functions in $D$, such that the input and output relationship is
$\mathbf{y}(D) = \mathbf{u}(D)G(D)$. Note that the output sequence is the set of all codewords generated
by $G(D)$ and its corresponding input sequence. To gain more understanding how one
can produce the codes, we consider a simple example of a binary convolutional encoder
with rate $R = 1/2$ and number of delays $m = 2$.

**Example 3.1.** Consider the rate 1/2 binary convolutional encoder with the generator
matrix $G(D) = [D + 1, D^2 + D + 1]$. Figure 3.2 shows the block diagram representation
of this code. The parameter $S_0$ and $S_1$ are denoted as the first shift register and the
second shift register, respectively. The state number is defined as $SN = 2S_1 + S_0$. Note
that the number of states is four. The first output bit $y_0$ and second bit $y_1$ are generated
according to the state number and the input bit $u$. Usually, the initial state is set to
zero. When $SN = 0$ and the input bit $u$ is zero, we obtain $y_0 = 0$ and $y_1 = 0$. Regardless
Fig. 3.2. Convolutional encoder block diagram

Table 3.1. All possible codewords generated by the convolutional encoder block diagram shown in 3.2.
of the time that the codes occur, all possible output bits corresponding to all possible input bits and state numbers are shown in Table 3.1.

Fig. 3.3. State diagram of convolutional encoder in Example 3.1.

Since we have the series of input bits connected as the input sequence, the later output bits depend not only on the current input bit, but also on a current state. Typically, a state diagram is used to represent the transition between each state and its corresponding input and output bit. The state diagram of the convolutional encoder in Example 3.1 is shown in Figure 3.3. The input and output bit relation is shown in the form of $u = y_1 y_0$. To see how the sequence of codewords is generated according to time, we often use a trellis diagram. Figure 3.4 illustrates the trellis diagram produced from the state diagram in Figure 3.3. Note that it is conventional to use the zero state as the initial
state. In Figure 3.4, $T$ denotes the symbol interval. Notice that after the time $t > 2T$, the complete trellis emerges, since the convolutional encoder in this example contains two shift registers. Generally, for the binary case, the complete trellis emerges after time $t > mT$, where $m$ is the length of the shift register. The trellis diagram is a useful tool for analyzing some important characteristics of a convolutional code, such as its minimum distance and its noncatastrophic error propagation property.

### 3.1.1 Distance structure of Convolutional Codes

The distance of a convolutional code is defined as a metric measuring the distance between each pair of codewords in the code. For instance, if we consider the number of bits that differ between two codewords as a distance metric, the distance between binary codeword 00110 and 11000 is four. In fact, this distance is often known as the Hamming distance. In block code, the minimum distance, $d_{\text{min}}$, is defined as the minimum metric between codeword sequence. If a code is linear (The binary sum of two codewords is a
codeword itself), the $d_{\text{min}}$ can be determined regardless of a reference codeword. Note that we often use all-zero codeword. The $d_{\text{min}}$ can be used to approximately determine the error correcting efficiency of a code.

For a convolutional code, the error control performance depends on the decoding method. If a decoder decodes at one output constraint length at a time, treating a code almost as if it were a block code, in that case, $d_{\text{min}}$ is a key parameter. However, if a decoder use a technique called maximum likelihood sequence estimator (MLSE), the minimum distance between codeword sequence, known as $d_{\text{free}}$, is significant. Since the number of all possible codewords of a convolutional code depends on the length of the observation time interval, it may not be trivial to compute distances among them. Fortunately, the minimum distance is no longer dependent upon the observation time interval if the observation length is set to be long enough, probably more than twice of the constraint length [51]. Like the case of a linear block code, if a convolutional code is linear, its free distance is unique, regardless of the choice of a reference codeword sequence. Therefore, the all-zero codeword is commonly used. For the sake of clarity, let us consider the trellis diagram in Figure 3.4. The free (Hamming) distance of this particular code occurs when the observation interval $t \geq 3T$, and it is equal to five. Figure 3.5 shows the path (or codeword) whose distance from the all-zero codeword corresponds to the minimum distance. Note that this path is the shortest path that diverges from and then merges back to the all-zero path. The minimum distance allows us to determine the smallest number of incorrect bits that changes one codeword into another. In this case, only one incorrect bit is received when the decoder follows the shortest incorrect path.
3.1.2 Equivalence of Convolutional Encoders

It is possible to find two encoders that produce the same set of codewords. These encoders are said to be in the same equivalence class of encoders. For two block encoders to be equivalent, they generate the same domain of codewords. For two convolutional encoders, they are said to be equivalent if not only they produce the same set of codewords but also the pair of corresponding states must be the same. This guarantees that the two equivalent convolutional encoders produce the same set of codeword sequences. This equivalent property allows one to select the encoder having the least complexity. To determine whether or not encoders are in the same equivalence class, we apply a theorem introduced in [30]. It states that for two rate $k/n$ convolutional encoding matrices noted as $G(D)$ and $G'(D)$ to be in the same equivalence class, there exists a $b \times b$ nonsingular matrix $T(D)$ such that $G(D) = T(D)G'(D)$. The proof of this theorem can be found in [30].
Example 3.2. Consider the rate $2/3$ convolutional encoder matrix,

$$G(D) = \begin{bmatrix} 1 & D & D^2 \\ D + 1 & 1 + D + D^2 & 1 \end{bmatrix}. \tag{3.3}$$

It is equivalent to the encoding matrix,

$$G'(D) = \begin{bmatrix} 1 & D & D^2 \\ 1 & 1 + D & D^3 + 1 \end{bmatrix} \tag{3.4}$$

because there exists $2 \times 2$ nonsingular matrix,

$$T(D) = \begin{bmatrix} 1 & 0 \\ D & 1 \end{bmatrix} \tag{3.5}$$

satisfying the expression, $G(D) = T(D)G'(D)$.

According to their structures, convolutional encoders can be categorized into two main classes, systematic and nonsystematic encoders. For a given time, when a rate $k/n$ convolutional encoder produces the $n$ output bits, $k$ bits of the $n$ output bits are the same as the $k$ input bits. This encoder is classified as a systematic encoder. Again from Theorem 25 in [30], we conclude that it is possible to find nonsystematic polynomial encoders that are equivalent to systematic rational encoders. For instance, the rate $1/2$ systematic rational encoder $G'(D) = [1, D/D^2 + 1]$ can be transformed into the rate $1/2$ nonsystematic encoder $G(D) = [D^2 + 1, D]$ by using $T(D) = D^2 + 1$. Typically, systematic
convolutional encoders are simpler to implement but unless we use systematic rational encoders, they are generally less powerful when used together with maximum likelihood decoding [28]. Note that the performance of convolutional codes mainly depends on their free distance. Table 3.2, taken from [28], illustrates the maximum of the free distance of convolutional codes generated among systematic polynomial and nonsystematic polynomial encoders for different numbers of delays. It can be shown that between the group of systematic polynomial and nonsystematic polynomial encoders at the same complexity, there exists at least one nonsystematic polynomial encoder corresponding to the convolutional code having largest maximum free distance.

<table>
<thead>
<tr>
<th>Number of delays</th>
<th>Systematic</th>
<th>Nonsystematic</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 3.2. The maximum free distance of systematic encoders and nonsystematic encoders

3.1.3 Catastrophic Error Propagation of Convolutional Codes

A convolutional code experiences an unpleasant event called a catastrophic error [32] if a finite number of codeword errors results in an infinite number of decoded data bit errors. By using the state diagram of a convolutional encoder, it is easily seen that catastrophic errors can occur if and only if any closed loop path in the diagram
corresponds to the all-zero codeword, where some of the data bits corresponding to the path are not zero. To illustrate this statement, we consider the follow example.

**Example 3.3.** Consider rate 1/2 binary convolutional encoder $G(D) = [1 + D^3, 1 + D + D^2]$ whose state diagram is shown in Figure 3.6. Assuming that the data bit sequence is all-zero and the initial state is also zero, we encode these bits resulting in all-zero codewords. By transmitting these codewords through any channel, they are usually corrupted by channel noise, possibly resulting in the different received codewords. In this case, if the received codewords appear to have three bit errors as 00 11 10 00 00 00 ⋯, the corresponding decoded data bit is 0 1 1 0 1 1 ⋯. The error data bits occur infinitely from only three bit errors if an infinite number of data bits is assumed. Note that the only way to stop this error propagation is to make another error.
In order to avoid using a catastrophic encoder, we must perform some kind of noncatastrophic encoder check. The following theorem allows one to examine whether a convolutional encoder over the finite field GF(2) is noncatastrophic.

**Theorem 3.1. (Invariant-Factor Theorem [27])** “Let R be a principal ideal domain and let G be a k × n matrix whose elements belong to R denoted as R-matrix. Then G has an invariant-factor decomposition $G = AB\Gamma$, where A is a square k × k R-matrix with unit determinant, hence with an R-matrix inverse $A^{-1}$; B is a square n × n R-matrix with R-matrix inverse $B^{-1}$; and $\Gamma$ is a k × n diagonal matrix, whose diagonal elements $\gamma_i$, $1 \leq i \leq k$, are called invariant factors of G with respect to R.”

It is stated in [27] that for a given encoder G, the invariant factors are unique and can be calculated as $\gamma_i = \frac{\Delta_i}{\Delta_{i-1}}$, where $\Delta_i$ is the greatest common divisor (GCD) of the $i \times i$ subdeterminants (minors) of G, with $\Delta_0 = 1$. To avoid catastrophic codes, the encoder G must have a feedback-free (polynomial) realizable pseudo-inverse $G^{-1}$. Note that if G and $G^{-1}$ are both feedback-free or polynomial, G is called a basic encoder. According to the invariant-factor theorem, G has pseudo-inverse $G^{-1}$, which is an R-matrix if and only if $\Delta_k = 1$.

**Example 3.4.** Consider a rate 2/3 binary convolutional encoder,

$$G(D) = \begin{bmatrix}
1 + D & D & D^2 + D + 1 \\
D^2 + 1 & D + 1 & D^2 \\
\end{bmatrix}. \quad (3.6)$$

---

1. This definition is not always used [27][32][30]
In this case, we have \( k = 2 \); thus, we check whether or not \( \Delta_2 = 1 \), since it is the GCD of the \( 2 \times 2 \) subdeterminants of \( G(D) \). The computation is shown below.

\[
\begin{align*}
\det \begin{bmatrix} 1 + D & D \\ D^2 + 1 & D + 1 \end{bmatrix} &= D^3 + D^2 + D + 1 \\
\det \begin{bmatrix} 1 + D & D^2 + D + 1 \\ D^2 + 1 & D^2 \end{bmatrix} &= D^4 + D^2 + D + 1 \\
\det \begin{bmatrix} D & D^2 + D + 1 \\ D + 1 & D^2 \end{bmatrix} &= 1
\end{align*}
\]

Since the GCD of \( D^3 + D^2 + D + 1 \), \( D^4 + D^2 + D + 1 \), 1 is one, therefore this encoder is noncatastrophic.

### 3.2 Convolutional Decoder

Convolutional decoder algorithms can be divided into two different approaches: algebraic decoders and probabilistic decoders. Details and further references regarding these decoders can be found in [28]. In this section, we emphasize one type of probabilistic decoder known as the Viterbi decoder. The Viterbi algorithm [52] was introduced in 1967 as a convolutional decoding method. This algorithm is said to be “optimal”, since it is the solution to the problem of maximum \textit{a posteriori} probability (MAP) estimation of the state sequence of a finite-state discrete-time Markov process. Note that the process is said to be a Markov process if the probability of being in state \( x_k \) at time \( k \), given all
states from time 0 to $K - 1$, depends only on the state $x_{k-1}$ at time $k - 1$, i.e.

$$P(x_k|x_0, x_1, \cdots, x_{K-1}) = P(x_k|x_{k-1}).$$ \hspace{1cm} (3.7)

A rate $k/n$ convolutional encoder can be viewed as a finite-state discrete-time Markov process, where its state diagram illustrates all branches connected between all possible previous states $x_{k-1}$ at time $k - 1$ and all possible current states $x_k$ at time $k$. The transition probability $P(x_k|x_{k-1})$ depends on the the transition probability $P(u_k|u_{k-1})$ where $u_k$'s are data bits. The MAP sequence estimation problem is to determine the state sequence for which $P(x|z)$ is maximum, where $z$ is the observed sequence associated with the state sequence $x$. Note that this estimation problem is equivalent to finding the state sequence for which $P(x, z) = P(x|z)P(z) = P(z|x)P(x)$ is maximum. The joint probability of state sequence $x$ can be expressed as

$$P(x) = P(x_0, x_1, \cdots, x_{K-1})$$

$$= P(x_{K-1}|x_{K-2}, \cdots, x_0)P(x_{K-2}|x_{K-3}, \cdots, x_0) \cdots P(x_1|x_0)P(x_0).$$ \hspace{1cm} (3.8)

Due to the property of a Markov process in Eq. (3.7), Eq. (3.8) reduces to

$$P(x) = P(x_{K-1}|x_{K-2})P(x_{K-2}|x_{K-3}) \cdots P(x_1|x_0)P(x_0).$$ \hspace{1cm} (3.9)

By using the Markov process and memoryless noise properties, we obtain [37]

$$\arg \max_x P(z|x)P(x) = \arg \max_x \prod_{k=1}^{K-1} P(z_k|x_k, x_{k-1})P(x_k|x_{k-1}).$$ \hspace{1cm} (3.10)
Since the natural log function, $\ln(.)$ is a monotonic function, we can apply this function
to Eq. (3.10) without changing its solution. The right-hand side of Eq. (3.10) becomes

$$\arg \max_x \sum_{k=1}^{K-1} \ln P(z_k|x_k, x_{k-1}) + \ln P(x_k|x_{k-1}).$$  \hspace{1cm} (3.11)$$

Moreover, if we define each branch metric as

$$\lambda(x_k, x_{k-1}) \overset{\triangle}{=} -\ln P(z_k|x_k, x_{k-1}) - \ln P(x_k|x_{k-1}),$$  \hspace{1cm} (3.12)$$

Eq. (3.11) is simplified to

$$\arg \min_x \sum_{k=1}^{K-1} \lambda(x_k, x_{k-1}).$$  \hspace{1cm} (3.13)$$

From Eq. (3.13), one can see that the Viterbi algorithm is associated with finding the
shortest path in the trellis diagram.

**Example 3.5.** Consider a communication system that consists of a channel encoder
connected with a memoryless modulator at the transmitter, and the branch metric com-
putation unit followed by a Viterbi decoder at the receiver. Assume that we use a rate
1/2 binary convolutional encoder, $G(D) = [D + 1, D^2 + D + 1]$. Figure 3.7 illustrates
the block diagram of the convolutional encoder. The branch metrics computed at each
symbol time interval, $(i - 1)T \leq t \leq iT$, where $i = 1, 2, \cdots, 5$, are shown in Table
3.5. The Viterbi algorithm begins by initializing the state cost $\Gamma(x_0)$. Usually, we
let $\Gamma(0) = 0$ and $\Gamma(x_0) = \infty$ (a large number in practice), where $x_0 \neq 0$. In order to
clearly illustrate the recursive process of Viterbi algorithm, we place all branch metrics
Fig. 3.7. The block diagram of the convolutional encoder $G(D) = [D + 1, D^2 + D + 1]$.

Table 3.3. The branch metrics computed at each time interval $(k - 1)T \leq t \leq kT$.
at symbol interval \( i = 1 \), corresponding to the second column in Table 3.5, into the trellis diagram as shown in Figure 3.8. Next, the computation of variable \( \Gamma(x_k, x_{k-1}) \) is performed at each transition by adding the branch metrics to their previous state costs. The state costs \( \Gamma(x_1) \) are then updated to be the minimum of all \( \Gamma(x_1, x_{k-1}) \) which is the shortest path segment (survivor path) coming into the state \( x_1 \). From Figure 3.8, we have two survivor paths after the first iteration. The process is then repeated for the second iteration as shown in Figure 3.9. Finally, the survivor paths after the fifth iteration are shown in Figure 3.10. If the decision is made at this time, the decision path is chosen as the survivor path associated with the minimum state cost. In this case, we obtain the minimum state cost at zero state (\( \Gamma(0) = 6 \)). Figure 3.10 shows the decision path which corresponds to the decoded bits 1,1,0,0,0.
Fig. 3.9. The trellis diagram labeled with branch metrics at $k = 2$ and the survivor paths after the second iteration.

Fig. 3.10. The survivor paths after the fifth iteration and the decision path (solid line).
### 3.3 Design of Convolutional Codes

In this section, we continue the discussion of the search for the best convolutional codes. To determine the goodness of convolutional codes, we must first define the criterion. The best criterion would be to select the code that allow us to achieve the smallest probability of error over the entire range of channel noise power. However, this is not an easy task. As previously known, the performance of a convolutional code at low noise power depends not only on its decoding algorithm, but also its free distance. Therefore, one possible criterion is to compare the codes on the basis of their free distances. The free (Hamming) distance, $d_{\text{free}}$, can be computed by using the trellis diagram of the codes as previously mentioned in section 1.1.1. The best code is the code that has the maximum of the free distance. Table 3.3 summarizes the best rate $\frac{1}{2}$ binary convolutional encoders found by Odenwalder [51]. Note that when the complexity (number of delays) of a convolutional encoder increases, it is possible that the performance of convolutional codes, expressed in term of $d_{\text{free}}$, is also increased. Since a convolutional

<table>
<thead>
<tr>
<th>Number of delays</th>
<th>Best encoders</th>
<th>$d_{\text{free}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>$[D^2 + D + 1, D^2 + 1]$</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>$[D^3 + D^2 + D + 1, D^2 + D + 1]$</td>
<td>6</td>
</tr>
<tr>
<td>4</td>
<td>$[D^4 + D^2 + D + 1, D^4 + D^3 + 1]$</td>
<td>7</td>
</tr>
<tr>
<td>5</td>
<td>$[D^5 + D^3 + D^2 + D + 1, D^5 + D^4 + D^2 + 1]$</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>$[D^6 + D^3 + D^2 + D + 1, D^6 + D^5 + D^3 + D^2 + 1]$</td>
<td>10</td>
</tr>
<tr>
<td>7</td>
<td>$[D^7 + D^4 + D^3 + D^2 + D + 1, D^7 + D^6 + D^5 + D^2 + 1]$</td>
<td>10</td>
</tr>
<tr>
<td>8</td>
<td>$[D^8 + D^7 + D^5 + D^3 + D^2 + D + 1, D^8 + D^4 + D^3 + D^2 + 1]$</td>
<td>12</td>
</tr>
</tbody>
</table>

Table 3.4. The best rate 1/2 binary convolutional encoders and their free (Hamming) distance.
code is employed as a channel coding method followed by some modulation technique, the process of designing convolutional codes sometimes includes the signal waveforms generated by a modulator. The advantage comes from the fact that it allows one to achieve significant coding gains without compromising bandwidth efficiency. The more the signal waveforms directly representing codeword sequences are different, the more resilient the system is to a channel noise. Mathematically speaking, each pair of signal sequences should have a large Euclidean distance between them. This important concept is known as trellis-coded modulation (TCM) [43]. Instead of using the Hamming distance, convolutional codes are jointly designed with signal mapping functions in order to maximize the minimum squared Euclidean distance between coded signal sequences. This topic will be discussed further in the next chapter.

3.4 Ring Convolutional Codes

Recently, many researchers ([32],[11],[12],[15],[16]) were interested in using linear codes over the ring of integers modulo $M$, $Z_M = \{0, 1, \cdots, M - 1\}$, since this approach provides a natural way to combine coding with $M$-ary phase modulation. A rate $k/n$ convolutional code over $Z_M$ is constructed the same way as in the case of convolutional codes over a binary field, except the coefficients are members of $Z_M$.

**Example 3.6.** Consider the rate 1/2 convolutional code over $Z_4$ whose encoding matrix is shown below.

$$G(D) = [D, 1 + 3D].$$ \hspace{1cm} (3.14)

Figure 3.11 (a) shows the block diagram representation of this code. In this case, a
Fig. 3.11. The block diagram of the encoder $G(D) = [D, 1 + 3D]$. 
multiplier must be placed after the delay unit. The state diagram representation is illustrated in Figure 3.11 (b). Similar to the binary case, the input-output relationship is shown in the form of $u/y_0y_1$. Note that since the output sequence $Y$ is nonbinary, it can be directly mapped to the signal sequence without using a mapper. In this case, the total number of states equals four since we use only one delay or memory unit. In general, the total number of states is computed as $M^m$, where $M$ is the number of elements in $Z_M$ and $m$ is the number of delays. However, the total number of states is sometimes reduced. For instance, if the encoder matrix, $G(D) = [1, 1 + 2D]$ is used and the multiplier is placed before the delay unit, the total number of states is reduced to two states.

Similar to convolutional codes over a finite field, a ring convolutional code may be catastrophic. The following theorem provides a way to check for a noncatastrophic encoder.

**Theorem 3.2.** [32] “A polynomial encoder $G(D)$ over the ring $Z_M$, where $M = p^m$ and $p$ is a prime, is catastrophic if and only if when the coefficients of the polynomials in $G(D)$ are each reduced modulo $p$, the resulting polynomial encoder over the finite field $GF(p)$ is catastrophic.”

**Example 3.7.** A polynomial encoder $G(D) = [D + 1, 3D + 1]$ over the ring $Z_4$ is catastrophic, since all coefficients of the elements of $G(D)$, reduced modulo 2, result in the polynomial encoder $[D + 1, D + 1]$ over the finite field $GF(2)$. From the invariant-factor theorem, it is easy to see that the encoder $[D + 1, D + 1]$ is catastrophic; therefore, so is $G(D)$. 
Chapter 4

Continuous Phase Modulation (CPM) and
Trellis-Coded Modulation (TCM)

4.1 The General Description of CPM

Continuous phase modulation (CPM) is classified as a type of nonlinear digital modulation method with memory. Since a phase of a CPM signal is constrained to be continuous, it allows dependence between the signals transmitted in successive symbol intervals. CPM has become more attractive for band-limited channel in recent years, due to the increasing demand for bandwidth efficient constant amplitude transmitted signals. This feature provides some noise immunity for a communication system over nonlinear channels. A good example of a CPM application can be found in a satellite communication system where

- in the downlink transmission, a highly nonlinear power amplifier is used, thus generating a large distortion;

- frequency bands of UHF (300-3000 MHz) and VHF (30-300 MHz), which are usually impaired by fading, are employed.

These problems can be alleviated by employing a modulation method that produces constant envelope signals.
4.1.1 General Definition

In general, a generic CPM system transmits signals in the form [24]

\[ sm(t, \omega) = \sqrt{\frac{2E}{T}} \cos(2\pi f_0 t + \varphi(t, \omega) + \varphi_0), \]  

(4.1)

where \( \omega \) is a sequence of uncorrelated \( M \)-ary data symbols with \( \alpha_i \in \{ \pm 1, \pm 3, ..., \pm (M - 1) \} \). The information-carrying phase \( \varphi(t, \omega) \) is defined [24] as

\[ \varphi(t, \omega) = 2\pi h \sum_{i=-\infty}^{\infty} \alpha_i f(t - iT), \quad -\infty < t < \infty, \]  

(4.2)

where \( h \) is the modulation index, and \( f(t) \) is the baseband phase response defined in terms of the frequency pulse \( g(t) \) as \( f(t) = \int_{-\infty}^{t} g(\tau)d\tau, \quad -\infty < t < \infty \). In a causal CPM scheme, \( g(t) \) must satisfy \( g(t) = 0, \ t < 0, \ t > LT \) and \( g(t) \neq 0, \ 0 \leq t \leq LT \). Thus the information-carrying phase within any symbol interval can be written [14] as

\[ \varphi(t, \omega) = \pi h \sum_{i=-\infty}^{n-L} \alpha_i + 2\pi h \sum_{i=n-L+1}^{n} \alpha_i f(t - iT), \quad nT \leq t \leq (n + 1)T. \]  

(4.3)

The CPM scheme is completely defined by \( h, M \) and \( g(t) \) (or \( f(t) \)). We often use \( g(t) \) in the form of a rectangular (L-REC)

\[ g(t) = \begin{cases} \frac{1}{2LT}, & 0 \leq t \leq LT, \\ 0, & \text{otherwise}, \end{cases} \]  

(4.4)
or raised cosine function \((L-RC)\)

\[
g(t) = \begin{cases} 
\frac{1}{2LT}[1 - \cos\left(\frac{2\pi t}{LT}\right)], & 0 \leq t \leq LT, \\
0, & \text{otherwise.}
\end{cases} 
\tag{4.5}
\]

If \(L = 1\), the modulation is called \textit{full response} CPM [24]. Otherwise \((L > 1)\), it is known as \textit{partial response} CPM [25]. Continuous phase frequency shift keying (CPFSK) is a special case of full response CPM with a linear baseband phase response (or \(g(t) = 1-\text{REC}\)).

