RATE ALLOCATION FOR MARKOV MODEL AIDED CONVOLUTIONAL CODING OF VECTOR QUANTIZED IMAGE DATA

by

XIAOPING LIN

B.Sc. (Electrical Engineering), Zhejiang University, China, 1993

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENTS FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES

DEPARTMENT OF ELECTRICAL ENGINEERING

We accept this thesis as conforming

to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA

Feb 2000

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Department of Electrical and Computer Engineering

The University of British Columbia
Vancouver, Canada

Date April 12

DE-6 (2/88)

Abstract

This thesis proposes an optimally rate allocated image transmission system that uses Vector Quantization (VQ) for source coding, and a family of variable rate Punctured Convolutional Codes (PCCs) for channel coding. At the receiver, we apply the source aided channel decoding technique known as Markov Model Aided Decoding (MMAD). Our optimality criterion is to maximize the average end-to-end Reconstruction Signal-to-Noise Ratio (RSNR) under the constraints of a fixed information rate (in pixels per second) and a fixed transmission bandwidth. For a given channel SNR, this joint source-channel coder design achieves the optimal rate allocation between the source coding and the channel coding operations. Compared with the conventional rate allocated system (the analogous system that does not use MMAD), the proposed system gives significant performance improvement. This is due to the fact that MMAD increases the strength of the channel codes, thus allowing the system to allocate more rate to the source coder, which results in a higher resolution image.

In the course of our study, we first investigate MMAD without explicit channel coding for VQ image transmission over the noisy memoryless channels comprising the Binary Symmetric Channel (BSC) and the Additive White Gaussian Noise (AWGN) channel. In order to evaluate the effects of the order of the Markov model of the data, we consider two types of decoding algorithms. One is based on the Viterbi sequence decoding algorithm, the other is based on the Bahl, Cocke, Jelinek and Raviv (BCJR) decoding algorithm. The former is computationally less complex, and is optimal (in the sense of minimizing the Bit Error Rate (BER)) for decoding with a first order (O(1)) model; while the latter allows an efficient, but slightly sub-optimal, decoding algorithm for decoding with a second order (O(2)) model. We find that most of the MMAD

coding gain is already achieved by using the O(1) model, and therefore in the remainder of the study consider it only.

We go on to analyze two types of O(1) MMAD with Convolutional Codes (CCs) employed for explicit channel coding. We call the decoders Markov Model Aided Convolutional Decoders (MMACDs), and show via simulation that the performance benefits attained by using the Markov model are similar to the large gains found for MMAD without explicit channel coding. One type of MMACD is based on the Viterbi algorithm, and applies a trellis merging technique. This decoder has an optimal BER performance, but has the constraint that the length of the source codewords be less than the memory of the CC. The other MMACD is a concatenation of a soft-output channel decoder followed by an MMAD without channel coding. This decoder does not have the constraint on the length of the source codewords, but has less coding gain than the trellis merged decoder.

Finally, we investigate the problem of optimal rate allocation between the source coding and the channel coding for VQ/PCC transmission systems that employ MMAD. Our simulation results over the AWGN channel show that the optimal rate allocated system is superior in RSNR performance to the optimally rate allocated system without MMAD. The MMAD coding gain depends on the image, but is typically 2 dB in channel SNR. We find that for the conventional system, the point of optimal rate allocation is fairly independent of the image; while for the MMAD system the allocation depends strongly on the image characteristics. Because of this, the rate allocation calculation is significantly more complicated when using MMAD. The rate allocated systems require an estimate of the channel SNR. Because in practice there will always be some inaccuracy in estimating this, to conclude our study we investigate the sensitivity of the

rate allocated systems to channel mismatch, and find them to be fairly robust.

Table of Contents

List of Tables

List of Figures

Acknowledgment

I would like to thank my parents, Genguan Lin &Yuping Chen, and my brother, Jinsong, for their eternal love and encouragement. I feel very fortunate to be a graduate student of my supervisor, Dr. Samir Kallel, and am very appreciative of his helpful suggestions and guidance throughout this project. I am deeply indebted to Dr. Robert Link for his inestimable guidance throughout this endeavour and for thoroughly proofreading this thesis report. I am very grateful to the National Sciences and Engineering Research Council of Canada, the British Columbia Advanced systems Institute and Mobile Data Solutions Inc. of Richmond, BC, for their financial aid. Finally I would like to thank all my friends in the communication group.