\textbf{Example 4.1.} Consider minimum shift keying (MSK) which is a kind of binary CPFSK with \(h = 1/2\) and \(g(t)\) is 1-REC. We obtain \(g(t)\) from Eq. (4.4) and so the phase response \(f(t)\) becomes

\[
f(t) = \int_{-\infty}^{t} g(\tau) d\tau = \begin{cases} 
0, & t < 0 \\
\frac{t}{2T}, & 0 \leq t \leq T \\
\frac{1}{2}, & t > T.
\end{cases} 
\tag{4.6}
\]

\subsection*{4.1.2 The Representation of CPM}

In general, a CPM system can be represented by using a phase diagram. The phase diagram that corresponds to all possible phases is called a \textit{phase tree}. Note that in general, a phase tree may or may not be linear. Furthermore, when the phase tree is reduced modulo \(2\pi\), it is called a \textit{phase trellis}. Figure 4.1 illustrates the phase tree and phase trellis of the MSK system described in Example 4.1. Observe that both the MSK phase tree and phase trellis are time variant since they are not the same at each channel use. This time-variant property is actually undesirable since the demodulation process
Fig. 4.1. (a) the MSK phase tree and (b) the MSK phase trellis

requires a more complicated receiver. In the next section, we shall show that it is possible to construct a time invariant phase trellis. Another way of representing a CPM system is to use a state diagram. This compact representation contains the transitions at the time instants \( t = nT \) between all phase states according to all possible input symbols \( \alpha_i \) without explicitly illustrating the time variable. The state diagram of MSK in Example 4.1 is shown in Figure 4.2. Note that four states are required in this case even though we have only two input symbols. Two out of four states are reached for each time \( t = nT \).

### 4.1.3 Spectral Characteristics of CPM

As described in [1], the bandwidth used for CPM signals generally depends on the choice of the modulation index \( h \), the frequency pulse \( g(t) \), and the number of signals \( M \). A large value of \( h \) results in the large bandwidth occupation. Due to the smoothness of the raised cosine frequency pulse \( (g(t) = L-RC) \), its bandwidth is less than that of the rectangular frequency pulse \( (g(t) = L-REC) \). For the same reason, the bandwidth
Fig. 4.2. The MSK state diagram.

Occupation of partial response CPM \((L > 1)\) is smaller than that of full response CPM \((L = 1)\) due to a smoother baseband phase response. For CPFSK, the power density spectrum can be expressed \([1]\) as

\[
\Phi_{\nu\nu}(f) = T \left[ \frac{1}{M} \sum_{n=1}^{M} A_n^2(f) + \frac{2}{M^2} \sum_{n=1}^{M} \sum_{m=1}^{M} B_{nm}(f) A_n(f) A_m(f) \right],
\]  

\[(4.7)\]

where

\[
A_n(f) = \frac{\sin \pi [fT - \frac{1}{2} (2n - 1 - M)h]}{\pi [fT - \frac{1}{2} (2n - 1 - M)h]}
\]

\[
B_{nm}(f) = \frac{\cos (2\pi fT - \alpha_{nm})}{1 + \psi^2 - 2\psi \cos 2\pi fT}
\]

\[
\alpha_{nm} = \pi h (m + n - 1 - M)
\]

\[
\psi = \frac{\sin M\pi h}{M \sin \pi h}
\]
In the special case of binary CPFSK with $h=1/2$, the power spectrum of MSK can be reduced to

$$
\Phi_{\mu\nu}(f) = \frac{16A^2T}{\pi^2} \left( \frac{\cos 2\pi fT}{1 - 16f^2T^2} \right)^2.
$$

(4.8)

Figure 4.3 shows the power spectral density of MSK and 4-ary CPFSK with $h=1/4$. Note that both spectral densities occupy almost the same amount of bandwidth.

4.2 The Alternative Representation of CPM

In 1988, Rimoldi [14] proposed a technique to decompose a CPM system into a linear continuous-phase encoder (CPE) and a time invariant memoryless modulator (MM). This technique leads to an alternative realization of CPM system which has less complexity. The decomposition is done by modifying an information-carrying phase
trellis to be a time invariant phase (tilted phase) trellis. In this case, the set of all possible phase trajectories over any one symbol period is identical. The tilted phase trellis is desirable because it yields a simpler decoder. The CPE part provides the memory to a modulating system, and it can be characterized as a convolutional encoder over the ring of integers modulo $P$ if the modulation index $h=K/P$ is considered, where $K$ and $P$ are relatively prime. The tilted phase is defined as follows,

$$\psi(t, \alpha) = \varphi(t, \alpha) + \pi h (M - 1) \frac{t}{T}, \quad (4.9)$$

where $\varphi(t, \alpha)$ is the traditional phase as previously defined in Eq.(4.3). It is convenient to rewrite the data sequence $\alpha$ as

$$U_i = \alpha_i + \frac{(M - 1)}{2}, \quad (4.10)$$

where $U_i \in 0, 1, \cdots, M - 1$. By substituting $t = \tau + nT$, $0 \leq \tau \leq T$ into Eq.(4.9), we obtain the physical tilted phase expression ($\tilde{\psi}(t, U) = R_{2\pi}[\psi(t, U)]$) in terms of $\tau$ and $U_i$ as

$$\tilde{\psi}(\tau + nT, U) = R_{2\pi} \left[ 2\pi h R_p \left( \sum_{i=0}^{n-L} U_i \right) + 4\pi h \sum_{i=0}^{L-1} U_{n-i} f(\tau + iT) + W(\tau) \right], \quad (4.11)$$

where $[W(\tau) = \frac{\pi h (M - 1) \tau}{T} - 2\pi h (M - 1) \sum_{i=0}^{L-1} f(\tau + iT) + (L - 1)(M - 1) \pi h]$ and $R_x[.]$ is the modulo $x$. Note that since in Eq. (4.11), the physical tilted phase depends only on the parameter $\tau$, it is time invariant.
Example 4.2. Consider an MSK system with \( \alpha \in (-1, 1) \). By using Eqs. (4.10) and (4.9), we obtain \( U_i \in (0, 1) \) and the tilted phase of MSK shown in Figure (4.4). Note that in this case, the state diagram contains only two physical phase states, which are zero and \( \pi \).

![Diagram](image)

**Fig. 4.4.** The MSK tilted phase tree (a) and the MSK tilted phase trellis (b).

The MM input sequence is denoted by \( X_n = [U_n, U_{n-1}, \ldots, U_{n-L+1}, V_n] \), where \( V_n = R_P \left[ \sum_{i=0}^{n-L} U_i \right] \). The task of the CPE is to update the MM input \( X_n \) in such a way that \( V_{n+1} = R_P[V_n + U_{n+L-1}] \). Figure 4.5 shows the block diagram of the CPE and MM. The realization of the MM system is decomposed into in-phase and quadrature-phase components. The output signal can be written as

\[
sm(\tau, X_n) = I(\tau, X_n)\Phi_I(\tau) + Q(\tau, X_n)\Phi_Q(\tau),
\]

(4.12)
Fig. 4.5. The block diagram of a CPE and MM system.

where

\[ I(\tau, X_n) = \sqrt{\frac{E}{T}} \cos \tilde{\psi}(\tau, X_n), \]

\[ Q(\tau, X_n) = \sqrt{\frac{E}{T}} \sin \tilde{\psi}(\tau, X_n), \]

\( \frac{E}{T} \) is the signal energy per symbol interval, \( \tilde{\psi} \) is the physical tilted phase defined as \( \tilde{\psi} = R_{2\pi}[\psi] \), and

\[ \Phi_I(\tau) = \sqrt{\frac{T}{2}} \cos[2\pi f_1(\tau + nT) + \varphi_0], \]

\[ \Phi_Q(\tau) = -\sqrt{\frac{T}{2}} \sin[2\pi f_1(\tau + nT) + \varphi_0], \]
with the new carrier frequency \( f_1 = f_0 - h(M - 1)/2T \). This adjustment is required to compensate for the offset between the tilted and the conventional phase. It is worth mentioning that the transfer function \( C(D) \) of the CPE is related to the input \( U(D) \) and output \( X(D) \) as \( X(D) = U(D)C(D) \), where [14]

\[
C(D) = \begin{bmatrix}
1 & D & \cdots & D^{L-1} & \frac{D^L}{1-D}
\end{bmatrix}
\]

\[
= \frac{1}{1-D} \begin{bmatrix}
1 - D & D - D^2 & \cdots & D^{L-1} - D^L & D^L
\end{bmatrix}
\]

\[
= T(D)C'(D).
\]

\( C'(D) \) is equivalent to \( C(D) \), and it provides the feedback free representation of the CPE.

4.3 Trellis-Coded Modulation for CPM

Trellis-coded modulation (TCM) is a combined coding and modulation technique for digital transmission. Due to significant coding gain over conventional uncoded multilevel modulation without compromising bandwidth efficiency, TCM is popularly used in band-limited channels. The task of this scheme is to generate coded signal sequences by using the combination of a finite-state encoder and the selection of a modulation signal. The crucial concept for designing a TCM system is that the coding and modulation process are designed jointly by maximizing the minimum (squared) Euclidean distance, known as the "free distance", between coded signal sequences. This results in the construction of coded signals whose free distance is larger than the minimum distance between uncoded signals at the same information rate, and approximately
the same bandwidth. The general block diagram of a TCM system is shown in Figure 4.6 [43]. The information sequence is denoted as $I = (I_0, I_1, \ldots, I_n, \ldots)$ where $I_n = (I_0^n, I_1^n, \ldots, I_m^n)$. The convolutional encoder with rate $\frac{m}{m+1}$ generates the coded sequence $J = (J_0, J_1, \ldots, J_n, \ldots)$ where $J_n = (J_0^n, J_1^n, \ldots, J_m^n)$. This coded sequence is then mapped to a signal sequence $s_m = (s_{m0}, s_{m1}, \ldots, s_{mn}, \ldots)$ according to the diagram in Figure 4.6. In 1982 the original work on a TCM scheme was presented by Ungerboeck [43]. In his work the joint design between the binary systematic convolutional encoder (CE) and the memoryless modulations (MM), such as $M$-ary phase shift keying (MPSK) and quadrature amplitude modulation (QAM), were primarily studied. A TCM scheme for modulation with memory, specifically CPM, was studied in [8], [9], and [10]. Lindell, Sundberg, and Aulin [8] consider a TCM system that consists of a rate 1/2 binary convolutional encoder and $M$-ary CPFSK modulation. For a given constraint length $\nu$ and a modulation index $h$, the optimal combinations between encoders and modulators were obtained by computing the minimum squared Euclidean distance.

**Fig. 4.6.** The block diagram of a TCM system.
(SED) for the cases $M = 2$ and $M = 4$. Note that for $M = 4$, the binary $M$-ary mapper is required to map two binary bits to one of four possible signals. Pizzi and Wilson’s work [9] emphasized rate 1/2 binary convolutional codes and both 4-ary full and partial response CPM, CPFSK and $L$-RC ($L > 1$). The mapper is again necessary in this scheme. In fact, two kinds of mapping methods were considered. With a modulation index ranging from $h=1/2$ to $h=1/16$, the performance of optimal encoders for the case of $\nu = 2, 3,$ and 4, is compared by computing their minimum SEDs. Rimoldi [10] treated the systematic convolutional encoder (CE) and the memory part of CPM (CPE) as the overall encoder. This results in a possible reduction in the total number of states, thus less decoder complexity. For a given constraint length the optimal rate 1/2 convolutional encoder for 4-ary CPFSK with $h=1/4$ was found to have larger SED compared to the previous schemes. The computation of SED was done by using the formula in [38].

In 1989 Massey and Mittelholzer [32] suggested that a convolutional encoder over the ring $\mathbb{Z}_M$ of integers modulo $M$, is a natural way to apply $M$-ary phase modulation without using a mapper. Motivated by this work, Baldini and Farrell [11] presented a trellis-coded modulation system by considering both systematic and nonsystematic ring convolutional codes. The performance of this system was compared with an uncoded $M$-ary phase modulation system. Recently, many researchers have been interested in the design of trellis-coding using ring convolutional codes and $M$-ary CPM system (e.g., [12], [15], and [16]). Ugrelidze and Shavgulidz proposed a design of trellis-coded modulation based on rate 1/2 convolutional codes over the ring of integer modulo $M$ and $M$-ary CPFSK. The TCM systems are categorized into three groups based on both the number of states in convolutional codes and in CPFSK. Note that the total number of states
used in the TCM system determines decoding complexity. Yang and Taylor [15] also considered a TCM system using a systematic ring convolutional encoder and a CPFSK system. Here, the decomposition approach to CPFSK system found in [14] was applied. The combination of a systematic ring convolutional code (CE) and a continuous phase encoder (CPE) can be viewed as an overall encoder of this coding scheme. The overall encoder can lead to a reduction in the overall number of states. For a rate 1/2 modulo-4 CE connected with h=1/4 quaternary CPFSK, the transfer function of the overall encoder (CE+CPE) can be computed as

\[ G_{all}(D) = G(D) \cdot \begin{bmatrix} 1 & 0 & 3 & 1 \\ 3D & D & 1 & 0 \end{bmatrix}. \] (4.13)

Also, for a rate 2/3 modulo-8 CE connected with h=1/8 octal CPFSK, the transfer function of the overall encoder can be computed as

\[ G_{all}(D) = G(D) \cdot \begin{bmatrix} 1 & 0 & 7 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 7 & 1 \\ 7D & D & 0 & 0 & 1 & 0 \end{bmatrix}. \] (4.14)

Both matrices in Eq. (4.13) and Eq. (4.14) represents the CPE part of CPFSK. The details of the derivations of these matrices can be found in Section 5.3.2 and also in [15]. It is worth mentioning that by using a precoder or scrambler in front of the conventional (systematic) CPE [14], one is able to remove feedback in the CPE, obtaining a feedback-free (nonsystematic) CPE. Figure 4.7 illustrates the block diagram of the \( M \)-ary full
response CPM with $h=K/M$ in both forms. Note that in this case, the feedback-free form of CPE is used. Given the same complexity of the MM, both the CPM scheme

\[
\text{Feedback-free CPE: } f(t, u_k) = D(X_1, k) + M \mod M
\]

\[
\text{Conventional CPE: } f(t, u_k) = X_1, k
\]

Fig. 4.7. Decomposition model of $M$-ary full response CPM with $h=K/M$. Feedback-free CPE (left) and conventional CPE (right).

with feedback and feedback-free form of CPE generate the same set of transmitted signals and utilize the same amount of bandwidth. However, a feedback-free CPE scheme is much simpler than that of a feedback CPE scheme. Although both systems produce the same probability of Viterbi decoding error events [37] (i.e. the same error symbol rate at high SNR), different bit error probabilities at low SNR may occur due to the different mapping between bit inputs and output signals. In the 4-ary CPFSK case with modulation index $h = 1/4$ over a memoryless additive white Gaussian noise channel, we discover by simulation that at $10^{-1}$ to $10^{-2}$ symbol error rate, the performance in terms of SNR of CPM with a feedback-free CPE is better than that of conventional CPM by about 1.5 to 0.9 dB, when coherent detection is assumed at the receiver.

Another advantage of using the feedback-free CPE occurs when it is combined with a noncatastrophic convolutional encoder (CE); the overall encoder is guaranteed to
be noncatastrophic and when it is combined with a catastrophic CE, the overall encoder will be catastrophic [15]. On the other hand, the combination of a noncatastrophic CE and a conventional CPE may result in a catastrophic overall encoder. For the case of a feedback-free CPE, in order to determine whether the overall encoder is noncatastrophic, we can simply check if the CE is noncatastrophic. Moreover, we will see later on that by using the feedback-free CPE, the source transition probability matrix can be combined into the CPM trellis branch metrics. The derivation of branch metrics in the next chapter illustrates this statement.

In order to search for the optimal trellis-coded CPFSK modulation system, we must compute the minimum SED of the outputs of the memoryless modulator (MM). The closed form formula of SED within one symbol interval is found as in [15]:

\[
D^2(X, X') = \begin{cases} 
2E_s \left[ 1 - \frac{\sin 2\pi h(X_{1,n} + X'_{2,n} - X'_{1,n}) - \sin 2\pi h(X_{2,n} - X'_{2,n})}{2\pi h(X_{1,n} - X'_{1,n})} \right], & X_{1,n} \neq X'_{1,n} \\
2E_s [1 - \cos 2\pi h(X_{2,n} - X'_{2,n})], & X_{1,n} = X'_{1,n},
\end{cases}
\]

(4.15)

where \(X = [X_{1,n}, X_{2,n}]\) are the input of the MM (see Figure 4.7). The optimal rate 1/2 and 2/3 systematic ring convolutional encoders were obtained in this case. For partial response (\(L > 1\)) CPM, specially \(L\)-REC and \(L\)-RC system, the optimal encoders can be found in [16].
Chapter 5

MPEG-4 Imagery Transmission System and Source-Controlled Channel Decoding

5.1 System Description

In this chapter we are interested in developing source-controlled channel decoding using a ring convolutional encoder and CPFSK modulation for an MPEG-4 imagery transmission system. In particular, the correlation of the four possible zerotree symbols found in the higher frequency subbands is used in the decoding process to combat channel noise. The overall system block diagram is illustrated in Figure 5.1.

Fig. 5.1. System block diagram for the image transmission system using a ring convolutional encoder and continuous phase modulation scheme.
The MPEG-4 source encoder produces the LFS bit stream and the HFS packet stream. Details about the MPEG-4 EZW algorithm and the HFS packet structure can be found in Chapter 2. The bits generated by compressing the LFS are placed into a fixed-length packet, while the bits from the HFS are placed into a number of variable-length packets. Since we are interested in using the source transition matrix of the four zerotree symbols at the MAP receiver, the bit stream is converted to be a 4-ary symbol sequence using a natural mapper \((00 \rightarrow 0, \ 01 \rightarrow 1, \ 10 \rightarrow 2, \text{and} \ 11 \rightarrow 3)\). Note that for the HFS when the packet length in bits is an odd number, an extra zero bit is used with the last bit for mapping to a 4-ary symbol. Then, the sequence is passed through the channel encoder and modulator system, which are a rate 1/2 convolutional encoder (CE) over the ring \(Z_4\) and CPFSK with \(h=1/4\), respectively.

In this work, the CPFSK system is decomposed into a continuous phase encoder (CPE) and a memoryless modulator (MM). The overall encoder is the combination of the CE and the CPE. Note that the overall encoder (CE+CPE) is reset for each packet. The memoryless modulator (MM) maps a code symbol into the in-phase and quadrature-phase components of a baseband CPFSK signal. By using baseband simulation, we do not consider a carrier signal. Therefore, these two components of a signal are then corrupted with channel noise and distortion. Even though we are primarily interested in the performance of the system over a fading channel, a Gaussian channel (AWGN) is first considered. The descriptions of both channel models are in the next section.

At the receiver, we employ the maximum \textit{a posteriori} (MAP) receiver consisting of a detection process, a branch metric calculator, and a Viterbi decoder. Note that the MAP receiver is used only when a 4-ary zerotree symbol sequence in each HFS packet
is transmitted. For comparison purposes, a maximum likelihood (ML) receiver (without using the source information) can be used instead of the MAP receiver. In both cases, the rest of the 4-ary symbols are ML decoded. This includes all 4-ary symbols in the LFS and the header in each HFS packet. Please refer to Section 2.1.2.3 for details on the packet structure. The decoded 4-ary sequence is mapped back to a bit stream. The MPEG-4 source decoder decodes this bit stream resulting in a reconstructed image.

We consider the performance of the system, firstly, in terms of the symbol error rate (SER), and secondly, in terms of peak signal to noise ratio (PSNR) and subjective quality.

5.2 Channel Models

5.2.1 Additive White Gaussian Noise (AWGN) Channel

In general, thermal noise can be described by a Gaussian random process. Besides thermal noise, many other types of noise sources are Gaussian (by using the central limit theorem) and have spectral densities that are flat over a wide range of frequencies. A noise signal having a flat power spectral density (PSD) over a wide range of frequencies is called white noise by analogy to white light. The power spectral density of white noise is denoted [3] by

\[ S_n(\omega) = \frac{N_0}{2} \text{ watts/Hz}. \] (5.1)

The factor 2 indicates that \( S_n(\omega) \) is a two-sided PSD. The autocorrelation of white noise is given [3] by

\[ R_{nn}(\tau) = F^{-1}(S_n(\omega)) = \frac{N_0}{2} \delta(\tau), \] (5.2)
where the operator $F^{-1} (\cdot)$ is the inverse Fourier transform. From Eq. (5.2), one can observe that any two different samples of a zero mean Gaussian white noise are uncorrelated and thus independent. In a digital communication system, digital information are transmitted by using $M$ signal waveforms denoted by $s^m(t), m = 1, 2, \ldots, M$. Each signal is transmitted within the symbol interval $T$ and corrupted by the addition of white Gaussian noise. Thus, the received signal in the interval $0 \leq t \leq T$ can be mathematically expressed [1] as

$$r(t) = s^m(t) + n_w(t), \quad 0 \leq t \leq T,$$

(5.3)

where $n_w(t)$ represents a sample function of the additive white Gaussian noise process with PSD given in Eq. (5.1). The representation of a bandpass signal $s^m(t)$ is [1]

$$s^m(t) = \text{Re} \left[ s^m_l(t) e^{j\omega_c t} \right],$$

(5.4)

where the operator $\text{Re}[x]$ is the real part of $x$ and $s^m_l(t)$ is usually known as the complex envelope (or equivalent lowpass) of the real signal $s^m(t)$. Thus, the equivalent lowpass received signal $r_l(t)$ can be expressed [1] as

$$r_l(t) = s^m_l(t) + n_{wl}(t), \quad 0 \leq t \leq T,$$

(5.5)

where $n_{wl}$ is called the complex envelope of $n_w(t)$. 
5.2.2 Fading Channel

In this section, we consider a simple type of fading channel known as the frequency-
nonselective slowly fading channel. A frequency-nonselective channel means that all of
the frequency components in the transmitted signal have the same attenuation and phase
shift. The term slow fading implies that the change in amplitude of the transmitted sig-
nal may be assumed to be a constant during at least one signaling interval. By using a
Rician frequency-nonselective slow fading model, we can write the received signal as [26]

\[
r(t) = (\sqrt{2S} + \sqrt{x_i^2(t) + x_q^2(t)}) \cos(\omega_c t + \theta(t)) + n_w(t), \quad (5.6)
\]

where \( x_i(t) \) and \( x_q(t) \) are the in-phase and quadrature phase components of the scattered
multipath signal, which are Gaussian random variables with zero mean and variance
equal to \( \sigma^2 \). Please refer to [1] for the mathematical representation of \( x_i(t) \) and \( x_q(t) \).

\( S \) is the signal power, \( S = \frac{E_s}{T} \). \( n_w(t) \) is white Gaussian noise with two-sided spectral
density \( N_0/2 \); \( \omega_c \) is the center IF filter radian frequency, and \( \theta(t) \) is the data phase.

Note that if the transmitted baseband signal is \( s_l^m(t) \), the equivalent lowpass
signal can be written as

\[
r_l(t) = (1 + \beta e^{-j\phi})s_l^m(t) + n_{wl}(t), \quad 0 \leq t \leq T, \quad (5.7)
\]

where \( n_{wl}(t) \) is the complex-valued white Gaussian noise random process. Since the
channel is sufficiently slow, the phase shift \( \phi \) can be estimated from the received signal.
Without loss of generality, we assume \( \phi \) to be zero. The parameter \( \beta \) is defined as

\[
\beta = \sqrt{\left[ \frac{x_i(t)}{v} \right]^2 + \left[ \frac{x_q(t)}{v} \right]^2}.
\]  
(5.8)

Let \( \gamma_b \) be \((1 + \beta)^2 E_s/N_0 \). The probability density function of \( \gamma_b \) can be found in [26] as

\[
p(\gamma_b) = \frac{1 + \gamma}{\gamma_b} \exp(-\gamma - (1 + \gamma)\gamma_b/\gamma_b)I_0 \left(2\sqrt{\frac{\gamma(1 + \gamma)\gamma_b}{\gamma_b}}\right),
\]  
(5.9)

where

\[
\gamma_b = E[(1 + \beta)^2 E_s/N_0] = \frac{1 + \gamma}{\gamma} E_s/N_0,
\]  
(5.10)

and \( \gamma = \frac{S}{\sigma^2} \) is the ratio of the direct path power to the scattered path power. If the direct path is blocked by any structures, the fading channel is called a *Rayleigh* channel. This channel can be simulated by letting the signal power \( S \) be small compared to the scattered path power \( \sigma^2 \).

### 5.3 MAP Decoding for the AWGN Channel

In this section, MAP decoding is developed for the additive white Gaussian noise channel. For simplicity, we classify the derivation of MAP decoding into two main parts. Firstly, we compute the branch metric equation when only the CPFSK system is used. Secondly, we consider the case when the CPFSK system is combined with an external convolutional encoder.
5.3.1 MAP Decoding Derivation for the CPFSK System

Forney [37] discussed the Viterbi algorithm as the solution to the problem of MAP estimation of the state sequence of a finite-state discrete-time Markov process. Since the transmission of information bits using CPFSK over an AWGN channel can be cast into this problem, we review the analysis of this algorithm. The process is said to be Markov if the probability of being in state \( x_{k+1} \) at time \( k+1 \) given all states from time 0 to \( k \), depends only on the state \( x_k \) at time \( k \) or

\[
P(x_{k+1}|x_0, x_1, \cdots, x_k) = P(x_{k+1}|x_k).
\]  

(5.11)

For convenience, the feedback-free CPE and MM system illustrated in Figure 4.7 is repeated here in Figure 5.2.

Fig. 5.2. Decomposition model of \( M \)-ary full response CPM with \( h=K/M \).
The time invariant phase state at time $kT$ is computed as $2\pi h x_{2,k}$ and the physical tilted phase $\Psi_k(t)$ is defined in [15] as,

$$\Psi_k(t) = R_{2\pi} \left[ 2\pi h(x_{2,k} + x_{1,k}(t - kT)) \right], \quad kT \leq t \leq (k + 1)T.$$ 

Since $x_{1,k} = R_M(u_k - x_{2,k})$ and $u_k = x_{2,k+1}$, the physical tilted phase $\Psi_k(t)$ depends only on the current phase state $2\pi h x_{2,k}$ and the next phase state $2\pi h x_{2,k+1}$. After specifying the modulation index $h$, the phase state depends only on the value of $x_{2,k}$. Thus, we refer to this value $(x_{2,k} \in 0, 1, \ldots, M - 1)$ as the state of the CPFSK system at time $kT$. For instance, consider a 4-ary CPFSK system with $h = \frac{1}{4}$. In this case, $u_k$, $x_{1,k}$ and $x_{2,k}$ have possible values of 0, 1, 2 and 3. The physical tilted phase $\Psi_k(t)$ is shown in Figure 5.3. Notice that at time $(k + 1)T$, the state $x_{2,k+1}$ is equal to the previous input $u_k$.

![Fig. 5.3. The phase trellis diagram of 4-ary CPFSK, h=1/4](image-url)
Let \( x_2 = (x_{2,0}, x_{2,2}, \ldots, x_{2,K-1}) \) be the sequence of the states, where \( x_{2,k} \) is the state at \( t = kT \). Also, let \( z = (z_0, z_1, \ldots, z_{K-1}) \) be the observed signal sequence, where \( z_k \) is a continuous signal during each interval, \( kT \leq t \leq (k+1)T \). As previously mentioned in Section 3.2, the objective of the MAP sequence estimation process is to determine the state sequence \( x_2 \) such that the probability \( P(z|x_2)P(x_2) \) is maximum.

By using the derivation shown in Section 3.2, we obtain

\[
\arg \max_{x_2} P(z|x_2) = \arg \max_{x_2} \left[ -\sum_{k=0}^{K-2} \lambda(x_{2,k+1}, x_{2,k}) \right], \tag{5.12}
\]

where \( \arg \max \) is the operation that gives us the state sequence \( x_2 \) which makes \( P(z|x_2) \) maximum. Let each data sample of \( z_k \) be denoted as \( z_{k,i} \) where \( i = 1, 2, \ldots, N \). \( z_{k,i} = s_{k,i}^m + n_{k,i} \) where \( s_{k,i}^m = s_k^m(t)|_{t=(k+\frac{i-1}{N-1})T} \), \( i = 1, 2, \ldots, N \) is a data signal sample during time interval \( kT \leq t \leq (k+1)T \). Note that \( s_k^m(t) \) can be viewed as the function of the pair of states \( (x_{2,k+1}, x_{2,k}) \) defined as \( s_k^m(t) = \sqrt{2E \over T} [\cos(\Psi_k(t)) + \sin(\Psi_k(t))] \) where \( \Psi_k(t) = R_{2\pi} \left[ 2\pi h(x_{2,k} + x_{1,k} \frac{(t-kT)}{T}) \right] \), \( kT \leq t \leq (k+1)T \). The noise sample, \( n_{k,i} \), is a random variable having a zero-mean Gaussian distribution with variance \( N_0/2 \). Thus \( z_{k,i} \) is also a Gaussian random variable with mean \( s_{k,i}^m \) and variance \( N_0/2 \). Therefore we have

\[
P(z_k|x_{2,k+1}, x_{2,k}) = P(z_k|s_k^m) = \frac{1}{(\sqrt{\pi N_0})^N} \exp \left[ -\sum_{i=1}^{N} \frac{(z_{k,i} - s_{k,i}^m)^2}{N_0} \right]. \tag{5.13}
\]
The detail of this derivation can be found in [5, pp. 173]. Substituting Eq. (5.13) into Eq. (3.12) yields

\[
\lambda(x_{2,k+1}, x_{2,k}) = - \ln \left[ \frac{1}{(\sqrt{\pi N_0})^N} \right] + \sum_{i=1}^{N} \frac{z_{k,i}^2}{N_0} + \sum_{i=1}^{N} \frac{(s_{k,i}^m)^2}{N_0} - \sum_{i=1}^{N} \frac{2z_{k,i}s_{k,i}^m}{N_0} - \ln P(x_{2,k+1}|x_{2,k}).
\]  

(5.14)

Since the first three terms do not depend on the choice of the state sequence \(x_2\) obtained by Eq. (5.12), the branch metric can be reduced to

\[
\lambda(x_{2,k+1}, x_{2,k}) = - \sum_{i=1}^{N} z_{k,i}s_{k,i}^m - \frac{N_0}{2} \ln P(x_{2,k+1}|x_{2,k}),
\]  

(5.15)

where the first term is the inner product between the observed and transmitted signals within the time interval \(kt \leq t \leq (k+1)t\). When using this first term alone, we refer to the decoder as the maximum-likelihood (ML) decoder.

Now let \(u\) be the input sequence generated according to a specified source transition matrix \(C\) whose elements \(c_{i,j} = P(u_{k+1} = j|u_k = i), i, j = 0, 1, \ldots, M - 1\). The task here is to find the relationship between source states and phase state \(x_{2,k}\). From the feedback-free CPE block diagram, it is obvious that \(u_k = x_{2,k+1}\) and \(u_{k-1} = x_{2,k}\). Thus,

\[
P(x_{2,k+1}|x_{2,k}) = P(u_{k+1}|u_k) = P(u_{k+1}|u_{k-1}).
\]  

(5.16)

Substitute Eq. (5.16) into Eq. (5.14) to obtain a new branch metric,

\[
\lambda(u_{k+1}, u_k) = - \sum_{i=1}^{N} z_{k,i}s_{k,i}^m - \frac{N_0}{2} \ln P(u_{k+1}|u_k).
\]  

(5.17)
We note that the knowledge of the noise parameter $N_0$ is required in order to implement the optimal MAP decoder.

5.3.2 MAP Decoding Derivation for the CPFSK system with an external convolutional encoder

When an external convolutional encoder (CE) is used in conjunction with the CPFSK system (CPE+MM), the state of the overall encoder (CE+CPE) is of concern. Furthermore, a rate 1/2 CE is used in conjunction with a rate 1/2 CPE, and to be able to algebraically combine them into a single encoder, we need to first convert the CPE into an equivalent encoder of rate 2/4. To illustrate this, we first discuss the conversion in detail, and then we show how to combine the rate 1/2 CE with an equivalent encoder 2/4, resulting in the overall encoder of rate 1/4. From the block diagram show in Figure 5.4,

![Block Diagram of an Equivalent Encoder of Rate 2/4](image)

Fig. 5.4. The block diagram of an equivalent encoder of rate 2/4

we obtain four equations corresponding to the four outputs $x_{1,1,k}$, $x_{1,2,k}$, $x_{2,1,k}$ and
\[ x_{2,2,k} \text{ as follows:} \]

\[ x_{1,1,k} = b_{1,k} - b_{2,k-1}, \quad (5.18) \]
\[ x_{1,2,k} = b_{2,k-1}, \quad (5.19) \]
\[ x_{2,1,k} = b_{2,k} - b_{1,k}, \quad (5.20) \]
\[ x_{2,2,k} = b_{1,k}. \quad (5.21) \]

Note that the input \( b_{2,k-1} \) is the initial value of the state which is stored in the delay \( D \). That is why we have the first two equations. Moreover, the modulo sum of the first two equations results in the value of \( b_{1,k} \) (\( b_{1,k} = x_{1,1,k} + x_{1,2,k} \mod 4 \)) and for the modulo sum of the last two equations, we obtain the value of \( b_{2,k} = x_{2,1,k} + x_{2,2,k} \mod 4 \). Figure 5.5 shows all possible phase transitions produced from the two input symbols \( b_{1,k} \) and \( b_{2,k} \). We observe that the value of \( b_{2,k} \) is associated with the value of the phase state at time \( kT \). In this section, the notation of the phase state \( b_{2,k} \) is used in the same manner as that of the phase state \( x_{2,k} \) explained in section 5.3.1. The difference here is that \( b_{2,k} \) is updated every two signal intervals, while \( x_{2,k} \) is updated every signal interval. Therefore, we are now interested in finding \( b_{2,k} \) for which the conditional probability \( P(z_k|b_{2,k+1}, b_{2,k}) \) is maximum. Note that \( z_k \) is the observed signal within two signal intervals. This implies that the branch metric shown in Eq. 5.15 is modified as follows

\[ \lambda(b_{2,k+1}, b_{2,k}) = - \sum_{i=1}^{2N} z_{k,i} s_{k,i}^m - \frac{N_0}{2} \ln P(b_{2,k+1}|b_{2,k}). \quad (5.22) \]
Fig. 5.5. The phase trellis diagram of the block diagram shown in Figure 5.4.

Note that in this case, the maximum number of possible paths in the trellis that are required to compute the branch metric is 64. Moreover, the \( s^m_k \) is the transmitted signal over two signal intervals. The four input-output relationships shown in Eqs. 5.18-5.21 can be rewritten in another form as follows. This is similar to taking the \( z \)-transform of the previous four equations, but we use the parameter \( D \) instead of \( z \); also, \( D \) is not a complex variable but a delay variable.

\[
X_{1,1}(D) = b_1(D) - D b_2(D), \quad (5.23)
\]
\[
X_{1,2}(D) = D b_2(D), \quad (5.24)
\]
\[
X_{2,1}(D) = b_2(D) - b_1(D), \quad (5.25)
\]
\[
X_{2,2}(D) = b_1(D). \quad (5.26)
\]
Thus, we obtain the transfer function [15] as follows.

\[
X = B \begin{bmatrix} 1 & 0 & 3 & 1 \\ 3D & D & 1 & 0 \end{bmatrix},
\]

(5.27)

where \(X\) and \(B\) are row vectors corresponding to outputs \(\{X_{1,1}(D) \ X_{1,2}(D) \ X_{2,1}(D) \ X_{2,2}(D)\}\) and inputs \(\{B_1(D) \ B_2(D)\}\), respectively. Note that the number 3 comes from -1 modulo 4.

For example, if the input row vector \(B = [2D + 1 \ 1]\), we obtain the output row vectors \(X = [D + 1 \ D \ 2D \ 2D + 1]\). The block diagram of the overall encoder \(X\) is shown in Figure 5.6. Notice that the realization of the overall encoder shows the relationship between input symbol \(u_k\) and four output symbols \(x_{1,1,k} \ x_{1,2,k} \ x_{2,1,k} \ x_{2,2,k}\) without explicitly showing the value of the parameters \(b_{1,k} \ b_{2,k}\). However, we can compute these parameters from Eq.(5.18)-Eq.(5.21) described earlier. It is worth mentioning that the parameters \(b_{1,k} \ b_{2,k}\) enable us to determine the transition of the phase state shown in Figure 5.5.

Figure 5.7 shows the trellis diagram of the block diagram in Figure 5.6 as well as its corresponding phase trellis diagram. In this example, the total number of the states is four and the total number of paths in the trellis is 16. It is emphasized that each path in the trellis diagram corresponds to the phase transition of two signal intervals and the value of \(S_0\) is identical to the value of \(b_2\). Therefore, the branch metric in Eq. (5.22)
Fig. 5.6. The block diagram of overall encoder

Fig. 5.7. The trellis diagram and the phase trellis diagram of the block diagram shown in Figure 5.6.
can be rewritten as
\[ \lambda(S_{0,k+1}, S_{0,k}) = -\sum_{i=1}^{2N} z_{k,i}^m z_{k,i}^m - \frac{N_0}{2} \ln P(S_{0,k+1} | S_{0,k}). \] (5.28)

Next we would like to find the relationship between the state variable \( S_0 \) and input symbol \( u \). By observing the structure of the overall encoder shown in Figure 5.6, we found out that \( S_{0,k+1} = u_k \). This fact is also evident by observing (in Figure 5.7) that all branches entering the state \( S_{0,k+1} \) correspond to the input \( u_k = S_{0,k+1} \). Thus, the branch metric in Eq. (5.28) now becomes
\[ \lambda(u_k, u_{k-1}) = -\sum_{i=1}^{2N} z_{k,i}^m z_{k,i}^m - \frac{N_0}{2} \ln P(u_k | u_{k-1}). \] (5.29)

When the total state number of the overall encoder is greater than four, this implies that we may use \( P(u_k | u_{k-1}) \) more than one time. To see how one can combine \( P(u_k | u_{k-1}) \) into the branch metrics, let us first observe the structure of the overall encoder \( G_{all}(D) \), whose block diagram is shown in Figure 5.8. The block \( G'' \) contains only adders and multipliers connected according to the transfer function of \( G_{all}(D) \); \( G'' \) does not contain any delay elements. The external multipliers, \( a_1, \ldots, a_{m-1} \), are necessary whenever a reduction of the total number of states is possible. From Figure 5.8, one sees that the first delay \( S_0 \) always contains the value of the previous input \( u_{k-1} \). It implies that when we know the value of \( S_0 \), we then know what the previous input is. It also implies that at each state, all incoming branches are associated with the same input.
This knowledge allows us to incorporate the source transition probabilities \( P(u_k|u_{k-1}) \) into each branch metric.

For example in Figure 5.7, only one kind of input will put the encoder into each state. Thus, the appropriate branch metric can be easily determined. For instance, the branch metric that connects state (2,0) to (0,0), described as 0/2200, uses \( P(u_k = 0|u_{k-1} = 2) \). The reason for this is that for state (2,0), all incoming branches correspond to the input \( u_{k-1} = 2 \). Moreover, the output symbols 2200 correspond to the phase transition of the two-interval CPFSK signal shown in Figure 5.5. For a trellis having a total number of states more than four, one will have to use \( P(u_k|u_{k-1}) \) more than once. The method for automatically incorporating \( P(u_k|u_{k-1}) \) into all branch metrics must be implemented. Note that the method is also necessary for performing a code search.

The technique we use here is that for a given state number \( SN \), the value of \( S_0 \) is automatically computed as follows

\[
S_0 = SN \mod 4, \quad (5.30)
\]
where \( \text{mod} \) indicates a modulo operation. For instance, when the state number is five, the value of \( S_0 \) becomes \( 5 \text{ mod } 4 = 1 \). It is worth mentioning that from the realization of the overall encoder shown in Figure 5.8, the state number \( S_N \) can be computed by

\[
S_N = S_0 + S_1 D_1 + S_2 D_2^2 + \cdots + S_{m-1} D_{m-1}^{m-1}.
\]

(5.31)

where

\[
D_i = \begin{cases} 
2, & i = k, k + 1, \ldots, m - 1, \\
4, & i < k,
\end{cases}
\]

and \( k \) is the minimum index \( i \) such that \( a_i = 2 \). Note that in the case when none of the external multipliers \( a_i \) is equal to two, \( D_i = 4 \), for all \( i \). To verify the validity of Eq. 5.30, we simply substitute Eq. 5.31 into Eq. 5.30 and then show that it can be reduced to \( S_0 \). Observe that the terms \( S_2 D_2^2, S_3 D_3^3, \ldots, S_{m-1} D_{m-1}^{m-1} \) contain a factor of 4 since \( D_i \) is either 2 or 4. As a result, the modulo 4 of these terms are identically zero. For the term \( S_1 D_1 \), we separate into two cases:

- **Case I when \( D_1 = 4 \):** the term \( S_1 D_1 \) contains a factor of 4 and so it is reduced to zero after a modulo 4 operation;

- **Case II when \( D_1 = 2 \):** the definition of \( D_i \) implies that \( a_1 = 2 \). In this case, \( S_1 \) can only be either 0 or 2 since from the realization of the overall encoder \( S_1 \) must contains a factor of \( a_1 \). Thus, the term \( S_1 D_1 \) also contains a factor of 4 and it is reduced to zero after a modulo 4 operation.
After we know the value of $S_{0,k}$, we then use it as the value of $u_{k-1}$. Thus, at state number SN, the branch metric that corresponds to the current input $u_k$ is incorporated with the source transition probability $P(u_k|u_{k-1}) = P(S_{0,k+1}|S_{0,k})$.

### 5.4 MAP Decoding for Fading Channel

By assuming sufficiently slow fading, the phase shift $\phi$ is usually assumed to be zero. Thus the received equivalent lowpass signal obtained in Eq. (5.7) is reduced to

$$r_l(t) = (1 + \beta) s_l^m(t) + n_{wl}(t). \quad (5.32)$$

To construct the MAP decoder for the CPFSK system, we again observe the data samples $z_{k,i}$. During time interval $kT \leq t \leq (k+1)T$, the observed samples can be written as $z_{k,i} = (1 + \beta_{k,i}) s_{k,i}^m + n_{k,i}$ where $i = 1, 2, ..., N$. For a slowly fading channel, $\beta_{k,i}$ is a constant during each signal interval. Thus the fading parameter $\beta_{k,i}$ is then denoted as $\beta_k$. Al Semari et al. [42] assume that the fading values can be determined, i.e. they estimate channel state information (CSI). Therefore, this information can be incorporated into the decoding metric. We first follow this assumption. However, we shall show later that by using only the mean value of $\beta_k$, no estimation is necessary, and some improvement is still obtained.

Assuming CSI, the conditional probability becomes

$$P(z_k|x_{2,k+1}, x_{2,k}) = P(z_k|s_k^m) = \frac{1}{(\sqrt{\pi N_0})^N} \exp \left[ -\sum_{i=1}^{N} \frac{(z_{k,i} - \alpha_{k,i} s_{k,i}^m)^2}{N_0} \right], \quad (5.33)$$
where $\alpha_k = (1 + \beta_k)$. By substituting Eq. (5.33) into Eq. (3.12) and simplifying as in Eq. (5.17), we obtain the branch metric $\lambda(x_{2,k+1}, x_{2,k})$ as

$$
\lambda(x_{2,k+1}, x_{2,k}) = -\sum_{i=1}^{N} z_{k,i}s_{k,i}^{m} - \frac{N_0}{2\alpha_k} \ln P(x_{2,k+1}|x_{2,k}).
$$

(5.34)

Again by using Eq. (5.16) and the fact that the input sequence $u$ is characterized by a Markov process, the new branch metric becomes

$$
\lambda(u_k, u_{k-1}) = -\sum_{i=1}^{N} z_{k,i}s_{k,i}^{m} - \frac{N_0}{2\alpha_k} \ln P(u_k|u_{k-1}).
$$

(5.35)

When an external encoder is included in the system, we have to consider the conditional probability $P(z_k|S_{0,k+1}, S_{0,k})$ instead of $P(z_k|x_{2,k+1}, x_{2,k})$ as explained in Section 5.3.2. The difference here is that the effect of the fading amplitude is included in the computation of the conditional probability $P(z_k|S_{0,k+1}, S_{0,k})$ via the relationship in Eq. 5.33. Thus, the branch metric becomes

$$
\lambda(S_{0,k+1}, S_{0,k}) = -\alpha_{k,1} \sum_{i=1}^{N} z_{k,i}s_{k,i}^{m} - \alpha_{k,2} \sum_{i=N+1}^{2N} z_{k,i}s_{k,i}^{m} - \frac{N_0}{2} \ln P(S_{0,k+1}|S_{0,k}),
$$

(5.36)

where $\alpha_{k,1}$ and $\alpha_{k,2}$ are the constant fading amplitude during the first and second signal interval, respectively.
Chapter 6

Design of Ring Convolutional Trellis Codes
for MAP Decoding

In this chapter we consider the design of trellis ring convolutional codes over $\mathbb{Z}_4$ combined with CPFSK for MPEG-4 image transmission over the AWGN channel. In addition to an MPEG-4 image coding process, the 4-ary zerotree symbols of the HFS of an image are modeled as a first order Markov source, which can be characterized by a source transition matrix (STM). The STM is then used in the design of trellis codes optimized for this particular source and channel. In previous work [34], the STM of the zerotree symbols of the HFS of the “Lena” image in multi-quantization mode is used in the design process. We continue this work by further considering longer constraint length codes and a different image, i.e. the “Barbara” image.

6.1 Trellis Ring Convolutional Coded CPFSK

The purpose of this section is to describe how the design of trellis coded modulation with ring convolutional codes can be achieved. The criterion for seeking a desirable code is similar to the case of binary convolutional codes. The optimal ring convolutional codes are designed by maximizing the minimum squared Euclidean distance, $D^2_{\text{free}}$. In general, we can express the squared Euclidean distance between two signal sequences as

$$D^2(X, X') = \int_0^{KT} \left[ sm(t, X) - sm(t, X') \right]^2 dt, \quad (6.1)$$
where $X$ and $X'$ are two different codeword sequences, and $K$ is the number of signal intervals. The $D_{\text{free}}^2$ is defined as

$$D_{\text{free}}^2 = \min_{X \neq X'} D^2(X, X').$$  \hspace{1cm} (6.2)

Note that the $D_{\text{free}}^2$ no longer depends on the number of signal intervals $K$ if we set $K$ to be large enough. In a Gaussian channel the $D_{\text{free}}^2$ gives a good estimate for the probability of error if the signal to noise ratio is not too small. In fact, the probability of an erroneous decision can be upper bounded by using a union bound method [24] as

$$P_e \leq \sum_X P(X) \sum_{X': X' \neq X} Q\left(\sqrt{\frac{D^2(X, X')}{2N_0}}\right),$$  \hspace{1cm} (6.3)

where

$$Q(x) = \frac{1}{\sqrt{2\pi}} \int_x^\infty e^{-\frac{t^2}{2}} dt.$$  \hspace{1cm} (6.4)

When the noise power is getting small, the summation in Eq. (6.3) can be approximately reduced [24] to

$$P_e = Q\left(\sqrt{\frac{D_{\text{free}}^2}{2N_0}}\right).$$  \hspace{1cm} (6.5)

From the definition of the function $Q(x)$ given in Eq. (6.4), one can observe that when $x$ increases, the function $Q(x)$ decreases, and so does the probability of error $P_e$. Thus in order to have the smallest $P_e$, we seek an optimal code that provides the maximum of the $D_{\text{free}}^2$. By assuming that the observation interval length is large enough, the $D_{\text{free}}^2$ can be found for each code. Conceptually, the computation for $D_{\text{free}}^2$ is to compare
the distance of all codewords and select for the minimum distance. Since the number of codewords increases (exponentially) when the constraint length of a code increases (The observation interval length increases), the computation for $D_{\text{free}}^2$ becomes significantly complicated. Fortunately, if a convolutional code is linear, it’s $D_{\text{free}}^2$ can be found regardless of the reference codeword (mentioned in Section 3.1.1). We often use all-zero codeword as the reference codeword. Moreover, we can apply the Viterbi algorithm in order to compute $D_{\text{free}}^2$.

6.2 Code Design Description for MAP Decoding

As previously known, the minimum squared Euclidean distance, $D_{\text{free}}^2$, plays an important role in a code design for a Gaussian channel. This fact is based on a union bound analysis illustrated in Eq. (6.3). When the SNR is large, such a bound is dominated by one term depending on $D_{\text{free}}^2$. When the SNR is getting small, the approximation of the probability of error in Eq. (6.5) becomes inappropriate. Kroll and Phamdo [40] considered the design of the optimal trellis codes for the binary symmetric Markov source (BSMS) with the crossover transition probability $p = 0.9$, and a MAP decoder over a Gaussian channel. They investigated the analysis of a union bound at low SNR. Please refer to [40] for the detail. In summary, a union bound on error probability $P_e$ may be written as

$$P_e \leq \sum_X P(X) \sum_{X', X' \neq X} Q \left( \sqrt{\frac{D^2(X, X')}{{2}N_0}} - \sqrt{\frac{N_0}{{2}D^2(X, X')}} \ln \left( \frac{P(X)}{P(X')} \right) \right),$$

(6.6)
where \( P(X) \) and \( P(X') \) are the \textit{a priori} probabilities of the codeword sequences \( X \) and \( X' \), respectively. Note that when \( P(X) \) is equal to \( P(X') \), the bound in Eq. (6.6) is reduced to the union bound previously shown in Eq. (6.3). The bound in Eq. (6.6) is employed to find optimal trellis codes customized for a particular source-channel model.