Chapter 1 Introduction

1.1 Motivation and Objectives

A conventional end-to-end digital transmission system is composed of a source encoder, a channel encoder, a channel, a channel decoder and a source decoder. Source coding, or source compression, reduces the symbol throughput requirement upon the transmitter by removing the redundancy in the source data. Channel coding, also called error control coding, introduces controlled redundancy into the transmitted information stream to protect it from channel noise. A basic result of Shannon's information theory states that nearly optimal communication of an information source over a noisy channel can be accomplished by separately optimizing the source coding and the channel coding operations [1]. However, this result is true only when there is no constraint on the source code dimension and the channel code block length.

Based on this result, a conventional source encoder removes as much redundancy in the source as possible in order to deliver a source encoded bit-stream of statistically independent and equally probable bits. However, the more the source data is compressed, the more vulnerable to the channel noise the coded data will be. Therefore, we need more bits for error protection, but it is expensive to design a good channel code over a very noisy channel in terms of bandwidth, delay and complexity. The Shannon theory does not give an algorithm to design good channel codes with finite block length. On the other hand, in practice, due to the lack of exact information about the source, it is not possible to design a source encoder which can remove all the redundancy – some residual redundancy always exists in the source coder output sequence. Shannon mentions that any redundancy in the source will usually help to combat noise if it is utilized at the receiving point [2]. Thus, for a practical system, the source and channel codes should not be treated sepa-

rately.

If one wishes to perform near the Shannon limit with moderate delay or channel coding block lengths, it is necessary to consider the design of the source and the channel codes jointly. It may not actually be necessary to combine the source and channel codes, but simply to jointly design them. The systems in which the source coding and channel coding operations are jointly designed are called Joint Source/Channel Coding (JSCC) systems. It has been shown by investigations [3]-[9] that JSCC systems reduce the distortion as well as complexity and delay.

JSCC systems have been divided into three broad classes [5]: *joint* source/channel coders, where the source and channel coding operations are truly integrated; *constrained* joint source channel coders, where channel coders use some knowledge of the properties of the output of a traditional source coder to mitigate the effects of the noisy channel; and *concatenated* source /channel coders, which allocate a fixed bit rate between a cascaded source coder and a channel coder. In this thesis, a system combining the latter two schemes is proposed and analyzed. Because entropy encoded data is very sensitive to noise, it has been found that for very noisy channels, that it is better to use fixed length source codes which result in substantial residual redundancy, rather than use variable length codes which effectively remove the redundancy [7]. Markov Model Aided Decoding (MMAD) is a source-aided channel decoding technique whereby the residual redundancy in the source is encapsulated in the form of a Markov Model which provides *a priori* information about the source to the channel decoder.

This thesis provides an extensive investigation into joint source-channel coding design with Vector Quantization (VQ) and Convolutional Codes (CCs), both of which are used widely in wireless image transmission systems. VQ is an efficient compression algorithm for removing

redundancy in the data source in order to maximize bandwidth utilization. Though MMAD has been investigated for Differential Pulse Code Modulation (DPCM) transmission over the Binary Symmetric Channel (BSC) and the Additive White Gaussian Noise (AWGN) channels [4][5], an extensive investigation into MMAD for VQ transmission has not been performed. This thesis contributes to this effort through the following:

- Reviewing the VQ design method including the LBG algorithm and two index assignment methods.
- Applying MMAD techniques using a first order or a second order model for the VQ transmission systems without explicit channel coding. These are based on the Viterbi and the Bahl, Cocke, Jelinek and Raviv (BCJR) algorithms. We also consider iterative model recovery techniques which allow the receiver to recover the model of the source data from the noise corrupted channel data.
- Applying the Viterbi algorithm with the trellis merging technique to develop a trellis merged Markov Model Aided Convolutional Decoder (MMACD) for VQ and CC transmission systems, and proposing an alternate concatenated MMACD based on the BCJR algorithm.