Kroll and Phamdo [41] continue their work by considering the design of trellis codes for text document image transmission over the AWGN channel. The data bit stream of such an image is modeled as a first order binary \textit{asymmetric} Markov source (BAMS). Based on observations, the transition probability matrix of a typical text document is assumed to be [41]

\[
P = \begin{bmatrix}
0.994 & 0.006 \\
0.147 & 0.853
\end{bmatrix}.
\]  

(6.7)

Our work differs from Kroll and Phamdo’s works in two main points. First, we are interested in the design of the optimal ring convolutional codes for 4-ary symbols, which represent zerotree symbols in the HFS of an image. In fact, we investigate the inclusion of a source transition probability matrix (STM) into the design of trellis coded modulation system consisting of ring convolutional codes and CPFSK modulation system. For a particular STM, the goal is to find the optimal nonsystematic ring convolutional code at the symbol error rate of interest. We will see later on that the structure of STM of zerotree symbols is not as good as that of text document considered in [41]. By observing the branch metric in Eq. (5.15), we know that MAP decoding affects the system performance when the noise power \( N_0 \) is not too small. Thus we consider the design of trellis codes at high symbol error rates (SERs) \( (10^{-2} - 10^{-3}) \). Secondly, we modify the squared Euclidean distance in such a way that it includes a source transition
matrix, instead of attempting to compute the union bound shown in Eq.(6.6). Thus we can reduce the complexity of the computation in the code design process. This modified distance replaces the SED in order to find the minimum distance of each code. And finally, the selected code is a code, in which, its minimum distance is maximum.

In this section, the modified squared Euclidean distance is proposed. Our derivation here is based on the derivation of pairwise error event probability for MAP decoder, given in [40]. Let \( Z \) be a received sequence, \( X \) be the correct transmitted sequence, and \( X' \) be the incorrect sequence. The MAP decoder wrongly chooses \( X' \) over \( X \) if

\[
B \triangleq [D_{map}(X, Z) - D_{map}(X', Z)] > 0, \tag{6.8}
\]

where

\[
D_{map}(X, Z) = \sum_{k=0}^{K-1} (D_k^2(X, Z) - N_0 \ln P(x_k | x_{k-1})) \text{ and } \tag{6.9}
\]

\[
D_{map}(X', Z) = \sum_{k=0}^{K-1} (D_k^2(X', Z) - N_0 \ln P(x'_k | x'_{k-1})). \tag{6.10}
\]

Note that the SED between two sequences can be simplified to

\[
D^2(X, X') = \sum_{k=0}^{K-1} D_k^2(X, X'), \tag{6.11}
\]

where \( D_k^2(X, X') \) is the incremental squared Euclidean distance (ISED) and \( K \) is the number of observation intervals. The ISED can be calculated by using the input of the MM.
With some simplification, the random variable $B$ can be reexpressed as

$$B = \sum_{k=0}^{K-1} \left[ D_k^2(X, Z) - D_k^2(X', Z) + N_0 \ln \frac{P(x_k|x_{k-1})}{P(x'_k|x'_{k-1})} \right].$$

In [39], it has been shown that the probability density function of the random variable $B$ is of normal distribution with the mean of $\sum_{k=0}^{K-1} \left[ -D_k^2(X, X') + N_0 \ln \frac{P(x_k|x_{k-1})}{P(x'_k|x'_{k-1})} \right]$ and the variance of $2N_0 \sum_{k=0}^{K-1} D_k^2(X, X')$. Before defining the code search criterion, let make the following two observations.

1. The mean of the random variable $B$ does not depend on the received sequence $Z$.
2. The smaller the random variable $B$, the less likely that the MAP decoder would make a wrong decoding decision and thus reducing the bit error rate.

Based on these observations, for MAP decoding, the code search criterion is proposed as finding the code whose codewords minimize the mean of the random variable $B$, i.e. finding the code whose codewords produce the largest modified minimum squared Euclidean distance,

$$D_{map}(X, X') = -\text{mean of the random variable } B$$

$$= \sum_{k=0}^{K-1} \left[ D_k^2(X, X') - N_0 \ln \frac{P(x_k|x_{k-1})}{P(x'_k|x'_{k-1})} \right].$$

(6.12)

The term inside the summation is used as the modified branch metric in order to find the minimum squared Euclidean distance. In the special case when the search of the best regular code is implemented by computing the distance between all codewords and
the all-zero sequence, one can replace the sequence $X'$ by an all-zero sequence $0$ and the condition probability $P(x'_k|x'_{k-1})$ by 1, resulting in the following simplified criterion.

$$D_{map}(X, 0) = \sum_{k=0}^{K-1} \left[ D_k^2(X, 0) - N_0 \ln P(x_k|x_{k-1}) \right].$$

Furthermore when the conditional probability $P(x_k|x_{k-1}) = P(x'_k|x'_{k-1})$, Eq. (6.12) reduces to the ML code search criterion in Eq. (6.11). To search for the best code for MAP decoding, the modified SED in Eq. (6.12) replaces the SED in Eq. (6.11) in order to determine the minimum SED. It is important to notice that our modified branch metric in Eq. (6.12) is effectively the same as the function $\sqrt{\frac{2D^2(X,X')}{N_0}}Y$, where $Y$ is the term inside the $Q$ function in Eq. (6.6). When a code maximizes the function above, this code will make $Y$ obtain its largest value. This then leads to the lowest value of $Q(Y)$ and subsequently the minimum BER.

### 6.3 Suboptimal Code Design Description

From Section 6.2, we conclude that in order to search for the optimal MAP code, the modified branch metric in Eq. (6.12) is used (instead of the metric in Eq. (6.11)) to compute the modified $D_{free}^2$. Again, we can use Viterbi algorithm to calculate the modified $D_{free}^2$. However, the process of finding the modified $D_{free}^2$ cannot be simplified by using only all zero codeword as the reference codeword. The reason is that the Euclidean distance (from Eq. (6.12)) between two codeword sequences (corresponding to distinct information sequences) does not depend only on the binary sum of the two information sequences. It also depend on the conditional probabilities of the information
sequences. Thus, the regularity property [36] of a trellis code does not hold. Thus, the modified $D^2_{free}$ found from using only all zero codeword as the reference codeword, may not be the same as the modified $D^2_{free}$ found from comparing all codewords. Note that the modified $D^2_{free}$ from the latter case is required to find the optimal MAP code. If we use only all zero codeword as the reference codeword to find the modified $D^2_{free}$, we only find suboptimal MAP code.

The following initial experiment illustrates the different results coming from the optimal and suboptimal MAP code in term of the image peak signal-to-noise ratio (PSNR). Table 6.1 summarizes the results for 4 state trellis ring convolutional codes. Two design approaches are used. The first one compares all non-zero codewords to the all zeros codeword, while the second one compares all codewords. Consider the “Lena” image; the best code obtained using the first approach is $[3D + 3, 3]$. Using this code with MAP decoding provides a gain of 0.227 dB over ML decoding of the same code. Yet the optimal code, found by comparing all codewords, is $[3D + 2, 3]$, which yields a further improvement of 0.119 dB (16.753-16.634 dB). Similar results apply for the “Barbara” image, as shown in the bottom of the table.

Thus, the regularity property [36] does not hold, and one should really design the codes by comparing all codewords. This latter approach is computationally intractable. For an 8 state code and only 10 symbol intervals, there are $32 \times 4^9 = 8,388,608$ sequences. So, we choose the first approach, which is suboptimal. All further results in this paper are obtained in this manner.
Table 6.1. Initial image coding results for 4 state codes. (a) Lena and (b) Barbara. Multi-quantization mode. 200 image trials per data point.

### 6.4 Code Search Results

In this section, the suboptimal ring convolutional codes for the 4-ary zerotree symbols are obtained. For a particular image, characterized by its STM, the goal is to find the polynomial ring convolutional code at the symbol error rate of interest. The code’s generator matrix and the STM are sent in the first packet as side information, using a suitable block code (or ARQ) to ensure correct reception.

For comparison purposes, the code search using the ML criterion is first done for all noncastastrophic nonsystematic (polynomial) codes. Table 6.2 illustrates the best rate 1/2 polynomial ring convolutional encoders and their maximum $d^2_{free}$ for the given total number of states of the overall encoders. Note that $d^2_{free}$ is a normalized $D^2_{free}$. That is $d^2_{free} = D^2_{free}/2E_s$. Since we are considering 4-ary symbols, we actually divide by $2E_s$ instead of $2E_b$. Please refer to Eq. (16) in [15] for the detail. The codes in Table 6.2 are called as the “best ML” codes. Note that these nonsystematic ring convolutional
encoders give larger $d_{\text{free}}^2$ than the systematic ring convolutional encoders found in [15].

Also, note that the $d_{\text{free}}^2$ of a code allows us to roughly predict the system performance when using the code at high energy per symbol to noise power spectral density, $E_s/N_0$. In this section we are interested in seeking for the suboptimal codes for MAP decoding. These codes are called the “MAP” codes. We consider four cases based on the use four source transition matrices (STM) generated from the “Lena” and “Barbara” images.

**Case 1:** The STM, computed from the zerotree symbol of the HFS of the “Lena” image with single quantization mode, is used in the design process. For convenience, this STM, previously shown in Eq. (2.7), is repeated here as

$$
\begin{bmatrix}
0.0389 & 0.1001 & 0.5172 & 0.3438 \\
0.0414 & 0.1774 & 0.4227 & 0.3585 \\
0.0487 & 0.1668 & 0.4086 & 0.3759 \\
0.0516 & 0.2327 & 0.4142 & 0.3015
\end{bmatrix}
$$

(6.13)
The code search for MAP decoding is performed at a symbol error rate (SER) of approximately $10^{-3}$, which corresponds to an $E_s/N_0$ of about 4.5 dB for 4 and 8 states, 3.5 dB for 16 and 32 states, and 3 dB for 64 and 128 states. Note that the $E_s/N_0$ is the energy per channel symbol, while the SER is for the information symbols. In this case the codes for 4, 8, 16, and 32 states are the same for both MAP and ML decoding. Note that the best ML codes are previously shown in Table 6.2. For the case of 64 and 128 states, the suboptimal MAP codes are different from the best ML codes. Figure 6.1 illustrates the performance of the best ML and suboptimal MAP codes for $m = 64$ and $m = 128$. ML and MAP decoding are used for the best ML codes and suboptimal MAP codes, respectively. For the 64 states code, the improvement is approximately 0.5-0.3 dB

![Fig. 6.1. Symbol error rate (SER) vs. signal-to-noise ratio $E_s/N_0$ for the STM given by Eq. (6.13). The best ML codes use ML decoding, while the suboptimal MAP codes use MAP decoding. The parameter $m$ is the overall number of states in the trellis.](image)
at SERs from $10^{-3}$ to $10^{-4}$. For the 128-state code, the improvement is about 0.5-0.3 dB at the same SER range. Note that these improvements come from two factors; the use of the MAP codes and MAP decoding. The improvements due to the MAP codes are found to be approximately 0.2 and 0.13 dB at SER about $10^{-3}$ for the 64-state codes and the 128-state codes, respectively. In this experiment, MAP decoding is used for both the best ML and suboptimal MAP codes. In summary, the MAP codes and the total improvements (due to the use of both the MAP codes and MAP decoding) at $SER = 10^{-3}$ are shown in Table 6.3. Since the best ML and MAP codes are the same for the 4, 8, 16, and 32-state codes, there is no improvement due to the MAP codes. However, there is still some improvement due to the use of MAP decoding. In Chapter 7, we investigate the performance comparison between ML and MAP decoding for the same code.

### Case 2:

The STM, computed from the zerotree symbols of the HFS of the "Lena" image with multi-quantization mode, is used in the design process. In this case, the STM

<table>
<thead>
<tr>
<th>$S_t$</th>
<th>$G$</th>
<th>$d_{free}^2$ (MAP)</th>
<th>Improvement at SER=$10^{-3}$ (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>$[3D^3 + 3, 3]$</td>
<td>3.58</td>
<td>0.08</td>
</tr>
<tr>
<td>8</td>
<td>$[2D^2 + 3D + 3, 3]$</td>
<td>4.36</td>
<td>0.25</td>
</tr>
<tr>
<td>16</td>
<td>$[3D^2 + D + 2, 2D + 3]$</td>
<td>5.79</td>
<td>0.31</td>
</tr>
<tr>
<td>32</td>
<td>$[2D^3 + D^2 + D + 3, D + 2]$</td>
<td>6.03</td>
<td>0.44</td>
</tr>
<tr>
<td>64</td>
<td>$[D^3 + 3D^2 + 2D + 3, 2D^2 + 3D + 2]$</td>
<td>7.09</td>
<td>0.50</td>
</tr>
<tr>
<td>128</td>
<td>$[3D^3 + 2D^2 + D + 2, 2D^3 + D^2 + 3D + 1]$</td>
<td>7.61</td>
<td>0.50</td>
</tr>
</tbody>
</table>

Table 6.3. The rate 1/2 nonsystematic ring convolutional encoders over $Z_4$ for 4-ary CPFSK with $h=1/4$ for MAP decoding. The STM in Eq. (6.13) is used. $S_t$ denotes the total number of states in the overall encoder.
Like the previous case, the code search for MAP decoding is obtained at a symbol error rate (SER) of approximately $10^{-3}$, which corresponds to an $E_s/N_0$ of 4.5 dB for 4 and 8 states, 3.5 dB for 16 and 32 states, and 3 dB for 64 and 128 states. For 4, 8, and 16 states, the MAP and ML codes are the same. For 32, 64, and 128 states, the MAP codes are different from the best ML codes. Here, the performance of the MAP codes is
Table 6.4. The rate 1/2 nonsystematic ring convolutional encoders over $\mathbb{Z}_4$ for 4-ary CPFSK with $h=1/4$ for MAP decoding. STM in Eq. (6.14) is used. $S_t$ denotes the total number of states in the overall encoder.

<table>
<thead>
<tr>
<th>$S_t$</th>
<th>$G$</th>
<th>$d_{fg}^2$ (MAP)</th>
<th>Improvement at SER=$10^{-3}$ (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>$[3D + 3, 3]$</td>
<td>3.58</td>
<td>0.23</td>
</tr>
<tr>
<td>8</td>
<td>$[2D^2 + 3D + 3, 3]$</td>
<td>4.36</td>
<td>0.19</td>
</tr>
<tr>
<td>16</td>
<td>$[3D^2 + D + 2, 2D + 3]$</td>
<td>5.79</td>
<td>0.17</td>
</tr>
<tr>
<td>32</td>
<td>$[D^2 + 3D + 1, 2D^2 + 3D + 2]$</td>
<td>5.67</td>
<td>0.31</td>
</tr>
<tr>
<td>64</td>
<td>$[D^3 + 2D^2 + D + 3, 2D^2 + 3D + 2]$</td>
<td>6.82</td>
<td>0.38</td>
</tr>
<tr>
<td>128</td>
<td>$[3D^3 + 2D^2 + 3D + 2, 2D^3 + D^2 + D + 3]$</td>
<td>7.88</td>
<td>0.44</td>
</tr>
</tbody>
</table>

shown in Figure 6.2. For the 32-state codes, we gain about 0.31-0.25 dB at SERs from $10^{-3}$ to $10^{-4}$. For the 64-state codes, the improvement is approximately 0.38-0.31 dB at SERs from $10^{-3}$ to $10^{-4}$. At the same SER range, the improvement is from 0.44-0.38 dB for the 128-state codes. Note that the improvements due to the use of the MAP codes for the 32, 64, and 128-state codes are approximately 0.1, 0.16, and 0.06 dB at $SER = 10^{-3}$, respectively. In summary, the MAP codes and the total improvements are shown in Table 6.3. Note that for 4, 8, and 16 states, the improvements at SER of $10^{-3}$ are from the use of MAP decoding.

**Case 3:** We use the STM computed from the zerotree symbol of the HFS of the “Barbara” image with single quantization mode in the design process. We obtain the
STM, $P$, as

$$P = \begin{bmatrix}
0.4422 & 0.4454 & 0.0770 & 0.0354 \\
0.3999 & 0.4614 & 0.0797 & 0.0590 \\
0.1761 & 0.1619 & 0.3804 & 0.2816 \\
0.0930 & 0.2066 & 0.4047 & 0.2957
\end{bmatrix}. \quad (6.15)$$

Again the code search for MAP decoding is obtained at a symbol error rate (SER) of approximately $10^{-3}$ which corresponds to an $E_s/N_0$ of 4.5 dB for the 4 and 8-state codes, 3.5 dB for the 16 and 32-state codes, and 3 dB for the 64 and 128 state codes.

![Symbol error rate vs. $E_s/N_0$ for the source transition matrix (STM) given by Eq. (6.15) for using the best ML and suboptimal MAP codes. The parameter $m$ is the overall number of states in the trellis.](image)

For 4, 8, 16, and 32 states, the MAP and ML codes are the same. For 64 and 128 states, the MAP codes are different from the best ML codes, and their performances in term of
Table 6.5. The rate 1/2 nonsystematic ring convolutional encoders over $\mathbb{Z}_4$ for 4-ary CPFSK with $h=1/4$ for MAP decoding. STM in Eq. (6.15) is used. $S_t$ denotes the total number of states in the overall encoder.

<table>
<thead>
<tr>
<th>$S_t$</th>
<th>$G$</th>
<th>$d_{free}^2$ (MAP)</th>
<th>Improvement at $\text{SER}=10^{-3}$ (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>$[3D + 3, 3]$</td>
<td>3.58</td>
<td>0.21</td>
</tr>
<tr>
<td>8</td>
<td>$[2D^2 + 3D + 3, 3]$</td>
<td>4.36</td>
<td>0.33</td>
</tr>
<tr>
<td>16</td>
<td>$[3D^2 + D + 2, 2D+3]$</td>
<td>5.79</td>
<td>0.38</td>
</tr>
<tr>
<td>32</td>
<td>$[2D^3 + D^2 + D + 3, D + 2]$</td>
<td>6.03</td>
<td>0.47</td>
</tr>
<tr>
<td>64</td>
<td>$[D^3 + 3D^2 + 2D + 1, 2D^2 + 3D + 2]$</td>
<td>7.09</td>
<td>0.55</td>
</tr>
<tr>
<td>128</td>
<td>$[2D^4 + D^3 + 2D^2 + 3D + 2, 3D^2 + D + 1]$</td>
<td>7.52</td>
<td>0.50</td>
</tr>
</tbody>
</table>

SER vs. $E_s/N_0$ are shown in Figure 6.3. For the 64-state codes, the total improvement is approximately 0.55-0.4 dB at SER from $10^{-3}$ to $10^{-4}$. The improvement due to the use of the MAP code is about 0.16 at SER of $10^{-3}$. For the 128-state codes, the total improvement is about 0.5-0.4 dB at the same SER range. In this case, we obtain about 0.06 dB improvement due to the use of the MAP code. Table 6.5 shows the MAP codes and the total improvements at SER of $10^{-3}$.

**Case 4:** We use the STM, $P$, shown in Eq. (6.16) in the design process. This STM is computed from the zerotree symbol of the HFS of “Barbara” image with multi-quantization mode.

$$
\begin{bmatrix}
0.5119 & 0.2930 & 0.1514 & 0.0437 \\
0.5751 & 0.2633 & 0.1190 & 0.0426 \\
0.2441 & 0.0880 & 0.4989 & 0.1690 \\
0.1878 & 0.1048 & 0.5328 & 0.1746
\end{bmatrix}
$$

(6.16)
As previously mentioned, the code search for MAP decoding is obtained at a symbol error rate (SER) of approximately $10^{-3}$, which corresponds to an $E_s/N_0$ of 4.5 dB for 4 and 8 states, 3.5 dB for 16 and 32 states, and 3 dB for 64 and 128 states. The codes for 4, 8, 16 and 32 states are the same for both MAP and ML decoding. In other words, the MAP codes are identical to the best ML codes. For 64 and 128 states the MAP codes are different from the best ML codes. Figure 6.4 illustrates the system performance of the best ML and suboptimal MAP codes using ML and MAP decoding, respectively. For the 64-state codes the total improvement is approximately 0.5-0.45 dB at SER from $10^{-3}$ to $10^{-4}$. For the 128 state code the improvement is obtained about 0.5-0.4 dB at
Table 6.6. The rate 1/2 nonsystematic ring convolutional encoders over $Z_4$ for 4-ary CPFSK with $h=1/4$ for MAP decoding. STM in Eq. (6.16) is used. $S_t$ denotes the total number of states in the overall encoder.

<table>
<thead>
<tr>
<th>$S_t$</th>
<th>$G$</th>
<th>$\frac{d_{\text{free}}^2}{\text{(MAP)}}$</th>
<th>Improvement at $10^{-3}$ (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>$[3D^3+3, 3]$</td>
<td>3.58</td>
<td>0.25</td>
</tr>
<tr>
<td>8</td>
<td>$[2D^2+3D+3, 3]$</td>
<td>4.36</td>
<td>0.33</td>
</tr>
<tr>
<td>16</td>
<td>$[3D^2+D+2, 2D+3]$</td>
<td>5.79</td>
<td>0.33</td>
</tr>
<tr>
<td>32</td>
<td>$[2D^3+D^2+D+3, D+2]$</td>
<td>6.03</td>
<td>0.46</td>
</tr>
<tr>
<td>64</td>
<td>$[3D^3+3D^2+2D+3, 2D^2+D+2]$</td>
<td>6.82</td>
<td>0.50</td>
</tr>
<tr>
<td>128</td>
<td>$[2D^4+D^3+2D^2+3D+2, 3D^2+D+1]$</td>
<td>7.52</td>
<td>0.50</td>
</tr>
</tbody>
</table>

In conclusion, different codes are found at 64 and 128 states for the four cases. The total improvements of about 0.31 to 0.55 dB are obtained at SER of about $10^{-3}$. Here we gain about 0.06 to 0.2 dB by using the MAP codes (instead of the best ML codes) for MAP decoding. For the binary case, Kroll and Phamdo [41] discovered that different codes were optimal at low and high bit error rates. There are a number of differences between their system and ours. Firstly, we use rate 1/2 polynomial ring convolutional codes with 4-ary CPFSK, instead of rate 2/3 systematic binary convolutional codes with 8-PSK. Secondly, the source transition matrix is generated from a real image. Since there are four zerotree symbols compared to two symbols (black and white) in [41], the source transition matrix is not as structured. Moreover, the MAP decoding is applied only for the zerotree symbols of the HFS of an image, not for the rest of the HFS and the LFS of an image where ML decoding is used.
Chapter 7

Symbol Error Rate Simulation Results

7.1 Channel Coding Simulation Results

In this chapter, we are interested in investigating the performance of source-controlled channel decoding compared to conventional ML decoding in terms of symbol error rate (SER) vs. signal to noise ratio (SNR). Instead of using the complete system shown in Figure 5.1, we consider only the part of the system without image coding/decoding, as illustrated in Figure 7.1. The input sequence $u$ is generated based on the particular Markov source, which can be characterized by its source transition matrix (STM). To consider a realistic case, we use the source transition probability matrix

---

Fig. 7.1. System block diagram for a Markov source consisting of a quaternary input sequence using a ring convolutional encoder and a continuous phase modulation scheme.
(STM) of the quaternary zerotree symbols in the higher frequency subbands (HFS) of the “Lena” image. The MPEG-4 single quantization mode, considered in [23], is chosen here to generate the input sequence. This quaternary symbol sequence is then coded by trellis-coded CPFSK with a ring convolutional code. And then its corresponding signal is transmitted over either the AWGN or Rayleigh fading channel. At the receiver, we approximate this signal by using the sampling method. We then obtain the received sequence which will then be used in the branch metric computation process. Note that instead of using the sampling technique to represent a signal waveform, one may apply the technique known as the matched filter method. Please refer to [1] for the detail.

7.1.1 Examples and Simulation Results for Uncoded CPFSK

These particular examples emphasize the usage of quaternary CPFSK with $h=1/4$, since its bandwidth is approximately the same as that of MSK. Consider a quaternary input sequence $u = (u_0, u_1, u_2, \ldots)$, where $u_k, k = 0, 1, 2, \ldots$ are generated according to the source transition probability matrix (STM)

$$
\begin{bmatrix}
0.8 & 0.1 & 0.05 & 0.05 \\
0.1 & 0.7 & 0.1 & 0.1 \\
0.03 & 0.07 & 0.8 & 0.1 \\
0.04 & 0.06 & 0.1 & 0.8
\end{bmatrix}.
$$

(7.1)

Note that this STM is chosen as a test case because it has a good structure (a lot of redundancy). Please refer to Section 2.2.1 for the computation of residual redundancy. Now, we use quaternary CPFSK modulation with $h=1/4$. The associated trellis consists
of 4 phase states and 16 branches, with each corresponding input symbol denoted by $u_k$, and output symbols denoted by $x_{1,k}$ and $x_{2,k}$. The resulting graph of symbol error rate (SER) versus signal-to-noise-ratio (SNR), $E_s/N_0$, is shown in Figure 7.2. In the simulation process, it is assumed that the variance of the noise, $N_0$, is known. Note that for the range of SER between $10^{-1}$ and $10^{-2}$, the improvement is approximately 1 dB. When the SNR is relatively high, the improvement becomes less obvious. This is not surprising because the second term of Eq. (5.17) tends to zero as SNR grows; this implies that the optimal MAP decoder is converging to the ML decoder. In order to make our simulation more meaningful, we use the source transition matrix of the zerotree symbols produced by the MPEG-4 embedded zerotree wavelet encoder. For coding the

Fig. 7.2. Symbol Error Rate (SER) vs. $E_s/N_0$. Not using source transition probability matrix (solid line), Using source transition probability matrix (dashed line).
Lena image in single quantization mode [23], we obtain STM as shown below.

\[
\begin{bmatrix}
0.0389 & 0.1001 & 0.5172 & 0.3438 \\
0.0414 & 0.1774 & 0.4227 & 0.3585 \\
0.0487 & 0.1668 & 0.4086 & 0.3759 \\
0.0516 & 0.2327 & 0.4142 & 0.3015
\end{bmatrix}
\] (7.2)

The associated graph of SER vs. $E_s/N_0$ is shown in figure 7.3. In this case, the improvement is approximately 0.4-0.6 dB for SERs between $10^{-1}$ and $10^{-2}$. The improvement is less pronounced compared to before because the STM has less redundancy.

Fig. 7.3. Symbol Error Rate(SER) vs. $E_s/N_0$. No source transition probability matrix (solid line). Source transition probability matrix (dashed line)
7.1.2 Examples and Simulation Results for Trellis-coded CPFSK

Now consider the case of the coded CPFSK system. The overall convolutional encoder is the combination of a polynomial convolutional code over the ring of integers modulo \( M \) and a feedback-free (polynomial) CPE of the CPFSK modulation system. Hence, the overall encoder is therefore polynomial. For a given state \( x_k \), a polynomial encoder produces the associated incoming branch metrics that correspond to the same input symbol \( u_{k-1} \). This feature allows us to incorporate \( P(u_{k+1}|u_k) \) into all possible branch metrics in order to perform MAP decoding at the receiver. See Section 5.3 for the details. Now let us consider the following examples. These examples are taken from [33].

Example 7.1. Consider a rate 1/2 convolutional encoder over the ring of integers modulo 4. The generator matrix is \( G(D) = [2D + 1, 1] \). This noncatastrophic encoder is equivalent to the recursive systematic encoder \([1, 1/2D + 1]\) found in [15] to be the best code for quaternary CPFSK. These two codes have the same minimum squared Euclidean distance (SED) equal to 3.15, which indicates a 1.97 dB gain over that of minimum shift keying (MSK) at high SNR. The overall encoder whose rate is 1/4 \((1/2 \cdot 2/4)\), can be calculated as

\[
G_{all}(D) = G(D)G'(D) = \begin{bmatrix} 2D + 1 & 1 \\ D + 1 & D \end{bmatrix} \begin{bmatrix} 1 & 0 & 3 & 1 \\ 3D & D & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 3 & 1 \\ 3D & D & 1 & 0 \end{bmatrix}.
\]

The details of the derivation of \( G'(D) \) can be found in [15]. The block diagram and trellis diagram of the overall encoder for this case are shown in Figures 7.4 and 7.5, respectively.
Fig. 7.4. The block diagram of overall encoder for Example 7.1

Fig. 7.5. The trellis diagram of rate 1/2 encoder \([2D + 1, 1]\) with quaternary CPFSK, \(h=1/4\)
The total number of states of the encoder is four, which is matched with that of the source. Thus, one can use $P(u_{k+1} = i|u_k = j), i, j = 0, 1, 2, 3$ to compute all branch metrics. Each $P(u_{k+1}|u_k)$ is assigned to each branch at time interval $[k + 1, k + 2]$ in such a way that $u_k$ is the state number at time $k+1$ and $u_{k+1}$ is the input corresponding to each branch. Notice that at each state $s$ of time $k + 1$, all four incoming branches are associated with the same input $u_k = s$. (because the input corresponds to the next state)

**Example 7.2.** Consider a rate 1/2 convolutional encoder over the ring of integers modulo 4. Its generator matrix is $G(D) = [D + 1, 2D + 1]$. quaternary CPFSK is used. Again, this noncatastrophic encoder provides the same SED (4.09) as the best recursive systematic encoder [1, $2D + 1/D + 1$] found in [15]. The overall encoder also has rate of 1/4, as shown in Figure 7.6.

From the trellis diagram of the overall encoder in Figure 7.7, the total number of states of the overall encoder is eight, which is greater than that of the source, which is four. Thus, it is required to incorporate each $P(u_{k+1}|u_k)$ into two branch metrics. In this case, unlike in Example 7.1, we can not assign $u_k$ as the state number at time $k$. In general, $u_k$ is the input corresponding to the incoming branch for the state at time $k + 1$. For instance, at state $(0, 2)$ of time $k + 1$, all incoming branches correspond to the input $u_k = 0$. This implies that state $(0, 2)$ of time $k$ is associated with the input $u_{k-1} = 0$. Thus, we use $P(u_k = 0|u_{k-1} = 0)$ in both the branch from state $(0, 2)$ at time $k$ to $(0, 0)$ at time $k + 1$, and the branch from $(0, 0)$ at time $k$ to state $(0, 0)$ at time $k + 1$. 
Figure 7.8 shows the simulation results using the source transition matrix given by Eq. (7.2). In this realistic case, the gain improvement of the four state code is approximately 0.25 dB at a SER of $3 \times 10^{-2}$, and about 0.5 dB for the case of the eight state code. The performance of MSK without channel coding is also included for comparison purposes.

Next, let us consider the longer constraint length codes in order to see how much we gain by trading off with the complexity of a decoder. The polynomial ring convolutional codes $[2 + 3D + 2D^2, 1 + D]$ and $[2 + D, 1 + D + 2D^3]$ were implemented. These codes are equivalent to the best recursive code in [15] for the overall number of trellis states $m = 16$ and $m = 32$, respectively. Figure 7.9 shows the simulation results for the source transition matrix given by Eq. (7.2). The performance of MAP decoding is slightly better than that of ML decoding, about 0.5 dB at a SER of $3 \times 10^{-2}$ for both the 16 state code and the 32 state code. Moreover, the 32 state code gives approximately 0.25 dB gain at a SER of $10^{-2}$ above the 16 state code if MAP decoding is assumed to be used in both examples.
Fig. 7.7. The trellis diagram of rate 1/2 encoder $[D + 1, 2D + 1]$ with quaternary CPFSK, $h=1/4$
Fig. 7.8. Symbol error rate vs. $E_s/N_0$ for the source transition matrix (STM) given by Eq. (7.2). The parameter $m$ is the overall number of states in the trellis.

Fig. 7.9. Symbol error rate vs. $E_s/N_0$ for the source transition matrix (STM) given by Eq. (7.2). The parameter $m$ is the overall number of states in the trellis.
7.1.3 Examples and Simulation Results for Fading Channel

The performance of the proposed system over a Gaussian channel was investigated in the previous section. By using the noncatastrophic polynomial CE that is equivalent to the best recursive systematic encoder for a given constraint length found in [15], the MAP decoder shows a lower symbol error rate at a low SNR compared to the traditional maximum likelihood decoder. We continue our research in the case of a fading channel. Even though the optimal choice of a polynomial CE over the ring of integers modulo $M$, $\mathbb{Z}_M$, does not now depend on the minimum SED as in the case of a Gaussian channel, the same codes as before are first used. As previously shown in Eq. (5.36), the new branch metric for a fading channel is equal to

$$\lambda(u_{k+1}, u_k) = -\alpha_{k,1} \sum_{i=1}^{N} z_{k,i} \delta_{k,i}^m - \alpha_{k,2} \sum_{i=N+1}^{2N} z_{k,i} \delta_{k,i}^m - \frac{N_0}{2} \ln P(u_{k+1}|u_k), \quad (7.3)$$

where the parameter $\alpha_1$ and $\alpha_2$ represent the Rayleigh random variables, which are assumed to be fixed over the first and second symbol interval, respectively. Since it is difficult to determine the exact value of $\alpha_{k,1}$ and $\alpha_{k,2}$ at each interval, they are approximated by their mean $\bar{\alpha}$. Thus, the approximated version of the branch metric is

$$\lambda(u_{k+1}, u_k) = -\sum_{i=1}^{2N} z_{k,i} \delta_{k,i}^m - \frac{N_0}{2\bar{\alpha}} \ln P(u_{k+1}|u_k). \quad (7.4)$$

Figure 7.10 illustrate the performance of both the MAP and ML decoders over a Rayleigh fading channel. In these simulations, we use $\gamma = 10 \log(S/\sigma^2) = -67$ dB. Note that when $\gamma$ goes to $-\infty$, the channel goes from a Rician to a Rayleigh fading
Fig. 7.10. Symbol error rate vs. $\bar{\gamma}_b = (1 + \gamma) E_s / (\gamma N_0)$ for the source transition matrix (STM) given by Eq. (7.2) The parameter $m$ is the overall number of states in the trellis.

Fig. 7.11. Symbol error rate vs. $\bar{\gamma}_b = (1 + \gamma) E_s / (\gamma N_0)$ for the source transition matrix (STM) given by Eq. (7.2) The parameter $m$ is the overall number of states in the trellis.
channel. Notice that by applying MAP decoder instead of ML decoder, the improvement of the eight state system is about 0.3 dB (at $3 \times 10^{-2}$ SER) higher than that of the four state system. Now let us consider the longer constraint length ring convolutional encoders, $m = 16$ and $m = 32$. The same codes used in section 3.1.4 are considered. The simulation results are shown in Figure 7.11 for the STMs given by Eq.(7.2). The performance improvement of the 32 state code with MAP decoding is approximately 0.6 dB at a SER of $3 \times 10^{-2}$. Notice that at SNRs from 5 dB to 7 dB, the system performance of the 32 state code with MAP decoding is close to MSK over a Gaussian channel.

7.2 Mismatched Channel Simulation Result

Previously, we assumed that the noise power $N_0$ was known. This assumption may not be realistic. In this section, we discuss the simulation results when a channel mismatch occurs. Note that we use the rate $1/2$ 32-state code with the generator matrix $[2D^3 + D^2 + D + 3, D + 2]$, and we use the source transition matrix computed from the “Lena” image in single quantization mode. For a Gaussian channel, the mismatched channel simulation results are shown in Tables 7.1, 7.2, 7.3, and 7.4. In Table 7.1, we set the channel’s actual $E_s/N_0$ to be 1 dB, while we set the assumed $E_s/N_0$ from 1 to 5 dB. Note that the noise power, $N_0$, corresponding to the assumed $E_s/N_0$ is used in the branch metric for MAP decoding; please refer to Eq. (5.17). In this case, the mismatched results are similar to the matched result. These results all show some improvement compared to the result when ML decoding is used. The number of symbols used in this case is $5 \times 10^6$. In Table 7.2, the actual channel $E_s/N_0$ is set to be 2 dB, and the number of
Table 7.1. Symbol error rate result on mismatch channel. * means symbol error rate for match channel.

<table>
<thead>
<tr>
<th>Channel actual $E_s/N_0$ (dB)</th>
<th>Assumed $E_s/N_0$ (dB)</th>
<th>SER</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0.03254*</td>
</tr>
<tr>
<td>1</td>
<td>2</td>
<td>0.03327</td>
</tr>
<tr>
<td>1</td>
<td>3</td>
<td>0.03603</td>
</tr>
<tr>
<td>1</td>
<td>4</td>
<td>0.03896</td>
</tr>
<tr>
<td>1</td>
<td>5</td>
<td>0.04250</td>
</tr>
<tr>
<td>1</td>
<td>ML decoding</td>
<td>0.07434</td>
</tr>
</tbody>
</table>

Table 7.2. Symbol error rate results on a mismatched channel. * Symbol error rate for matched channel.

<table>
<thead>
<tr>
<th>Channel actual $E_s/N_0$ (dB)</th>
<th>Assumed $E_s/N_0$ (dB)</th>
<th>SER</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
<td>0.00791</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>0.00801*</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>0.00809</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>0.00881</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0.00944</td>
</tr>
<tr>
<td>2</td>
<td>ML decoding</td>
<td>0.01743</td>
</tr>
</tbody>
</table>

Table 7.3. Symbol error rate results on a mismatched channel. * Symbol error rate for a matched channel.

<table>
<thead>
<tr>
<th>Channel actual $E_s/N_0$ (dB)</th>
<th>Assumed $E_s/N_0$ (dB)</th>
<th>SER</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>1</td>
<td>0.00160</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>0.00136</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>0.00132*</td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>0.00137</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>0.00144</td>
</tr>
<tr>
<td>3</td>
<td>ML decoding</td>
<td>0.00273</td>
</tr>
</tbody>
</table>
symbols used in the simulation is $7 \times 10^6$. Again, the mismatched results still show some improvement compared to the ML decoding result. For Tables 7.3 and 7.4, we increase

<table>
<thead>
<tr>
<th>Channel actual $E_s/N_0$ (dB)</th>
<th>Assumed $E_s/N_0$ (dB)</th>
<th>SER</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>1</td>
<td>$2.344 \times 10^{-4}$</td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td>$1.780 \times 10^{-4}$</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>$1.495 \times 10^{-4}$</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>$1.444 \times 10^{-4}$*</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>$1.477 \times 10^{-4}$</td>
</tr>
<tr>
<td>4</td>
<td>ML decoding</td>
<td>$2.741 \times 10^{-4}$</td>
</tr>
</tbody>
</table>

Table 7.4. Symbol error rate results on a mismatched channel. *Symbol error rate for a matched channel.

the number of symbols to be $10^7$ and $2 \times 10^7$, respectively. The actual channel $E_s/N_0$ is set to be 3 and 4 dB, respectively. In Table 7.3, we still obtain a slight improvement compared to the ML decoding result. In Table 7.4, the improvements are obviously seen when the assumed $E_s/N_0$ ranges from 2 to 5 dB. However, when the assumed $E_s/N_0$ is 1 dB, the performance of MAP decoding is close to that of ML decoding.
Chapter 8

Image Transmission Results Part I: Performance Comparison Between ML and MAP Decoding

As mentioned earlier, the improvement in quality of the reconstructed image is our ultimate goal. One way to achieve this task is to use maximum a posteriori (MAP) decoding instead of conventional maximum likelihood (ML) decoding. In this chapter, we investigate the performance comparison in average peak signal-to-noise ratio (PSNR) between these two decoding methods. Note that the results in this chapter are obtained by using the external ring convolutional code designed for ML decoding. We consider both the AWGN and approximate Rayleigh fading channels. Please refer to Sections 5.2.2 and 5.4 on how to configure the fading channel. Throughout this chapter, the term “ML decoding” is used when the LFS and all of the HFS packets of an image are decoded using the ML criterion. The phrase “MAP decoding” is used when only the zerotree symbols in the HFS packets are decoded using the MAP criterion, and the rest of each packet is still ML decoded. Note that the nominal length of an HFS packet is about 500 bits. ML and MAP decoding are implemented by the Viterbi algorithm.

In this chapter, we use the 32 state polynomial ring convolutional code, $[1+D+2D^3, 2+D]$, combined with a continuous phase encoder (CPE) of 4-ary CPFSK, to be the overall channel encoder, $[1+3D+3D^2+2D^3, 2D+D^2, 1+3D^3, 1+D+2D^3]$. This noncatastrophic nonsystematic encoder is equivalent to the rational systematic encoder $[1, 2+D/1+D+2D^3]$ found in [15].
Presently, we assume that the packet length and the number of zerotrees in each packet are perfectly known by the channel decoder. In each packet, these important numbers are located in the overhead bits. In practice, one could protect these significant bits by using a high rate block encoder as an inner encoder. At the receiver, these bits are decoded first by the channel decoder and then following by the block decoder.

Our experiment is performed on both the “Lena” and “Barbara” images, respectively. The reason for using the “Barbara” image is because this image contains a larger number of zerotree symbols in the HFS packets compared to the “Lena” image. We would like to see how this difference affects the performance of MAP decoding. Moreover, we use both modes of coding the HFS as specified by MPEG-4: single quantization and multi-quantization modes. For each image, simulation results are obtained first for single quantization mode and then for multi-quantization mode. We obtain the performance comparison between these two modes. Note that in the single quantization mode, simulation results for the “Lena” image can also be found in [33].

8.1 The Simulation Results from using Single and Multi-quantization Modes for the “Lena” Image

The “Lena” image is used as our information source. This grayscale image has a size of 512 × 512 pixels. In our case, five decomposition levels are implemented, resulting in 16 subbands; one subband is for the LFS and 15 subbands for the HFS. As previously mentioned in Section 2.1.2.2, the wavelet coefficients of an image in single quantization mode are quantized only once with a multi-level quantizer. The bit allocation of the HFSs not only depends on the image, but also on the wavelet decomposition level.
The multi-quantization mode is similar to the original EZW method. In this mode, all wavelet coefficients (except those in the LFS) are compared to a given (initial) threshold for the first scanning pass. Then, this threshold is updated for the next scanning pass, typically decreased by a half of the current threshold. In each scanning pass, each wavelet coefficient is classified as one of four symbols, the so-called zerotree symbols. After each scan, the encoder “inverse quantizes” each non-zero value coefficient and subtracts this value from the original coefficient. Please refer to Section 2.1.2.2 for the details.

In this work, we use different threshold values for each HFS. Threshold values in the finer scale subbands are larger than those in the coarser scale subbands. By setting the threshold larger, we treat the coefficients as less significant. Moreover, the initial thresholds (for the first scanning pass) are selected in such a way that the compression ratio of an encoded image is approximately the same compared to that in single quantization mode.

8.1.1 The Results from using Single Quantization Mode

For a noiseless channel, the peak signal-to-noise ratio (PSNR) of the reconstructed image and its compression ratio depend on the type of image and the bit allocation. In single quantization mode, the wavelet coefficients of the “Lena” image are quantized to six bits for the LFS and 81 bits for the HFS: seven bits each for subbands 1-6, six bits each for subbands 7-9, five bits each for subbands 10-12 and two bits each for subband 13-15. By using this configuration, we obtain a PSNR for the “Lena” image of 31.83 dB at a compression ratio (CR) of 30.80:1 for a noiseless channel. Figures 8.1 and 8.2
Fig. 8.1. The original “Lena” image.

Fig. 8.2. The reconstructed “Lena” image for a noiseless channel. PSNR is 31.83 dB at a CR of 30.80:1.
show the original and the reconstructed “Lena” images, respectively. Observe that the contrast level of the reconstructed image is somewhat poorer than that of the original image. Furthermore, the smoothness in terms of both gray level and edge definition of the reconstructed image becomes slightly degraded. The simulation is performed first on a Gaussian channel and then a Rayleigh channel. For MAP decoding, the source transition matrix of the “Lena” image, previously given in Eq. (6.13) in Chapter 6, is used here.

The performance comparison between the ML and MAP decoding system is indicated in terms of PSNR (dB). Table 8.1 summarizes the simulation results on the Gaussian channel when strategies 1 (Top ten rows) and 2 (Bottom ten rows) are used. For strategy 1, we decode only the packets that pass a CRC, while for strategy 2, we decode all the packets. All simulation results in Table 8.1 are computed by averaging from 200 trials of the “Lena” images, which are obtained by repeating the simulation with different random number generator seeds.

For strategy 1, we observe that at $E_s/N_o = 3.5$ dB, the PSNR of the MAP decoding scheme is about 0.28 dB higher than that of the ML decoding scheme. This is equivalent to a reduction of 6.35% ($\frac{10^{2.3329} - 10^{2.3044}}{10^{2.3329}} \times 100\%$) in terms of noise power. For strategy 2, the simulation results are summarized in the bottom part of Table 8.1. In this case a significant improvement is obviously seen from $E_s/N_o = 3.5$ to 4.5 dB. We gain about 0.65 dB at $E_s/N_o = 4$ dB. The noise power is reduced about 13.96%. For both strategies, the PSNR improvement seems to be less evident when $E_s/N_o$ is relatively very small (between $E_s/N_o = 0 - 1$ dB). When $E_s/N_o$ is high (between $E_s/N_o = 5 - 6$ dB), both decoding schemes also yield similar performance. Representative reconstructed
<table>
<thead>
<tr>
<th>$E_s/N_o$ (dB)</th>
<th>PSNR (dB) with ML decoding</th>
<th>PSNR (dB) with MAP decoding</th>
<th>Improvement (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>6.79</td>
<td>6.82</td>
<td>0.03</td>
</tr>
<tr>
<td>1.0</td>
<td>8.00</td>
<td>8.10</td>
<td>0.10</td>
</tr>
<tr>
<td>2.0</td>
<td>11.92</td>
<td>11.99</td>
<td>0.07</td>
</tr>
<tr>
<td>3.0</td>
<td>18.93</td>
<td>19.19</td>
<td>0.26</td>
</tr>
<tr>
<td>3.5</td>
<td>23.04</td>
<td>23.33</td>
<td>0.29</td>
</tr>
<tr>
<td>4.0</td>
<td>26.03</td>
<td>26.21</td>
<td>0.18</td>
</tr>
<tr>
<td>4.5</td>
<td>28.75</td>
<td>28.86</td>
<td>0.11</td>
</tr>
<tr>
<td>5.0</td>
<td>30.06</td>
<td>30.08</td>
<td>0.02</td>
</tr>
<tr>
<td>5.5</td>
<td>30.74</td>
<td>30.85</td>
<td>0.11</td>
</tr>
<tr>
<td>6.0</td>
<td>31.15</td>
<td>31.15</td>
<td>0.00</td>
</tr>
<tr>
<td></td>
<td>0.0</td>
<td>6.74</td>
<td>6.76</td>
</tr>
<tr>
<td></td>
<td>1.0</td>
<td>7.79</td>
<td>7.80</td>
</tr>
<tr>
<td></td>
<td>2.0</td>
<td>10.14</td>
<td>10.57</td>
</tr>
<tr>
<td></td>
<td>3.0</td>
<td>15.65</td>
<td>16.15</td>
</tr>
<tr>
<td></td>
<td>3.5</td>
<td>19.76</td>
<td>20.39</td>
</tr>
<tr>
<td></td>
<td>4.0</td>
<td>23.97</td>
<td>24.63</td>
</tr>
<tr>
<td></td>
<td>4.5</td>
<td>28.07</td>
<td>28.63</td>
</tr>
<tr>
<td></td>
<td>5.0</td>
<td>30.50</td>
<td>30.88</td>
</tr>
<tr>
<td></td>
<td>5.5</td>
<td>31.46</td>
<td>31.58</td>
</tr>
<tr>
<td></td>
<td>6.0</td>
<td>31.83</td>
<td>31.83</td>
</tr>
</tbody>
</table>

Table 8.1. The average PSNR of the reconstructed “Lena” image using ML and MAP decoding for the AWGN channel. Strategy 1 (Top ten rows) and strategy 2 (Bottom ten rows) are used.

“Lena” images from using ML and MAP decoding are shown in Figures 8.3 and 8.4, respectively. It is worth noting that the choice of a pair of images depends not only on the average improvement but also on the average PSNR at particular $E_s/N_o$. In fact, we choose a pair of images of the same trial (thus, identical noise pattern) that have visual improvement close to the average improvement, and also have PSNR as close as possible $^1$ to the average PSNR. In this way, the images can be used to represent the

$^1$In general there are three criteria to be considered here: 1) the demonstrated pair of images should have the PSNR difference about the same as the average improvement; 2) Each of the pair should have PSNR about the average PSNR and 3) The visual improvement should be clearly pronounced by human eyes.
case that happen more frequently. Since strategy 1 is used, it is possible that some high frequency components of the image are lost. Thus, the sharpness of the reconstructed images becomes less evident. Note that the same trial is used for both images. The improvement is about 0.33 dB at $E_s/N_o = 3.5$ dB from using the MAP decoder. When $E_s/N_o = 3.5$ dB, the simulation result yields an average symbol error rate (SER) of about $10^{-3}$. Despite a marginal improvement in terms of PSNR (around the average), some significant visual improvement is noticeable around the region of her left eye and hair.

For strategy 2, since we decode all the packets, artifacts in the reconstructed image are expected. Reconstructed “Lena” images are shown in Figures 8.5 and 8.6. Again, the two images are obtained at the same trial with the different decoding processes, ML and MAP decoding, respectively. A moderate improvement of about 0.6 dB at $E_s/N_o = 4$ dB is obtained using the MAP decoder. At $E_s/N_o = 4$ dB, an average SER of about $3 \times 10^{-4}$ is measured. Notice the more severe defect at the middle of the edge of her hat (right above her left eye).

Table 8.2 shows the simulation results on a Rayleigh fading channel for strategies 1 and 2. For both strategies, slight improvements are obtained between $\gamma_b = 6.25$ and 8.75 dB. For strategy 1, the improvement is about 0.25 dB at $\gamma_b = 7.5$ dB, corresponding to an average SER of about $8 \times 10^{-4}$. For strategy 2, the gain is about 0.53 at $\gamma_b = 7.5$ dB. All of the simulation results in this section can also be found in [33]. Note that in [33], we focused on the single quantization mode in order to allow comparisons to the discrete channel results in [23]. In the discrete case, improvements in the 0.0 to 0.2 dB range were obtained. Therefore, by including the trellis-coded modulation in the design,
Fig. 8.3. The reconstructed “Lena” image using ML decoding with strategy 1. Note that PSNR = 26.22 dB at SNR = 3.5 dB. The Gaussian channel is used.