Many authors, including the present, find large Markov Model Aided Decoding gains, but this is usually because the large gain is usually in a region where the channel code is not effective. There is no MMAD gain for very high channel Signal-to-Noise Ratio (SNR). How can we evaluate MMAD more fairly? The third generation wireless systems allow multi-rate transmission, making rate-allocated systems possible. We believe that a fair system for judging the performance improvements attainable by employing MMAD techniques would be a rate allocated

system.

Therefore, in this thesis we propose and analyze an optimally rate allocated image transmission system that uses VQ for source coding, and a family of variable rate Punctured Convolutional Codes (PCCs) for channel coding. We consider the rate allocation problem for both MMAD and conventional channel decoding. The combination of source and channel coding used maximizes the end-to-end Reconstruction Signal-to-Noise-Ratio (RSNR) under the constraints of a fixed channel bandwidth, and a fixed information transmission rate. More specifically, this thesis contributes through the following:

- Finding the best PCCs for our system requirements.
- Proposing a rate allocated system with MMAD. Applying model simulation to study the optimal rate allocation for the rate allocated systems with and without MMAD. Comparing the performance of the proposed system with that of the conventional rate allocated system without MMAD.
- Examining the channel mismatch effect for the optimal rate allocated systems with or without MMAD.

1.2 Thesis Outline

This thesis is composed of the following:

Chapter 2 presents the background necessary for a basic understanding of VQ. The LBG algorithm is introduced. We investigate the performance of two index assignment algorithms and choose one for our later investigations.

Chapter 3 focuses on the basic VQ transmission system over the memoryless channels BSC and AWGN without explicit channel coding. We consider both symbol-by-symbol and sequence decoding algorithms. As well we consider using both first order (O(1)) and second (O(2)) Markov models. The symbol-by-symbol decoding algorithms, as well as the sequence decoding algorithms that use the O(2) model, require feedback of previous decoding decisions. We consider two types of feedback: feedback of hard decoding decisions, and feedback of *a posteriori* symbol probabilities. These are called Hard-Decision-Feedback (HDF) and Soft-Decision-Feedback (SDF), respectively. The HDF sequence decoders are based on the Viterbi algorithm, while the SDF sequence decoders are based on the BCJR algorithm. In the last part, an iterative source recovery technique is applied to obtain the source model without *a priori* source information.

In Chapter 4, the investigation is continued to MMACDs for VQ data transmission with CCs. One type of MMACD uses a trellis merging technique, which allows one to efficiently use a Markov model while employing the Viterbi decoding algorithm. Another type of MMACD proposed is a concatenated decoder, which uses the BCJR algorithm in a pure channel decoder, followed by the Viterbi based MMAD for no channel coding. The former is computationally simpler, and is optimal in the sense of minimizing the bit error rate. However, it has a restriction (that the length of the source codewords be less than the channel code memory) that the latter decoder does not have.

In Chapter 5, we propose an optimally rate allocated MMAD system for VQ with PCC transmission over the AWGN channel. The model simulation scheme is used to determine the optimal rate allocation between VQ and PCC for the systems with and without MMAD. Their

optimal rate allocation for the different images is studied. We compare the RSNR performance of the proposed system with that of the conventional rate allocated non-MMAD system. Also, their sensitivity to channel mismatch is examined.

In Chapter 6, we make concluding remarks and present suggestions for future work.

Chapter 2 Vector Quantization

In this thesis, all our image transmission systems are based on Vector Quantization [10]-[20] as the source coding technique. VQ provides two attractive features: high compression ratio and simple decoder structure; thus, it has aroused wide attention for many years. It has been extensively used for source coders in digital transmission of analog signals. Whether or not the source output values are correlated or uncorrelated, for a given rate in bits per pixel (bpp), properly designed vector quantizers will always perform better (less distortion) than scalar quantizers [10].

VQ is like scalar quantization except that all components of Ω successive source samples, are quantized simultaneously. Normally these Ω successive source samples are formed by grouping a block of pixels (quantization block) together in a vector. As such, a vector quantizer is characterized by a Ω-dimensional partition, a Ω-dimensional *codebook* (containing Ω-dimensional *points, reproduction codewords or codevectors*), and an assignment of binary codewords (called *indices*) to the cells of the partition (equivalently, to the codevectors).