Fig. 8.4. The reconstructed “Lena” image using MAP decoding with strategy 1. Note that PSNR = 26.55 dB at SNR = 3.5 dB. The Gaussian channel is used.
Fig. 8.5. The reconstructed “Lena” image using ML decoding with strategy 2. Note that PSNR = 26.35 dB at SNR = 4 dB. The Gaussian channel is used.

Fig. 8.6. The reconstructed “Lena” image using MAP decoding with strategy 2. Note that PSNR = 26.96 dB at SNR = 4 dB. The Gaussian channel is used.
Table 8.2. The average PSNR of the reconstructed “Lena” image using ML and MAP decoding for the Rayleigh channel. Strategies 1 (Top eight rows) and 2 (Bottom eight rows) are used.

<table>
<thead>
<tr>
<th>$\gamma_b$ (dB)</th>
<th>PSNR (dB) with ML decoding</th>
<th>PSNR (dB) with MAP decoding</th>
<th>Improvement in dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.500</td>
<td>7.38</td>
<td>7.38</td>
<td>0.00</td>
</tr>
<tr>
<td>3.750</td>
<td>9.05</td>
<td>9.07</td>
<td>0.02</td>
</tr>
<tr>
<td>5.000</td>
<td>12.74</td>
<td>12.81</td>
<td>0.07</td>
</tr>
<tr>
<td>6.250</td>
<td>17.81</td>
<td>17.98</td>
<td>0.17</td>
</tr>
<tr>
<td>7.500</td>
<td>23.32</td>
<td>23.57</td>
<td>0.25</td>
</tr>
<tr>
<td>8.750</td>
<td>27.42</td>
<td>27.53</td>
<td>0.11</td>
</tr>
<tr>
<td>10.00</td>
<td>29.79</td>
<td>29.87</td>
<td>0.08</td>
</tr>
<tr>
<td>11.25</td>
<td>30.78</td>
<td>30.79</td>
<td>0.01</td>
</tr>
</tbody>
</table>

MAP decoding is more effective. It should be pointed out that the decoding is slightly different since this work uses sequence MAP (Viterbi algorithm) while the results in [23] used symbol MAP. Sequence MAP is a method to find the best single sequence given observed intervals, while the symbol MAP procedure finds the most likely state for each time $t$. Figures 8.7 and 8.8 show both ML and MAP results for the Rayleigh fading channel at $\gamma_b = 7.5$ dB. Strategy 1 is used, and the improvement in PSNR is about 0.27 dB for this trial. Visually, both of her eyes and her forehead become much clearer when the MAP decoder is used.

It is worth mentioning that there are some trials where the results from MAP decoding are worse than those from ML decoding. However, these events rarely happen and even when they do, the results from MAP decoding are not much less than those from
Fig. 8.7. The reconstructed “Lena” image using ML decoding with strategy 1. Note that PSNR= 25.08 dB at $\gamma_b = 7.5$ dB. The Rayleigh channel is used.

Fig. 8.8. The reconstructed “Lena” image using MAP decoding with strategy 1. Note that PSNR= 25.35 dB at $\gamma_b = 7.5$ dB. The Rayleigh channel is used.
ML decoding. For instance, we show the histogram of the improvement (PSNR) of 500 trials in Figure 8.9. About 22.8% of the time (114 out of 500) ML decoding outperforms MAP decoding. A mean of about 0.22 dB is obtained. Note that this mean is slightly different from the mean of the improvement (0.25 dB) shown in Table 8.2 because of the different number of trials. For strategy 2, the histogram of the improvement is shown in Figure 8.10. We can see that only about 15.2% of the time (76 out of 500) is ML better than MAP decoding. The average is about 0.49 dB. Note that the distribution functions of the improvement shown in Figure 8.9 and 8.10 are obtained for the same 500 trials of the “Lena” image for a Rayleigh fading channel at $\tilde{\gamma}_b = 7.5$ dB.

![Histogram of improvements](image-url)

**Fig. 8.9.** The histogram of the improvements (dB) of 500 trials of the “Lena” image. Strategy 1 and the Rayleigh channel at $\tilde{\gamma}_b = 7.5$ dB are used.

The results from the histograms obtained from both strategies suggest that most of the time, MAP decoding can improve image quality (measured in terms of PSNR) and
on some rare occasion, it can even slightly degrade the image. The relatively long tails on the righthand side of each histogram indicates that at times the PSNR of the image may be significantly improved. Even though using strategy 2 may increase improvement for MAP decoding, the average PSNR at this particular $\gamma_b$ is lower than that of strategy 1. Please refer to Table 8.2 for the summerized average PSNR. Moreover, since both strategies cause a different kind of visual noise, the choice may depend on the application.

8.1.2 The Results from using Multi-quantization Mode

In the multi-quantization mode, the initial thresholds are set to be $T/2$, $T$, $2T$, $4T$ and $8T$ for subbands 1-3, 4-6, 7-9, 10-12, and 13-15, respectively. The parameter $T$ is the maximum of all wavelet coefficient absolute values. The result is that the compressed image has a compression ratio of 30.87:1 and PSNR of 29.326 dB for a
noiseless channel. The reconstructed “Lena” image is shown in Figure 8.12. The contrast and detail definition of this image are somewhat poorer than those of Figure 8.2.

Note that in single quantization mode, we previously had a compression ratio of 30.80:1 and a PSNR of 31.833 dB. Also, note that the number of zerotree symbols in this mode is equal to 31,139 compared to only 13,976 for single quantization mode. The source transition matrix (STM) of the “Lena” image previously given in Eq. (6.16) is used here for MAP decoding. Notice that this STM is different from the STM used in Section 8.1.1.

Table 8.3 summarizes the average PSNRs for the Gaussian channel with both strategies. The average PSNRs are computed from the 200 “Lena” image trials. For strategy 1, the improvement for multi-quantization mode is higher than that for single quantization mode in the range of $E_s/N_o$ between 2 and 5 dB. Specifically, at $E_s/N_o = 3.5$, we obtain an improvement of about 0.768 dB, compared to only 0.285 dB for single quantization mode. As expected, we obtain better improvement in multi-quantization mode. Probably the reason is that a larger number of zerotree symbols is used in this mode.

However, for both ML and MAP decoding, the average PSNRs for multi-quantization mode at particular $E_s/N_o$s ranging from 2 to 6 dB are less (about 2 dB) than those for single quantization mode. The difference of 2 dB probably comes from the initial difference in PSNR of these two modes over a noiseless channel.

By using strategy 2 we obtain performance improvements of about 0.6 to 1.3 dB for MAP decoding at signal-to-noise ratios, $E_s/N_o$, in the range of 3 to 4 dB. In this $E_s/N_o$ range, not only improvements, but also average PSNRs for both ML and
Fig. 8.11. The original “Lena” image.

Fig. 8.12. The reconstructed “Lena” image for a noiseless channel. The PSNR is about 29.33 dB at a CR of 30.87:1.
<table>
<thead>
<tr>
<th>$E_s/N_0$ (dB)</th>
<th>PSNR (dB) with ML decoding</th>
<th>PSNR (dB) with MAP decoding</th>
<th>Improvement in dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>6.85</td>
<td>6.85</td>
<td>0.00</td>
</tr>
<tr>
<td>1.0</td>
<td>8.25</td>
<td>8.26</td>
<td>0.01</td>
</tr>
<tr>
<td>2.0</td>
<td>11.74</td>
<td>11.84</td>
<td>0.10</td>
</tr>
<tr>
<td>3.0</td>
<td>17.85</td>
<td>18.49</td>
<td>0.64</td>
</tr>
<tr>
<td>3.5</td>
<td>21.17</td>
<td>21.94</td>
<td>0.77</td>
</tr>
<tr>
<td>4.0</td>
<td>23.78</td>
<td>24.31</td>
<td>0.53</td>
</tr>
<tr>
<td>4.5</td>
<td>26.08</td>
<td>26.46</td>
<td>0.38</td>
</tr>
<tr>
<td>5.0</td>
<td>27.55</td>
<td>27.70</td>
<td>0.15</td>
</tr>
<tr>
<td>5.5</td>
<td>28.21</td>
<td>28.23</td>
<td>0.02</td>
</tr>
<tr>
<td>6.0</td>
<td>28.61</td>
<td>28.61</td>
<td>0.00</td>
</tr>
<tr>
<td>0.0</td>
<td>6.85</td>
<td>6.86</td>
<td>0.01</td>
</tr>
<tr>
<td>1.0</td>
<td>8.23</td>
<td>8.28</td>
<td>0.05</td>
</tr>
<tr>
<td>2.0</td>
<td>11.68</td>
<td>12.02</td>
<td>0.34</td>
</tr>
<tr>
<td>3.0</td>
<td>18.95</td>
<td>20.20</td>
<td>1.25</td>
</tr>
<tr>
<td>3.5</td>
<td>23.26</td>
<td>24.58</td>
<td>1.32</td>
</tr>
<tr>
<td>4.0</td>
<td>26.37</td>
<td>26.97</td>
<td>0.60</td>
</tr>
<tr>
<td>4.5</td>
<td>28.33</td>
<td>28.65</td>
<td>0.32</td>
</tr>
<tr>
<td>5.0</td>
<td>29.08</td>
<td>29.21</td>
<td>0.13</td>
</tr>
<tr>
<td>5.5</td>
<td>29.28</td>
<td>29.30</td>
<td>0.02</td>
</tr>
<tr>
<td>6.0</td>
<td>29.31</td>
<td>29.31</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 8.3. The average PSNR of the reconstructed “Lena” image using ML and MAP decoding for AWGN channel. Strategies 1 (Top ten rows) and 2 (Bottom ten rows) are used.

<table>
<thead>
<tr>
<th>$\gamma_b$ (dB)</th>
<th>PSNR (dB) with ML decoding</th>
<th>PSNR (dB) with MAP decoding</th>
<th>Improvement in dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.500</td>
<td>7.31</td>
<td>7.31</td>
<td>0.00</td>
</tr>
<tr>
<td>3.750</td>
<td>9.12</td>
<td>9.13</td>
<td>0.01</td>
</tr>
<tr>
<td>5.000</td>
<td>12.80</td>
<td>12.91</td>
<td>0.11</td>
</tr>
<tr>
<td>6.250</td>
<td>17.06</td>
<td>17.40</td>
<td>0.34</td>
</tr>
<tr>
<td>7.500</td>
<td>21.58</td>
<td>22.12</td>
<td>0.54</td>
</tr>
<tr>
<td>8.750</td>
<td>25.21</td>
<td>25.55</td>
<td>0.34</td>
</tr>
<tr>
<td>10.00</td>
<td>27.25</td>
<td>27.47</td>
<td>0.22</td>
</tr>
<tr>
<td>11.25</td>
<td>28.25</td>
<td>28.29</td>
<td>0.04</td>
</tr>
<tr>
<td>2.500</td>
<td>7.30</td>
<td>7.31</td>
<td>0.01</td>
</tr>
<tr>
<td>3.750</td>
<td>9.07</td>
<td>9.13</td>
<td>0.06</td>
</tr>
<tr>
<td>5.000</td>
<td>12.72</td>
<td>13.09</td>
<td>0.37</td>
</tr>
<tr>
<td>6.250</td>
<td>17.68</td>
<td>18.42</td>
<td>0.74</td>
</tr>
<tr>
<td>7.500</td>
<td>23.33</td>
<td>24.25</td>
<td>0.92</td>
</tr>
<tr>
<td>8.750</td>
<td>27.16</td>
<td>27.53</td>
<td>0.37</td>
</tr>
<tr>
<td>10.00</td>
<td>28.73</td>
<td>28.92</td>
<td>0.19</td>
</tr>
<tr>
<td>11.25</td>
<td>29.18</td>
<td>29.20</td>
<td>0.02</td>
</tr>
</tbody>
</table>

Table 8.4. The average PSNR of the reconstructed “Lena” image using ML and MAP decoding for a Rayleigh channel. Strategies 1 and 2 are used.
MAP decoding in multi-quantization mode, are noticeably larger than those in single-quantization mode. The simulation results suggest to us that multi-quantization mode is more suitable than single quantization mode for $E_s/N_o$ in the range of 2-4.5 dB. Another advantage of using multi-quantization mode is the capability to do progressive transmission.

The simulation results for a Rayleigh channel using strategies 1 and 2 are shown in Table 8.4. For strategy 1, we observe a significant improvement of about 0.3 to 0.5 dB for MAP decoding at $\bar{\gamma}_b$ in the range of 6.25 to 8.75 dB. Performance improvements for MAP decoding in multi-quantization mode are better than those in single quantization mode at $E_s/N_o$ in the range of 5 to 11.25 dB. However, the average PSNRs of both ML and MAP decoding in this mode are somewhat less than those in single quantization mode. The difference ranges from about 0.07 to 2.5 dB at $\bar{\gamma}_b$ from 2.5 to 11.25. For strategy 2, we obtain improvements of about 0.4 to 0.9 dB at $\bar{\gamma}_b$ in the range of 5 to 8.75 dB. Like the results for the Gaussian channel shown in Table 8.3, we obtain not only slightly larger improvements by using multi-quantization mode, but also moderately larger average PSNRs.

### 8.1.3 Mismatched channel

All the previous simulation results are obtained by assuming the knowledge of the noise power $N_o$. One may wonder how the performance in average PSNR of MAP decoding is affected when the noise power $N_o$ is unknown. In this section, we investigate the issue of a mismatched channel. In single quantization mode, Table 8.5 shows the simulation results for a Gaussian channel when the noise estimated is mismatched with
the value of $N_o$ and the symbol error rate ranges from $10^{-3}$ to $10^{-4}$ (corresponding to the $E_s/N_o$ approximately equal to $3 - 4$ dB). The assumed $E_s/N_o$ ranges from 2 to 4 dB. For comparison, the results from ML decoding are shown in the last column. The top three rows are for strategy 1 and the rest are for strategy 2. For strategy 1, we still obtain some improvement even though channel mismatched occurs. In fact, the improvements are not much different from those of the matched channel. For instance, when the actual $E_s/N_o$ is 3.5 dB, the improvement for using MAP decoding ranges from 0.252 to 0.285 dB. For strategy 2, improvements ranging from 0.528 to 0.634 dB are obtained at $E_s/N_o = 3.5$ dB.

Table 8.6 shows the simulation results for a mismatched fading channel. The mismatch in this case is due to the mismatch of $N_o$. The fading parameter $\alpha$ is still assumed to be known. This is a necessary condition for the branch metric $\lambda(sm_k) = -\sum_{i=1}^{N} z_{k,i}sm_{k,i} - \frac{N_0}{2\alpha} \ln P(u_k|u_{k-1}).$ (previously shown in Eq. (5.35)) to be accurate. Moreover, we relax the assumption of exactly knowing $\alpha$ by using only its mean ($\bar{\alpha}$).

In this case, slight improvements are still obtained. For example, at actual $\bar{\gamma}_b = 7.5$,
Table 8.6. The average PSNR of the reconstructed “Lena” image using MAP decoding with mismatched noise power $N_o$ for the Rayleigh fading channel. The asterisk indicates the matched case. Strategy 1 (Top three rows) and 2 (Bottom three rows) are used.

<table>
<thead>
<tr>
<th>Actual $\gamma b$ (dB)</th>
<th>Assumed $\gamma b$ (dB)</th>
<th>ML decoding (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.75</td>
<td>5.00</td>
</tr>
<tr>
<td>6.25</td>
<td>18.013</td>
<td>17.999</td>
</tr>
<tr>
<td>7.50</td>
<td>23.574</td>
<td>23.598</td>
</tr>
<tr>
<td>8.75</td>
<td>27.507</td>
<td>27.529</td>
</tr>
<tr>
<td>7.50</td>
<td>20.293</td>
<td>20.297</td>
</tr>
</tbody>
</table>

we have about 0.17-0.28 dB for strategy 1, and about 0.34-0.58 dB for strategy 2. For

Table 8.7. The average PSNR of the reconstructed “Lena” image using MAP decoding with mismatched noise power $N_o$ for the Gaussian channel. The asterisk indicates the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used.

<table>
<thead>
<tr>
<th>Actual $E_s/N_o$ (dB)</th>
<th>Assumed $E_s/N_o$ (dB)</th>
<th>ML decoding (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>3.0</td>
<td>18.571</td>
<td>18.534</td>
</tr>
</tbody>
</table>

multi-quantization mode, the mismatched simulation results are shown in Tables 8.7 and 8.8 for the Gaussian and fading channels, respectively. The results from using strategies 1 and 2 are shown in the same manner as shown in the previous tables. For the Gaussian channel, some improvements at an actual $E_s/N_o = 3.5$ dB are obtained, about 0.07-0.31 dB and 1.17-1.44 dB for strategies 1 and 2, respectively. For a fading channel at $\gamma b = 7.5$
Table 8.8. The average PSNR of the reconstructed “Lena” image using MAP decoding with mismatched noise power $N_o$ for the Rayleigh channel. The asterisk indicates the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used.

<table>
<thead>
<tr>
<th>Actual $\gamma_b$ (dB)</th>
<th>Assumed $\gamma_b$ (dB)</th>
<th>ML decoding (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>6.25</td>
<td>17.634</td>
<td>17.056</td>
</tr>
<tr>
<td>7.50</td>
<td>22.441</td>
<td>21.577</td>
</tr>
<tr>
<td>8.75</td>
<td>25.730</td>
<td>25.213</td>
</tr>
<tr>
<td>6.25</td>
<td>18.806</td>
<td>17.679</td>
</tr>
<tr>
<td>7.50</td>
<td>24.683</td>
<td>23.329</td>
</tr>
<tr>
<td>8.75</td>
<td>27.728</td>
<td>27.155</td>
</tr>
</tbody>
</table>

dB, we obtain improvements around 0.41-0.86 dB for strategy 1 and around 0.70-1.35 dB for strategy 2.

The cases, indicated by ($\cdot$)* above the numbers in all tables, refer to the situation when the channel is matched. Although the matched cases generally represent the optimal MAP receiver, this only implies minimum symbol error rate (SER)[37], it is not guaranteed to have maximum PSNR in these cases. In practice, when PSNR is of concern or becomes the goal of optimization, it is advisable to estimate $N_o$ to be of a higher value. This suggestion is deduced from the observation of Tables 8.5 to Table 8.8.

8.2 Simulation Results from using Single and Multi-quantization Modes for the “Barbara” Image

8.2.1 The Results from using Single Quantization Mode

In this section, we obtain simulation results for the “Barbara” image. For the single quantization mode, the wavelet coefficients of the “Barbara” image are quantized to six bits for the LFS and 81 bits for HFS: seven bits each for subbands 1-3, six bits each
for subbands 4-9, and four bits each for subbands 10-15. We obtain a PSNR of 27.52 dB at a compression ratio of 20:1 for a noiseless channel. Figures 8.13 and 8.14 show the original and reconstructed “Barbara” image for a noiseless channel, respectively. Note that the original “Barbara” image contains more high frequency components (higher detail in the spatial domain) than the “Lena” image, meaning that some wavelet coefficients in the higher frequency subbands have large magnitudes (possibly resulting in a larger number of bits per symbol). As a result, the number of zerotrees is then generally lower, and thus the compression ratio is reduced. At the same compression ratio, the reconstructed “Barbara” image therefore is expected to be of lower quality than that of the reconstructed “Lena” image. Figure 8.14 clearly has less detail in various areas such as the table cloth, her face, her pants, and her scarf.

In this mode, the source transition matrix of the “Barbara” image, previously given in Eq. (6.15) in Chapter 6, is used here for MAP decoding. Again, we obtain first the simulation results on a Gaussian channel and then a Rayleigh fading channel.

The top eight rows of Table 8.9 summarize the simulation results on a Gaussian channel for strategy 1, and the rest is for strategy 2. For both strategies, the use of MAP decoding improves the image quality. This is most evident at $E_s/N_o$ values in the 3 and 4 dB range. Note that the number of trials used for computing all average PSNRs at the particular $E_s/N_o$ is shown in parenthesis in the first column of the table. For more accurate results, this number increases when the noise power decreases. Representative reconstructed images are shown in Figures 8.15 and 8.16. Similar to the case of the “Lena” image, these reconstructed images become blurry when strategy 1 is used. For the same trial, a somewhat significant improvement in PSNR of
Fig. 8.13. The original “Barbara” image.

Fig. 8.14. The reconstructed “Barbara” image for a noiseless channel. PSNR is about 27.52 dB at a CR of 20:1.
<table>
<thead>
<tr>
<th>$E_s/N_o$ (dB) (no. of trials)</th>
<th>PSNR (dB) with ML decoding</th>
<th>PSNR (dB) with MAP decoding</th>
<th>Improvement in dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 (200)</td>
<td>8.10</td>
<td>8.10</td>
<td>0.00</td>
</tr>
<tr>
<td>2.0 (200)</td>
<td>11.71</td>
<td>11.75</td>
<td>0.04</td>
</tr>
<tr>
<td>3.0 (300)</td>
<td>18.38</td>
<td>18.62</td>
<td>0.24</td>
</tr>
<tr>
<td>3.5 (300)</td>
<td>21.97</td>
<td>22.23</td>
<td>0.26</td>
</tr>
<tr>
<td>4.0 (400)</td>
<td>24.40</td>
<td>24.58</td>
<td>0.18</td>
</tr>
<tr>
<td>4.5 (400)</td>
<td>25.81</td>
<td>25.90</td>
<td>0.09</td>
</tr>
<tr>
<td>5.0 (500)</td>
<td>26.61</td>
<td>26.64</td>
<td>0.03</td>
</tr>
<tr>
<td>5.5 (500)</td>
<td>27.01</td>
<td>27.02</td>
<td>0.01</td>
</tr>
<tr>
<td>1.0 (200)</td>
<td>7.78</td>
<td>7.79</td>
<td>0.01</td>
</tr>
<tr>
<td>2.0 (200)</td>
<td>10.33</td>
<td>10.40</td>
<td>0.07</td>
</tr>
<tr>
<td>3.0 (300)</td>
<td>15.89</td>
<td>16.12</td>
<td>0.23</td>
</tr>
<tr>
<td>3.5 (300)</td>
<td>19.71</td>
<td>20.01</td>
<td>0.30</td>
</tr>
<tr>
<td>4.0 (400)</td>
<td>23.23</td>
<td>23.43</td>
<td>0.20</td>
</tr>
<tr>
<td>4.5 (400)</td>
<td>25.64</td>
<td>25.73</td>
<td>0.09</td>
</tr>
<tr>
<td>5.0 (500)</td>
<td>26.90</td>
<td>26.96</td>
<td>0.06</td>
</tr>
<tr>
<td>5.5 (500)</td>
<td>27.34</td>
<td>27.34</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 8.9. The average PSNR of the reconstructed “Barbara” image using ML and MAP decoding for the AWGN channel. Strategies 1 and 2 are used.

approximately 0.27 dB is obtained at $E_s/N_o = 3.5$ dB for MAP decoding. The slight improvement is noticeable around Barbara’s left knee and hand. For strategy 2, the reconstructed “Barbara” images are shown in Figures 8.17 and 8.18. As expected, we can see artifacts in the reconstructed images since strategy 2 is used. For the same trial, the two figures illustrate an improvement in PSNR of about 0.31 dB at $E_s/N_o = 3.5$ dB from using MAP decoding. A slight visual improvement is obtained in this case. This can be seen around the area of Barbara’s right knee and elbow.

Table 8.10 concludes the simulation results for a flat Rayleigh fading channel for strategies 1 (Top seven rows) and 2 (Bottom seven rows). For strategy 1, at the particular low $E_s/N_o$ values from 2.5 to 3.75 dB, one can see that the performance of MAP decoding is effectively identical to that of ML decoding. However, the average
Fig. 8.15. The reconstructed “Barbara” image using ML decoding with strategy 1. Note that PSNR = 21.23 at SNR = 3.5 dB. The Gaussian channel is used.

Fig. 8.16. The reconstructed “Barbara” image using MAP decoding with strategy 1. Note that PSNR = 21.50 at SNR = 3.5 dB. The Gaussian channel is used.
Fig. 8.17. The reconstructed “Barbara” image using ML decoding with strategy 2. Note that PSNR = 20.28 at SNR = 3.5 dB. The Gaussian channel is used.