For Ω-dimensional VQ, image vectors are formed by dividing an image into non-overlapping blocks of Ω pixels. Each input vector is compared with the codevectors of the codebook. The index of the nearest codevector is sent to the decoder. The decoder has a codebook identical to the encoder, and decoding can be implemented by a simple table look-up operation. The pictorial representation of this procedure is shown in Figure 2.1 [10].

There are the two main problems in VQ design. One of the problems is how to generate the reproduction vectors, or the codebook over the source; the other is how to choose the binary

representation of the reproduction codevectors (or the indices), so that the effect of channel errors is not too degrading on the performance.

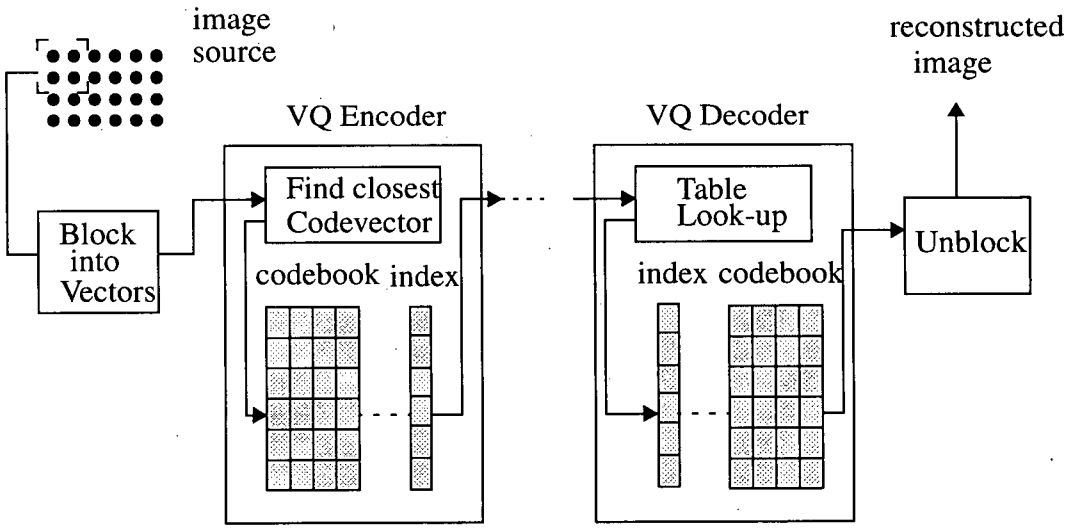


Figure 2.1: Diagram of Vector Quantizer

2.1 The LBG Design Algorithm

The popularly known Linde-Buzo-Gray (LBG) algorithm or the generalized Lloyd algorithm (GLA) [14] forms the basis of most vector quantizer designs. There exists various VQ design algorithms, and although each has found its adherents, none convincingly yields significant benefits over the LBG algorithm and its variations in terms of trading off rate and distortion [15]. In this paper we use the LBG algorithm as our VQ codebook design algorithm due to its simplicity and its effectiveness in the compression of various source inputs [10][16].

Before we talk about the LBG algorithm, let us define some terminology first. The amount of compression will be described by the rate, in bpp. Assume we have a codebook of size M, and the input vector is of dimension Ω. Each possible codevector is mapped to an *index* in order to

inform the decoder which codevector was selected. The index can be a fixed length (fixed-rate VQ) or a variable length (variable-rate VQ). In our system we only consider fixed rate VQ, because variable-rate VQ, though it has better compression performance, results in data that is more sensitive to the channel noise. Thus, each codevector is represented by an index with $q=\lceil \log_2 M \rceil$ bits. For simplicity, we call this VQ a *q-bit VQ*. As each codevector represents the reconstruction values for Ω source output samples, the rate for a Ω-dimensional vector quantizer with a codebook size M would be $R_s = \frac{\lceil \log_2 M \rceil}{\Omega}$ bpp.