Fig. 8.18. The reconstructed “Barbara” image using MAP decoding with strategy 2. Note that PSNR = 20.59 at SNR = 3.5 dB. The Gaussian channel is used.
<table>
<thead>
<tr>
<th>$\gamma_b$ (dB)</th>
<th>PSNR (dB) with ML decoding</th>
<th>PSNR (dB) with MAP decoding</th>
<th>Improvement in dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.500 (200)</td>
<td>7.48</td>
<td>7.48</td>
<td>0.00</td>
</tr>
<tr>
<td>3.750 (300)</td>
<td>9.42</td>
<td>9.42</td>
<td>0.00</td>
</tr>
<tr>
<td>5.000 (300)</td>
<td>12.77</td>
<td>12.81</td>
<td>0.04</td>
</tr>
<tr>
<td>6.250 (400)</td>
<td>17.43</td>
<td>17.63</td>
<td>0.20</td>
</tr>
<tr>
<td>7.500 (400)</td>
<td>21.92</td>
<td>22.16</td>
<td>0.24</td>
</tr>
<tr>
<td>8.750 (500)</td>
<td>25.03</td>
<td>25.17</td>
<td>0.14</td>
</tr>
<tr>
<td>10.00 (500)</td>
<td>26.47</td>
<td>26.50</td>
<td>0.03</td>
</tr>
<tr>
<td>2.500 (200)</td>
<td>7.30</td>
<td>7.30</td>
<td>0.005</td>
</tr>
<tr>
<td>3.750 (300)</td>
<td>8.65</td>
<td>8.68</td>
<td>0.03</td>
</tr>
<tr>
<td>5.000 (300)</td>
<td>11.00</td>
<td>11.08</td>
<td>0.08</td>
</tr>
<tr>
<td>6.250 (400)</td>
<td>14.93</td>
<td>15.09</td>
<td>0.16</td>
</tr>
<tr>
<td>7.500 (400)</td>
<td>19.81</td>
<td>20.03</td>
<td>0.22</td>
</tr>
<tr>
<td>8.750 (500)</td>
<td>24.09</td>
<td>24.25</td>
<td>0.16</td>
</tr>
<tr>
<td>10.00 (500)</td>
<td>26.45</td>
<td>26.49</td>
<td>0.04</td>
</tr>
</tbody>
</table>

Table 8.10. The average PSNR of the reconstructed “Barbara” image using ML and MAP decoding for a Rayleigh channel. Strategies 1 and 2 are used.

PSNRs in this range are quite low. There is not much difference between performance of ML and MAP decoding, and in both cases the images are not useful. The improvement by using MAP decoding is noticeably seen in the range of $\gamma_b$ between 6.25 and 8.75 dB. The improvement varies between 0.14 to 0.24 dB. For strategy 2, a slight improvement, about 0.16 to 0.22 dB, can be seen in the same range of $\gamma_b$. Figures 8.19 and 8.20 show the reconstructed “Barbara” image using ML and MAP decoding for a Rayleigh fading channel, respectively. Strategy 1 is used in both cases. For this particular trial, we obtain a performance improvement of about 0.24 dB. We can observe the improvement around Barbara’s forehead.

Like the case of the “Lena” image, it is useful to see the histogram for both strategies. For strategy 1, the histogram is shown in Figure 8.21. In only 13.4 % of the trials (67 out of 500) is ML decoding better than MAP decoding. The average
Fig. 8.19. The reconstructed “Barbara” image using ML decoding with strategy 1. Note that PSNR = 21.69 at $\gamma_b = 7.5$ dB. The Rayleigh channel is used.

Fig. 8.20. The reconstructed “Barbara” image using MAP decoding with strategy 1. Note that PSNR = 21.93 at $\gamma_b = 7.5$ dB. The Rayleigh channel is used.
Fig. 8.21. The histogram of the improvements (dB) of 500 trials of the “Barbara" image. Strategy 1 and the Rayleigh channel at $\gamma_b = 7.5$ dB are used.

Fig. 8.22. The histogram of the improvements (dB) of 500 trials of the “Barbara" image. Strategy 2 and the Rayleigh channel at $\gamma_b = 7.5$ dB are used.
PSNR with MAP decoding and the average improvement are about 22.28 and 0.237 dB, respectively. Figure 8.22 shows the histogram when strategy 2 is used. The average improvement is about 0.208 dB. The average PSNR for MAP decoding is about 19.99 dB. About 22.2% (111 out of 500), the performance of ML decoding is slightly greater than MAP decoding. We may conclude that at this particular $\gamma_b = 7.5$ dB, strategy 1 is the method of choice since not only is the average PSNR higher, but the PSNR improvement of MAP decoding over ML decoding is also greater.

### 8.2.2 The Results from using Multi-quantization Mode

In this case, we use the initial thresholds $T, 2T, 4T, 8T$ for subbands 1-3, 4-6, 7-9, 10-12 and 13-15, respectively, where $T$ is the maximum of the wavelet coefficient absolute values for the “Barbara” image. The reconstructed Barbara image has a 19.96:1 compression ratio, which results in 26.81 dB PSNR for a noiseless channel. This compression ratio is chosen to be comparable to the case when the single quantization mode is used. Figure 8.24 shows the reconstructed “Barbara” image for a noiseless channel. The definition around Barbara’s upper left arm has been lost when compared to that from the single quantization mode (Figure 8.14).

The source transition matrix (STM) of the “Barbara” image is given by Eq.(6.16). For the “Barbara” image, the number of zerotree symbols is equal to 82,040 for the multi-quantization mode. Note that for single quantization mode, we only have 26,136 zerotree symbols. Assuming a Gaussian channel, we collect the simulation results for strategies 1 and 2 in Table 8.11. For strategy 1, the most significant improvement is 0.589 dB at $E_S/N_o = 3.5$ dB. This improvement is more than twice the improvement found in the
Fig. 8.23. The original “Barbara” image.

Fig. 8.24. The reconstructed “Barbara” image for a noiseless channel. PSNR is about 26.81 dB at a CR of 19.96:1.
single quantization mode (0.256). However, the average PSNRs (both ML and MAP decoding) for the single quantization mode are approximately 1 dB larger than those for multi-quantization mode. Therefore, one may want to use the single quantization mode with strategy 1 for the “Barbara” image.

For strategy 2, the significant improvement for MAP decoding is in the range of about 0.8-0.9 dB at $E_s/N_o = 3 - 3.5$ dB. At this range, the improvement is more than twice the improvement of single quantization mode. Moreover, the average PSNRs for both ML and MAP decoding are about 3 dB larger than those in single quantization mode. The simulation results suggest that one may want to use the multi-quantization mode with strategy 2 for the “Barbara” image when $E_s/N_o$ ranges from 2 to 4 dB. When $E_s/N_o$ is larger than 4 dB, the improvement for both modes seem to be approximately the same. One may notice that at high $E_s/N_o$ (more than 5 dB), the average PSNRs for both ML and MAP decoding in single quantization mode is slightly larger than those in multi-quantization mode. This is due to the difference of the PSNRs of both modes for a noiseless channel: 27.52 dB for single quantization and 26.809 dB for multi-quantization mode.

For the Rayleigh fading channel, we summarize the simulation results in Table 8.12 for strategies 1 and 2. For strategy 1, we find a performance improvement of about 0.3 to 0.5 dB at $\gamma_b$ in the range of 6.25 to 8.75 dB. The most significant improvement is 0.537 dB at $\gamma_b$ 7.5 dB. For strategy 2, the noticeable improvements, ranging from 0.2 to 0.7 dB, lie between $\gamma_b = 5$ and 8.75 dB. For comparison between the results from the two modes and the two strategies, we conclude that for the Rayleigh fading channel, the single quantization mode is suitable for strategy 1 (due to the greater average PSNRs),
Table 8.11. The average PSNR of the reconstructed “Barbara” image using ML and MAP decoding for AWGN channel. Strategies 1 (Top eight rows) and 2 (Bottom eight rows) are used.

<table>
<thead>
<tr>
<th>$E_b/N_0$ (dB) (no. of trials)</th>
<th>PSNR (dB) with ML decoding</th>
<th>PSNR (dB) with MAP decoding</th>
<th>Improvement in dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0 (200)</td>
<td>8.25</td>
<td>8.26</td>
<td>0.01</td>
</tr>
<tr>
<td>2.0 (200)</td>
<td>11.96</td>
<td>12.03</td>
<td>0.07</td>
</tr>
<tr>
<td>3.0 (300)</td>
<td>17.53</td>
<td>18.00</td>
<td>0.47</td>
</tr>
<tr>
<td>3.5 (300)</td>
<td>20.41</td>
<td>21.00</td>
<td>0.59</td>
</tr>
<tr>
<td>4.0 (400)</td>
<td>23.00</td>
<td>23.38</td>
<td>0.38</td>
</tr>
<tr>
<td>4.5 (400)</td>
<td>24.65</td>
<td>24.81</td>
<td>0.16</td>
</tr>
<tr>
<td>5.0 (500)</td>
<td>25.74</td>
<td>25.80</td>
<td>0.06</td>
</tr>
<tr>
<td>5.5 (500)</td>
<td>26.13</td>
<td>26.14</td>
<td>0.01</td>
</tr>
</tbody>
</table>

Table 8.12. The average PSNR of the reconstructed “Barbara” image using ML and MAP decoding for the Rayleigh channel. Strategies 1 (Top seven rows) and 2 (Bottom seven rows) are used.

<table>
<thead>
<tr>
<th>$\gamma_b$ (dB) (no. of trials)</th>
<th>PSNR (dB) with ML decoding</th>
<th>PSNR (dB) with MAP decoding</th>
<th>Improvement in dB</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.500 (200)</td>
<td>7.44</td>
<td>7.44</td>
<td>0.00</td>
</tr>
<tr>
<td>3.750 (300)</td>
<td>9.29</td>
<td>9.30</td>
<td>0.01</td>
</tr>
<tr>
<td>5.000 (300)</td>
<td>12.67</td>
<td>12.74</td>
<td>0.07</td>
</tr>
<tr>
<td>6.250 (400)</td>
<td>16.81</td>
<td>17.13</td>
<td>0.32</td>
</tr>
<tr>
<td>7.500 (400)</td>
<td>20.63</td>
<td>21.16</td>
<td>0.53</td>
</tr>
<tr>
<td>8.750 (500)</td>
<td>23.69</td>
<td>23.99</td>
<td>0.30</td>
</tr>
<tr>
<td>10.00 (500)</td>
<td>25.32</td>
<td>25.38</td>
<td>0.06</td>
</tr>
<tr>
<td>2.500 (200)</td>
<td>7.39</td>
<td>7.43</td>
<td>0.04</td>
</tr>
<tr>
<td>3.750 (300)</td>
<td>9.18</td>
<td>9.28</td>
<td>0.10</td>
</tr>
<tr>
<td>5.000 (300)</td>
<td>12.53</td>
<td>12.86</td>
<td>0.33</td>
</tr>
<tr>
<td>6.250 (400)</td>
<td>17.49</td>
<td>18.19</td>
<td>0.70</td>
</tr>
<tr>
<td>7.500 (400)</td>
<td>22.33</td>
<td>22.93</td>
<td>0.60</td>
</tr>
<tr>
<td>8.750 (500)</td>
<td>25.28</td>
<td>25.52</td>
<td>0.24</td>
</tr>
<tr>
<td>10.00 (500)</td>
<td>26.36</td>
<td>26.38</td>
<td>0.02</td>
</tr>
</tbody>
</table>
while the multi-quantization mode is the method of choice for strategy 2 (due to both
the larger improvement and average PSNRs). This conclusion is also the same for the
Gaussian channel.

8.2.3 Mismatched channel

As in the “Lena” case, one may wonder how a mismatched channel would affect
the performance of MAP decoding. For the single quantization mode, the results from
estimating the noise power, $N_0$, for a Gaussian channel are shown in Table 8.13. We find
that for strategy 1, improvements are obtained in the range of 0.137-0.314 dB at actual
$E_s/N_0 = 3.5$ dB. For strategy 2, we obtain slightly more improvements (compared to the
case of strategy 1) of about 0.25-0.37 dB at the same actual SNR. Table 8.14 shows the

<table>
<thead>
<tr>
<th>Actual $E_s/N_0$ (dB)</th>
<th>Assumed $E_s/N_0$ (dB)</th>
<th>ML decoding (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>3.0 (300)</td>
<td>18.614</td>
<td>18.650</td>
</tr>
<tr>
<td>3.5 (300)</td>
<td>22.245</td>
<td>22.236</td>
</tr>
<tr>
<td>4.0 (400)</td>
<td>23.431</td>
<td>23.448</td>
</tr>
</tbody>
</table>

Table 8.13. The average PSNR of the reconstructed “Barbara” image using MAP
decoding with mismatched noise power $N_0$ for a Gaussian channel. The asterisk indicates
the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used.

simulation results when mismatched noise power $N_0$ is considered with a fading channel.
Table 8.14. The average PSNR of the reconstructed “Barbara” image using MAP decoding with mismatched noise power $N_o$ for the Rayleigh fading channel. The asterisk indicates the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used.

<table>
<thead>
<tr>
<th>Actual $\gamma_b$ (dB)</th>
<th>Assumed $\gamma_b$ (dB)</th>
<th>ML decoding (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.75</td>
<td>5.00</td>
</tr>
<tr>
<td>6.25 (400)</td>
<td>17.672</td>
<td>17.644</td>
</tr>
<tr>
<td>7.50 (400)</td>
<td>22.174</td>
<td>22.168</td>
</tr>
<tr>
<td>8.75 (500)</td>
<td>24.189</td>
<td>25.177</td>
</tr>
<tr>
<td>6.25 (400)</td>
<td>15.122</td>
<td>15.107</td>
</tr>
</tbody>
</table>

We obtain improvements between 0.218-0.268 dB and 0.173-0.278 dB at actual average SNR $\gamma_b = 7.5$ dB for strategies 1 and 2, respectively.

For multi-quantization mode, we show the mismatched channel results in Tables 8.15 and 8.16 for the cases of the Gaussian and fading channels, respectively. In Table 8.15, improvements are obtained from 0.398 to 0.589 dB for strategy 1, and from 0.62 to 0.83 dB for strategy 2 at actual $E_s/N_o = 3.5$ dB. For the Rayleigh channel, we obtain improvements from 0.287 to 0.583 dB at actual $\gamma_b = 7.5$ dB when strategy 1 is used. Also, improvements of about 0.35 to 0.78 dB are achieved at the same actual $\gamma_b$ when strategy 2 is used.
Table 8.15. The average PSNR of the reconstructed “Barbara” image using MAP decoding with mismatched noise power $N_o$ for the Gaussian channel. The asterisk indicates the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used.

<table>
<thead>
<tr>
<th>Actual $E_s/N_o$ (dB)</th>
<th>Assumed $E_s/N_o$ (dB)</th>
<th>ML decoding (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1.0</td>
<td>2.0</td>
</tr>
<tr>
<td>3.0 (300)</td>
<td>17.977</td>
<td>17.981</td>
</tr>
<tr>
<td>4.0 (400)</td>
<td>23.281</td>
<td>23.300</td>
</tr>
<tr>
<td></td>
<td>22.681</td>
<td>22.755</td>
</tr>
</tbody>
</table>

Table 8.16. The average PSNR of the reconstructed “Barbara” image using MAP decoding with mismatched noise power $N_o$ for the Rayleigh channel. The asterisk indicates the matched case. Strategies 1 (Top three rows) and 2 (Bottom three rows) are used.

<table>
<thead>
<tr>
<th>Actual $\gamma_b$ (dB)</th>
<th>Assumed $\gamma_b$ (dB)</th>
<th>ML decoding (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.75</td>
<td>5.00</td>
</tr>
<tr>
<td>6.25 (400)</td>
<td>17.205</td>
<td>17.158</td>
</tr>
<tr>
<td>8.75 (500)</td>
<td>23.958</td>
<td>23.979</td>
</tr>
<tr>
<td>6.25 (400)</td>
<td>18.339</td>
<td>18.252</td>
</tr>
<tr>
<td>7.50 (400)</td>
<td>23.106</td>
<td>23.062</td>
</tr>
<tr>
<td>8.75 (500)</td>
<td>25.541</td>
<td>25.566</td>
</tr>
</tbody>
</table>
Chapter 9

Image Transmission Results Part II: Performance Comparison Between ML and MAP codes

In this chapter, we are interested in comparing the performance of the best non-systematic ML and the suboptimal MAP codes using MAP decoding. These latter codes were found in Section 6.4. Unlike the best ML codes, the suboptimal MAP codes were designed by using the minimum squared Euclidean distance criterion with the addition of a source transition matrix (STM). In previous work [34], we presented the suboptimal 32-state MAP code, which is designed by using the STM of the “Lena” image in multi-quantization mode.

<table>
<thead>
<tr>
<th>$E_s/N_o$ (dB)</th>
<th>PSNR (dB) MAP (YTE)</th>
<th>PSNR (dB) MAP (SC)</th>
<th>$\Delta$ PSNR (dB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0</td>
<td>8.26</td>
<td>8.49</td>
<td>0.23</td>
</tr>
<tr>
<td>2.0</td>
<td>11.83</td>
<td>13.42</td>
<td>1.59</td>
</tr>
<tr>
<td>3.0</td>
<td>18.56</td>
<td>20.35</td>
<td>1.79</td>
</tr>
<tr>
<td>3.5</td>
<td>22.14</td>
<td>23.18</td>
<td>1.04</td>
</tr>
<tr>
<td>4.0</td>
<td>24.50</td>
<td>25.10</td>
<td>0.60</td>
</tr>
<tr>
<td>4.5</td>
<td>26.68</td>
<td>26.73</td>
<td>0.05</td>
</tr>
<tr>
<td>5.0</td>
<td>27.82</td>
<td>27.82</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Table 9.1. The comparison of reconstructed image average PSNR between our suboptimal code (SC) and Yang and Taylor’s equivalent code (YTE) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. AWGN channel.
In Table 9.1, the performance comparison in term of PSNR, between this code and the equivalent Yang and Taylor’s code found in [15], is obtained [34]. Note that since our best non-systematic ML code has larger $d^2_{free}$ (see Table 6.2) compared to the equivalent Yang and Taylor’s code. Thus, the performance of our code in term of SER is slightly better. These results are based on the use of Strategy 1, meaning that all HFS packets that pass an outer cyclic redundancy check (CRC) are decoded. Note that this check requires only 4 bits/packet so it is fairly weak. Consequently, undetected errors can be noticeable at high symbol error rates. In the range of symbol error rates from $10^{-2}$ to $10^{-3}$, corresponding to $E_s/N_o$ from 2 to 3 dB, the improvement is from 1.6 to 1.8 dB. One reason for this difference is that by constraining the code search to systematic codes, the better noncatastrophic ring convolutional codes are not necessarily found. This result is consistent with Table I in [28], which is for rate one-half binary codes. Also, converting from systematic form to polynomial form can increase $d^2_{free}$.

We continue our work by considering the performance improvement obtained in the case of 64-state codes designed especially for the “Lena” and the “Barbara” images. Again, we first discuss the simulation results obtained from using single quantization mode, and then those from using multi-quantization mode. Moreover, simulations are performed for both Gaussian and Rayleigh channels.
9.1 Simulation Results from Single Quantization Mode

9.1.1 Simulation Results for the “Lena” Image

The simulation results for the “Lena” image in single quantization mode are presented and discussed. For a Gaussian channel, we illustrate the performance comparison between the best ML code (ML-MAP), and the suboptimal MAP code (MAP-MAP) using MAP decoding for strategies 1 and 2 in Tables 9.2 and 9.3, respectively. Note that “ML-MAP” denotes the best “ML” code using “MAP” decoding, and “MAP-MAP” is for the suboptimal “MAP” code with “MAP” decoding. We also include performance using the best ML code with ML decoding (ML-ML). The best ML code, taken from Table 6.2 is \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) with \(d^2_{free} = 7.45\). In this case, the suboptimal MAP code is \([D^3 + 3D^2 + 2D + 3, 2D^2 + 3D + 2]\), obtained in Section 6.4. The \(d^2_{free}\) of this code is 7.09. Note that \(d^2_{free}\) of this code is lower than that of the

<table>
<thead>
<tr>
<th>(E_s/N_o) (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>(\Delta) PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>(\Delta) PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>13.327</td>
<td>13.337</td>
<td>14.167</td>
<td>0.550</td>
<td>0.630</td>
</tr>
<tr>
<td>3.0</td>
<td>21.714</td>
<td>21.988</td>
<td>22.624</td>
<td>0.636</td>
<td>0.910</td>
</tr>
<tr>
<td>3.5</td>
<td>25.458</td>
<td>25.692</td>
<td>26.123</td>
<td>0.431</td>
<td>0.665</td>
</tr>
<tr>
<td>4.0</td>
<td>27.650</td>
<td>27.773</td>
<td>28.223</td>
<td>0.450</td>
<td>0.573</td>
</tr>
</tbody>
</table>

Table 9.2. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) \([D^3 + 3D^2 + 2D + 3, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. AWGN channel.
best ML code (7.45). Thus, at high $E_s/N_o$, the asymptotic performance of this code is (very slightly) less than that of the best ML code. Fortunately, this is less important than the performance at low $E_s/N_o$ ($2 - 4$ dB).

In Table 9.2, we obtain a performance improvement of about 0.43 to 0.64 dB (PSNR) by using the MAP code with MAP decoding (MAP-MAP) at $E_s/N_o$ in the range of 2 to 4 dB. Note that $E_s/N_o = 3$ dB corresponds to the symbol error rate (SER) about $10^{-3}$. At this point, we gain about 0.64 dB in terms of an average PSNR. If we compare between ML-ML and MAP-MAP, we obtain the gain of 0.91 dB. In the case of strategy 2, shown in Table 9.3, the improvement is about 0.42 to 1.06 at $E_s/N_o$ in the range of 2 to 4 dB. At $E_s/N_o = 3$ dB, we obtain a significant improvement of about 1.06 dB. Note that the average PSNRs of all cases (ML-ML, ML-MAP, and MAP-MAP)

<table>
<thead>
<tr>
<th>$E_s/N_o$ (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>$\Delta$ PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>$\Delta$ PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>11.536</td>
<td>11.868</td>
<td>12.291</td>
<td>0.423</td>
<td>0.755</td>
</tr>
<tr>
<td>3.0</td>
<td>18.768</td>
<td>19.382</td>
<td>20.443</td>
<td>1.061</td>
<td>1.675</td>
</tr>
<tr>
<td>3.5</td>
<td>23.894</td>
<td>24.539</td>
<td>25.421</td>
<td>0.882</td>
<td>1.527</td>
</tr>
<tr>
<td>4.0</td>
<td>27.820</td>
<td>28.339</td>
<td>29.202</td>
<td>0.863</td>
<td>1.382</td>
</tr>
</tbody>
</table>

Table 9.3. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) $[D^3 + 3D^2 + 2D + 3, 2D^2 + 3D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. AWGN channel.

Using strategy 1 are better than those using strategy 2 at $E_s/N_o$ in the range of 2 to 3.5 dB (corresponding to a SER from $10^{-2}$ to $10^{-4}$). At $E_s/N_o = 4$ dB (an SER of
about $10^{-4}$), the results from using strategy 2 are better than those from using strategy 1. These results suggest that the use of strategy 2 yields higher PSNR in the region of a SER less than $10^{-4}$.

<table>
<thead>
<tr>
<th>$\tilde{\gamma}_b$ (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>$\Delta$ PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>$\Delta$ PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.00</td>
<td>15.647</td>
<td>15.671</td>
<td>15.812</td>
<td>0.141</td>
<td>0.165</td>
</tr>
<tr>
<td>6.25</td>
<td>21.863</td>
<td>22.095</td>
<td>22.236</td>
<td>0.141</td>
<td>0.373</td>
</tr>
<tr>
<td>7.50</td>
<td>26.349</td>
<td>26.524</td>
<td>26.891</td>
<td>0.367</td>
<td>0.542</td>
</tr>
<tr>
<td>8.75</td>
<td>28.835</td>
<td>28.885</td>
<td>29.198</td>
<td>0.313</td>
<td>0.363</td>
</tr>
</tbody>
</table>

Table 9.4. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) $[D^3 + 3D^2 + 2D + 3, 2D^2 + 3D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. Rayleigh channel.

Tables 9.4 and 9.5 illustrate the simulation results for a flat Rayleigh channel using strategies 1 and 2, respectively. For strategy 1, a slight performance improvement from using MAP-MAP is obtained, about 0.14 to 0.37 dB at $\tilde{\gamma}_b$ in the range of 5 to 8.75 dB. Note that at $\tilde{\gamma}_b = 6.25$ dB, corresponding to a SER about $10^{-3}$, an improvement of about 0.14 dB is found. The greatest improvement of about 0.37 dB is obtained at $E_s/N_0 = 7.5$ dB (corresponding to SER of about $10^{-4}$). In Table 9.5, one can observe that the improvement ranges from about 0.2 to 0.6 dB at $\tilde{\gamma}_b$ ranging from 5 to 7.5 dB. Note that at $\tilde{\gamma}_b = 6.25$ dB, we obtain a gain of about 0.49 dB.
Table 9.5. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) $[D^3 + 3D^2 + 2D + 3, 2D^2 + 3D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. Rayleigh channel.

<table>
<thead>
<tr>
<th>$\gamma_b$ (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>$\Delta$ PSNR (dB) ML-ML vs. MAP-MAP</th>
<th>$\Delta$ PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.00</td>
<td>12.860</td>
<td>13.215</td>
<td>13.403</td>
<td>0.188</td>
<td>0.543</td>
</tr>
<tr>
<td>6.25</td>
<td>18.625</td>
<td>19.137</td>
<td>19.625</td>
<td>0.488</td>
<td>1.000</td>
</tr>
<tr>
<td>7.50</td>
<td>25.006</td>
<td>25.697</td>
<td>26.342</td>
<td>0.645</td>
<td>1.336</td>
</tr>
<tr>
<td>8.75</td>
<td>29.755</td>
<td>29.929</td>
<td>30.416</td>
<td>0.487</td>
<td>0.661</td>
</tr>
</tbody>
</table>

9.1.2 Simulation Results for the “Barbara” Image

In this section, we obtain simulation results for the “Barbara” image. The average PSNRs, using ML-ML, ML-MAP, and MAP-MAP over the AWGN channel, are summarized in Tables 9.6 (strategy 1) and 9.7 (strategy 2). In this case, the best MAP code is $[D^3 + 3D^2 + 2D + 1, 2D^2 + 3D + 2]$, obtained in Section 6.4. The $d^2_{free}$ of this code is 7.09. Note that the improvement ($E_s/N_o$) is about 0.16 dB using the MAP code at $E_s/N_o = 3$ dB. The total improvement of about 0.55 dB is obtained when using MAP-MAP, instead of ML-ML. Notice that this improvement is about the same as the improvement in the case of using the “Lena” image (0.5 dB).

In Table 9.6, we obtain a performance improvement of about 0.2 to 0.9 dB by using the MAP code at $E_s/N_o$ in the range of 2 to 4 dB. Gains of about 0.2 to 1 dB are found by using MAP-MAP instead of ML-ML. At $E_s/N_o = 3$ dB, the improvement due to the MAP code is about 0.4 dB, and the total improvement (ML-ML vs. MAP-MAP) is about 0.6 dB. In the case of strategy 2 shown in Table 9.7, the improvement due to the MAP code is about 0.4 to 0.8 at $E_s/N_o$ in the range of 2 to 4 dB. The total
Table 9.6. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) \([D^3 + 3D^2 + 2D + 1, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. AWGN channel.

<table>
<thead>
<tr>
<th>(E_s/N_o) (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>(\Delta) PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>(\Delta) PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>13.353</td>
<td>13.430</td>
<td>14.359</td>
<td>0.929</td>
<td>1.006</td>
</tr>
<tr>
<td>3.0</td>
<td>20.845</td>
<td>21.020</td>
<td>21.419</td>
<td>0.399</td>
<td>0.574</td>
</tr>
<tr>
<td>3.5</td>
<td>23.706</td>
<td>23.816</td>
<td>24.268</td>
<td>0.452</td>
<td>0.562</td>
</tr>
<tr>
<td>4.0</td>
<td>25.420</td>
<td>25.469</td>
<td>25.655</td>
<td>0.186</td>
<td>0.235</td>
</tr>
</tbody>
</table>

Improvement is about 0.4 to 1 dB in the same \(E_s/N_o\) range. In both strategies, the most evident improvements are found at \(E_s/N_o = 3.5\) dB, which is corresponding to the SER of about \(10^{-4}\). Like the case of using the “Lena” image, the average PSNRs at \(E_s/N_o = 4\) dB, obtained by using strategy 2, are higher than those obtained by using strategy 1.

Tables 9.8 and 9.9 show the simulation results for a Rayleigh channel using strategies 1 and 2, respectively. For Strategy 1, a performance improvement of about 0.4 to 0.7 dB from using the MAP code is obtained at \(\gamma_b\) in the range of 5 to 8.75 dB. Note that a SER of about \(10^{-3}\) corresponds to \(\gamma_b\) about 6.25 dB. In Table 9.9, one can observe that the improvement due to the MAP code ranges from about 0.4 to 0.9 dB. Unlike the case of using the “Lena” image, the greatest improvements are found at \(E_s/N_o = 6.25\) for both strategies.

In summary, the 64-state MAP codes, designed for both the “Lena” and the “Barbara” images in single quantization mode, can be used with MAP decoding to improve the reconstructed image quality. It is worth mentioning that even though these
Table 9.7. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) $[D^3 + 3D^2 + 2D + 1, 2D^2 + 3D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. AWGN channel.

<table>
<thead>
<tr>
<th>$E_b/N_0$ (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>Δ PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>Δ PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>11.707</td>
<td>11.802</td>
<td>12.443</td>
<td>0.641</td>
<td>0.736</td>
</tr>
<tr>
<td>3.0</td>
<td>18.964</td>
<td>19.133</td>
<td>19.898</td>
<td>0.765</td>
<td>0.934</td>
</tr>
<tr>
<td>3.5</td>
<td>22.855</td>
<td>23.062</td>
<td>23.888</td>
<td>0.826</td>
<td>1.033</td>
</tr>
<tr>
<td>4.0</td>
<td>25.668</td>
<td>25.722</td>
<td>26.110</td>
<td>0.388</td>
<td>0.442</td>
</tr>
</tbody>
</table>

Table 9.8. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) $[D^3 + 3D^2 + 2D + 1, 2D^2 + 3D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. Rayleigh channel.

<table>
<thead>
<tr>
<th>$\gamma_b$ (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>Δ PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>Δ PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.00</td>
<td>14.651</td>
<td>14.748</td>
<td>15.369</td>
<td>0.621</td>
<td>0.718</td>
</tr>
<tr>
<td>6.25</td>
<td>19.997</td>
<td>20.151</td>
<td>20.893</td>
<td>0.742</td>
<td>0.896</td>
</tr>
<tr>
<td>7.50</td>
<td>24.307</td>
<td>24.406</td>
<td>24.812</td>
<td>0.406</td>
<td>0.505</td>
</tr>
<tr>
<td>8.75</td>
<td>25.983</td>
<td>26.027</td>
<td>26.465</td>
<td>0.438</td>
<td>0.482</td>
</tr>
</tbody>
</table>

Table 9.9. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) $[D^3 + 3D^2 + 2D + 1, 2D^2 + 3D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. Rayleigh channel.

<table>
<thead>
<tr>
<th>$\gamma_b$ (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>Δ PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>Δ PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.00</td>
<td>12.641</td>
<td>12.789</td>
<td>13.323</td>
<td>0.534</td>
<td>0.682</td>
</tr>
<tr>
<td>6.25</td>
<td>17.922</td>
<td>18.134</td>
<td>19.030</td>
<td>0.896</td>
<td>1.108</td>
</tr>
<tr>
<td>7.50</td>
<td>23.839</td>
<td>24.010</td>
<td>24.491</td>
<td>0.481</td>
<td>0.652</td>
</tr>
<tr>
<td>8.75</td>
<td>26.489</td>
<td>26.557</td>
<td>26.930</td>
<td>0.373</td>
<td>0.441</td>
</tr>
</tbody>
</table>
MAP codes are designed based on Gaussian channel by specifying a value for $N_o$, the improvement of using these MAP codes with MAP decoding is also found at other neighboring values of $N_o$. Also, the gain is found for the Rayleigh channel.

9.2 Simulation Results from Multi-Quantization Mode

9.2.1 Simulation Results for the “Lena” Image

The simulation results for the “Lena” image in multi-quantization mode are presented and discussed in this section. For a Gaussian channel, we show a performance comparison between the best ML code (ML-MAP), and the suboptimal MAP code (MAP-MAP) using MAP decoding for strategies 1 and 2 in Tables 9.10 and 9.11, respectively. We also include average PSNRs obtained by using the best ML code with ML decoding (ML-ML). The best ML code $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ with $d_{\text{free}}^2 = 7.45$, taken from Table 6.2, is used. In this case, the suboptimal MAP code is $[D^3 + 2D^2 + D + 3, 2D^2 + 3D + 2]$ obtained in Section 6.4. The $d_{\text{free}}^2$ of this code is 6.82. Note that the $d_{\text{free}}^2$ of this code is even less than that of the suboptimal MAP code for the “Lena” image in single quantization mode (7.09). Even so, the asymptotic performance of this code is only slightly less than that of the best ML code at high $E_s/N_o$ (more than 7 dB). The gain ($E_s/N_o$) due to the MAP code is about 0.16 dB (see Section 6.4) from using MAP code with MAP decoding at a SER of about $10^{-3}$.

In Table 9.10, we obtain a significant performance improvement of about 0.5 to 1.2 dB by using the MAP code with MAP decoding (MAP-MAP) at $E_s/N_o$ in the range of 2 to 4 dB. At this point, we gain about 0.55 dB in term of average PSNR. In the
### Table 9.10. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) \([D^3 + 2D^2 + D + 3, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. AWGN channel.

<table>
<thead>
<tr>
<th>(E_s/N_0) (dB)</th>
<th>PSNR (dB) ML-ML</th>
<th>PSNR (dB) ML-MAP</th>
<th>PSNR (dB) MAP-MAP</th>
<th>(\Delta) PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>(\Delta) PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>13.754</td>
<td>13.927</td>
<td>15.100</td>
<td>1.173</td>
<td>1.346</td>
</tr>
<tr>
<td>3.0</td>
<td>20.647</td>
<td>21.118</td>
<td>21.666</td>
<td>0.548</td>
<td>1.019</td>
</tr>
<tr>
<td>3.5</td>
<td>23.584</td>
<td>24.031</td>
<td>24.878</td>
<td>0.847</td>
<td>1.294</td>
</tr>
<tr>
<td>4.0</td>
<td>25.485</td>
<td>25.699</td>
<td>26.456</td>
<td>0.757</td>
<td>0.971</td>
</tr>
</tbody>
</table>

### Table 9.11. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) \([D^3 + 2D^2 + D + 3, 2D^2 + 3D + 2]\) and the best ML code (ML-MAP) \([D^3 + D^2 + 2D + 1, D^2 + 2D + 2]\) using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. AWGN channel.

<table>
<thead>
<tr>
<th>(E_s/N_0) (dB)</th>
<th>PSNR (dB) ML-ML</th>
<th>PSNR (dB) ML-MAP</th>
<th>PSNR (dB) MAP-MAP</th>
<th>(\Delta) PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>(\Delta) PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>14.045</td>
<td>14.495</td>
<td>15.799</td>
<td>1.304</td>
<td>1.754</td>
</tr>
<tr>
<td>3.0</td>
<td>23.265</td>
<td>24.091</td>
<td>24.306</td>
<td>0.251</td>
<td>1.041</td>
</tr>
<tr>
<td>3.5</td>
<td>26.709</td>
<td>27.340</td>
<td>27.785</td>
<td>0.445</td>
<td>1.076</td>
</tr>
<tr>
<td>4.0</td>
<td>28.428</td>
<td>28.695</td>
<td>28.782</td>
<td>0.087</td>
<td>0.354</td>
</tr>
</tbody>
</table>
case of strategy 2 shown in Table 9.11, the improvement is about 0.3 to 1.3 at $E_s/N_o$ in the range of 2 to 3.5 dB. Notice that in the range of 2 to 4 dB ($E_s/N_o$), the average PSNR performance using strategy 2 is better that that using strategy 1. This may be

<table>
<thead>
<tr>
<th>$\hat{\gamma}_b$ (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>$\Delta$ PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>$\Delta$ PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.00</td>
<td>14.820</td>
<td>15.016</td>
<td>16.043</td>
<td>1.027</td>
<td>1.223</td>
</tr>
<tr>
<td>6.25</td>
<td>19.711</td>
<td>20.016</td>
<td>20.912</td>
<td>0.896</td>
<td>1.201</td>
</tr>
<tr>
<td>7.50</td>
<td>23.571</td>
<td>24.004</td>
<td>24.937</td>
<td>0.935</td>
<td>1.366</td>
</tr>
<tr>
<td>8.75</td>
<td>26.208</td>
<td>26.280</td>
<td>27.019</td>
<td>0.739</td>
<td>0.811</td>
</tr>
</tbody>
</table>

Table 9.12. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) $[D^3 + 2D^2 + D + 3, 2D^2 + 3D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. Rayleigh channel.

because there are many packets that do not pass a CRC. To alleviate this problem, one may consider increasing the number of bits used in CRC.

Table 9.12 and 9.13 illustrate the simulation results for a Rayleigh channel using strategies 1 and 2, respectively. For strategy 1, a significant performance improvement for using the MAP code with the MAP decoding (MAP-MAP) is obtained about 0.7 to 1.0 dB at $\hat{\gamma}_b$ in the range of 5 to 8.75 dB. Note that at $\hat{\gamma}_b$ about 6.25 dB, corresponding to SER about $10^{-3}$, the significant improvement of about 0.9 dB is obtained. In Table 9.13, one can observe that the improvement ranges from about 0.3 to 1.1 dB at $\hat{\gamma}_b$ from 5 to 7.5 dB. Note that at $\hat{\gamma}_b = 6.25$ dB, we obtain the significant gain of about 0.8 dB.
Table 9.13. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) \(D_3^3 + 2D_2^2 + D_1 + 3, 2D_2^2 + 2D + 2\) and the best ML code (ML-MAP) \(D_3^3 + D_2^2 + 2D + 1, D_2^2 + 2D + 2\) using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. Rayleigh channel.

<table>
<thead>
<tr>
<th>(\gamma_b) (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>(\Delta) PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>(\Delta) PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.00</td>
<td>15.184</td>
<td>15.736</td>
<td>16.814</td>
<td>1.078</td>
<td>1.630</td>
</tr>
<tr>
<td>6.25</td>
<td>21.937</td>
<td>22.532</td>
<td>23.365</td>
<td>0.833</td>
<td>1.428</td>
</tr>
<tr>
<td>7.50</td>
<td>26.707</td>
<td>27.302</td>
<td>27.644</td>
<td>0.342</td>
<td>0.937</td>
</tr>
<tr>
<td>8.75</td>
<td>28.886</td>
<td>28.899</td>
<td>28.988</td>
<td>0.089</td>
<td>0.102</td>
</tr>
</tbody>
</table>

9.2.2 Simulation Results for the “Barbara” Image

In this section, we obtain and discuss the simulation results for the “Barbara” image in multi-quantization mode. A performance comparison between using ML-MAP and MAP-MAP for strategies 1 and 2 is shown in Tables 9.14 and 9.15, respectively. Here, the AWGN channel is used. In this case, the suboptimal MAP code is \([3D_3^3 + 3D_2^2 + 2D + 3, 2D_2^2 + D + 2]\) obtained in Section 6.4. The \(d_{free}^2\) of this code is 6.82. The gain in terms of \(E_s/N_o\) is about 0.2 dB when using the MAP code with MAP decoding at a SER about \(10^{-3}\) (See Section 6.4).

In Table 9.14, we obtain a performance improvement of about 0.7 to 0.9 dB comparing between ML-MAP and MAP-MAP at \(E_s/N_o\) in the range of 2 to 4 dB. At \(E_s/N_o = 3\) dB, the significant gain of about 0.89 dB is obtained in terms of an average PSNR. In the case of strategy 2 shown in Table 9.15, the improvement is about 0.5 to 1 at \(E_s/N_o\) in the same \(E_s/N_o\) range.

Tables 9.16 and 9.17 illustrate the simulation results for a Rayleigh channel using strategies 1 and 2, respectively. For strategy 1, a performance improvement from using
<table>
<thead>
<tr>
<th>$E_s/N_0$ (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>Δ PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>Δ PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>13.252</td>
<td>13.330</td>
<td>14.197</td>
<td>0.847</td>
<td>0.945</td>
</tr>
<tr>
<td>3.0</td>
<td>19.891</td>
<td>20.170</td>
<td>21.058</td>
<td>0.888</td>
<td>1.167</td>
</tr>
<tr>
<td>3.5</td>
<td>22.178</td>
<td>22.373</td>
<td>23.293</td>
<td>0.920</td>
<td>1.115</td>
</tr>
<tr>
<td>4.0</td>
<td>23.925</td>
<td>24.023</td>
<td>24.712</td>
<td>0.689</td>
<td>0.787</td>
</tr>
</tbody>
</table>

Table 9.14. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) $[3D^3 + 3D^2 + 2D + 3, 2D^2 + D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. AWGN channel.

<table>
<thead>
<tr>
<th>$E_s/N_0$ (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>Δ PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>Δ PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.0</td>
<td>13.480</td>
<td>13.744</td>
<td>14.737</td>
<td>0.993</td>
<td>1.257</td>
</tr>
<tr>
<td>3.0</td>
<td>21.898</td>
<td>22.376</td>
<td>23.342</td>
<td>0.966</td>
<td>1.444</td>
</tr>
<tr>
<td>3.5</td>
<td>24.437</td>
<td>24.720</td>
<td>25.608</td>
<td>0.888</td>
<td>1.171</td>
</tr>
<tr>
<td>4.0</td>
<td>25.948</td>
<td>26.035</td>
<td>26.520</td>
<td>0.485</td>
<td>0.572</td>
</tr>
</tbody>
</table>

Table 9.15. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) $[3D^3 + 3D^2 + 2D + 3, 2D^2 + D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. AWGN channel.

<table>
<thead>
<tr>
<th>$\gamma_b$ (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>Δ PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>Δ PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.00</td>
<td>14.432</td>
<td>14.558</td>
<td>15.111</td>
<td>0.553</td>
<td>0.679</td>
</tr>
<tr>
<td>6.25</td>
<td>19.332</td>
<td>19.584</td>
<td>20.037</td>
<td>0.453</td>
<td>0.705</td>
</tr>
<tr>
<td>7.50</td>
<td>22.420</td>
<td>22.529</td>
<td>23.398</td>
<td>0.869</td>
<td>0.978</td>
</tr>
<tr>
<td>8.75</td>
<td>24.534</td>
<td>24.619</td>
<td>25.183</td>
<td>0.564</td>
<td>0.649</td>
</tr>
</tbody>
</table>

Table 9.16. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) $[3D^3 + 3D^2 + 2D + 3, 2D^2 + D + 2]$ and the best ML code (ML-MAP) $[D^3 + D^2 + 2D + 1, D^2 + 2D + 2]$ using MAP decoding. 200 image trials per data point. Strategy 1 is used, and it decodes all HFS packets that pass a parity check. Rayleigh channel.
Table 9.17. The comparison of reconstructed image average PSNR between our suboptimal MAP code (MAP-MAP) \([3D^3+3D^2+2D+3, 2D^2+D+2]\) and the best ML code (ML-MAP) \([D^3+D^2+2D+1, D^2+2D+2]\) using MAP decoding. 200 image trials per data point. Strategy 2 is used, and it decodes all HFS packets. Rayleigh channel.

<table>
<thead>
<tr>
<th>(\gamma_b) (dB)</th>
<th>PSNR ML-ML (dB)</th>
<th>PSNR ML-MAP (dB)</th>
<th>PSNR MAP-MAP (dB)</th>
<th>(\Delta) PSNR (dB) ML-MAP vs. MAP-MAP</th>
<th>(\Delta) PSNR (dB) ML-ML vs. MAP-MAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.00</td>
<td>14.810</td>
<td>15.173</td>
<td>15.676</td>
<td>0.503</td>
<td>0.866</td>
</tr>
<tr>
<td>6.25</td>
<td>21.118</td>
<td>21.603</td>
<td>21.888</td>
<td>0.285</td>
<td>0.770</td>
</tr>
<tr>
<td>7.50</td>
<td>24.666</td>
<td>24.824</td>
<td>25.533</td>
<td>0.709</td>
<td>0.867</td>
</tr>
<tr>
<td>8.75</td>
<td>26.383</td>
<td>26.446</td>
<td>26.636</td>
<td>0.190</td>
<td>0.253</td>
</tr>
</tbody>
</table>

the MAP code with MAP decoding is about 0.4 to 0.9 dB at \(\gamma_b\) in the range of 5 to 8.75 dB. The greatest improvement is found at \(\gamma_b = 7.5\) dB. In Table 9.17 (strategy 2), one can observe that the improvement ranges from about 0.2 to 0.7 dB. For the case of using “Barbara” image and multi-quantization mode, the average PSNRs obtained by using strategy 2 (Table 9.15 and 9.17) is better than those obtained by using strategy 1 (Tables 9.14 and 9.16).
Chapter 10

Conclusions and Future Work

10.1 Conclusions

In this dissertation, we proposed an error resilient image transmission system based on a method of joint source-channel (JSC) coding. The transmitting system consists of a MPEG-4 image encoder, a ring convolutional encoder, and an $M$-ary CPFSK modulation scheme. At the receiver, maximum a posteriori (MAP) decoding, based on the Viterbi algorithm and the use of residual source redundancy, is applied. In particular, the 4-ary zerotree symbols in the compressed image bit stream are modeled as a first order Markov source. This source can be characterized by using a source transition matrix (STM). The “Lena” and “Barbara” images in both single-quantization and multi-quantization modes are used, resulting in four STMs. The performance in terms of SER vs. $E_s/N_0$ for MAP decoding is shown in Chapter 7 for both the AWGN and Rayleigh fading channels. For 32 state code, the improvement in terms of $E_s/N_0$ ranges approximately from 0.6 to 0.4 dB. In Chapter 8, the simulation results show that MAP decoding can effectively improve the reconstructed image quality. While the average PSNR change is less than 1 dB, in many cases the actual improvement is more than 2 dB, significantly improving the subjective quality.

To further improve the system performance, the design of ring convolutional codes specially for MAP decoding was investigated in Chapter 9. The four STMs are used in
the design process. In the case of using the STM of zerotree symbols of the HFS of the “Lena” image in multi-quantization mode and 32 state codes, the MAP-designed code is found to be different from the best polynomial code designed for maximum likelihood (ML) decoding. A slight SNR improvement, due to using the MAP codes, of about 0.1 dB is obtained at a symbol error rate (SER) of approximately $10^{-3}$. For 64 and 128 state codes, we found different codes for all four STMs. The improvements ($E_s/N_0$) due to the MAP codes range from about 0.06 to 0.20 dB. For the 64 state MAP code, the improvement in terms of an average PSNR range from about 0.25 to 1.06 dB at $E_s/N_0 = 3$ dB.

10.2 Future Work

We propose extending the proposed image transmission system by considering serial concatenated coding (SCC), and an iterative decoding technique. There has been much research in this technique due to its promising performance [62], [63], [66], [67], and [64]. A serial concatenated coding system consists of two cascaded set of codes: outer and inner codes. The iterative decoding system consists of two cascaded decoders (for the inner and outer code) based on soft-input soft-output (SISO) algorithms [61]. Note that the outer decoder in an SCC must produce extrinsic a posteriori probabilities (APPs) of the outer code symbols. These APPs are passed through an interleaver and then fed back to an inner SISO decoding. The detail of an SCC scheme, and an iterative decoding can be found in [62]. Moqvist [63] considered SCC systems consisting of a convolutional code and continuous phase modulation (CPM) with iterative decoding. CPM is viewed as the inner code. In our case, we are interested in investigating the performance of
a SCC scheme combined with a source-controlled channel decoding technique. Figure 10.1 shows the block diagram of the SCC system using a ring convolutional code and trellis-coded CPFSK with iterative decoding. The previous trellis-coded CPFSK system, using a ring convolutional code, is shown in Figure 5.1. It is modified and treated as an inner code. At the receiver, the Viterbi algorithm is replaced by the SISO algorithm. The outer encoder is also a ring convolutional code. The system configuration will be investigated. Note that the source transition matrix of the zerotree symbols can be used as a priori probabilities $P(U^O; in)$ in an outer SISO decoder.

![Diagram of SCC system](image)

Fig. 10.1. Serial concatenated coding system using a ring convolutional code and trellis-coded CPFSK with iterative decoding.

Another possible future research area is to consider other types of CPM systems (not just CPFSK) such as $L$-RC CPM and $L$-REC CPM with $L > 1$ (partial response).
Such systems are suitable for a bandwidth limited channel since the modulated signals occupy less bandwidth than those produced by a CPFSK system. Moreover, one can apply the proposed source-controlled channel decoding technique for the image transmission system to video transmission system over a noisy channel. Some modification in the technique may be necessary in order to integrate the use of residual source redundancy into the existing video compression and transmission systems.
References


[57] P. G. Sherwood and K. Zeger, “Progressive Image Coding for Noisy Channels,”

[58] H. Man and F. Kossentini and M. J. T. Smith, “Robust EZW image coding for
1997.

zerotree wavelet algorithm,” IEEE Trans. Image Processing, vol. 6, no. 10, pp. 1436-


APP module for iterative decoding of concatenated codes,” IEEE Commun. Letters,

terleaved codes: performance analysis, design, and iterative decoding,” IEEE Trans.

[63] P. Moqvist, Serially concatenated systems: An iterative decoding approach with
application to continuous phase modulation, Ph.D. Dissertation, Dept. of Comp.


Vita

Srijidtra Mahapakulchai was born in 1969 in Lopburi, Thailand. She received her B.Eng with first class honors with Gold medal from Kasetsart University in 1992. Right after graduation, she joined the faculty member of Electrical Engineering department at Kasetsart University. In 1993, she awarded Thai Government scholarship to pursue her M.S. and Ph.D degrees in the field of communication in the United States of America. She finished her M.S. degree in Electrical Engineering from The Pennsylvania State University in 1995. In 1996, she enrolled in the Ph.D. program in the department of electrical engineering at The Pennsylvania State University. Her area of interest includes communication, detection theory, coding theory, image compression and transmission.