Define a Ω-dimensional vector quantizer as a mapping function y=q(x), where x represents a source vector, y represents the corresponding reproduction codevector. Each time it assigns a typical *source vector* (a block of Ω source symbols) $x_n = (x_n^0, x_n^1, \ldots, x_n^{\Omega-1})$ to a codevector $y=q(x)$, $y_n = (y_n^0, y_n^1, \ldots, y_n^{\Omega-1})$ drawn from a finite reproduction codebook $C = \{c_0, c_1, \ldots, c_{M-1}\}$, where c_i is a reproduction codevector. The vector quantizer q is completely described by the codebook C with the *quantization regions* $R = \{R_0, R_1, \ldots, R_{M-1}\}$. $q(x) = c_i$, if $x \in R_i$, $R_i = \{x: d(x, c_i) < d(x, c_j) \quad \forall i \neq j\}$, $i = 0, 1, \ldots, M-1$. Normally we measure the distortion between a source vector and the reproduction codevector by the *mean squared error*:

$$d(x, y) = \|x - y\|^2 = \sum_{i=0}^{\Omega-1} (x^i - y^i)^2 \tag{2.1}$$

this is also called *Euclidean distance*.

We assume to use the image source data as the training set for the codebook design. Suppose the total number of the source symbols in one image is N_t. Then, for Ω-dimensional VQ, the total number of source vectors is $N_v = N_t/\Omega$. The LBG or GLA algorithm proceeds as follows [10]:

1: Start with an initial set of reconstruction values or an initial codebook $\{y_i\}^{(0)}$, $y_i = (y_i^0, y_i^1, \ldots, y_i^{\Omega-1})$, $i = 0, 1, \ldots, M-1$ and a set of source vector $\{x_n\}$, $n = 0, 1, \ldots, M-1$. Set the repeating time variable k=0 and the original average distortion $D^{(0)}$=0. Select threshold ε.

2: Calculate the quantization regions $\{R_i^{(k)}\}$, $i = 0, 1, \ldots, M-1$ by:

$$R_i^{(k)} = \{x_n: d(x_n, y_i) < d(x_n, y_j) \quad \forall i \neq j\} \; i = 0, 1, \ldots, M-1 \tag{2.2}$$

3: Compute the average distortion $D^{(k)}$ between the source vectors and the reproduction codevectors.

$$D^{(k))} = \frac{\sum_{n=0}^{N_v-1} d(x_n - q(x_n))}{N_v} \tag{2.3}$$

4: If $\frac{D^{(k)} - D^{(k-1)}}{D^{(k)}} < \varepsilon$ stop; otherwise, continue.

5: k=k+1. Obtain new reconstruction values $\{y_i\}^{(k)}$, $i = 0, 1, \ldots, M-1$ by calculating the average value of the elements of each of the quantization regions $R_i^{(k)}$. Go to Step 2.

The selection of the initial codebook is very important. Linde, Buzo, and Gray described the *splitting* algorithm in their original paper [14]. It starts by designing a codebook of size one, or

a one level vector quantizer. For the one-level quantizer, the output point of the codebook is the average value of the entire source vectors. From this point, one can obtain the initial codebook for a two-level vector quantizer by including the reproduction codevector of the one-level quantizer and a second codevector, obtained by adding a *perturbation vector* δ, which can be a fixed or a random vectors. Then the LBG algorithm is applied to get the final reproduction codevectors of the two-level VQ. In the same way, one can obtain the initial codebook of a four level quantizer by splitting the final output points of the two-level quantizer. Then the LBG algorithm is used to get the final codebook for the four-level VQ. In this manner, we keep doubling the number of quantization levels until we reach the desired number of levels [10].

After the final reconstruction values $\{y_i\}^{(k)}$, $i = 0, 1, \ldots, M-1$ are obtained, we can form the final codebook $C = \{c_0, c_1, \ldots, c_{M-1}\}$, where c_i represents one of these reconstruction codevectors. Then these codevectors are represented by q-bit indices, $q = \lceil \log_2 M \rceil$, which are the most efficient way to represent source information.

Suppose we sent these indices into a noisy channel. Because essential information of source data is extracted and source redundancy is removed by VQ, the vector quantized data are very sensitive to channel noise. Proper assignment of binary indices to VQ codewords can reduce the distortion due to the channel noise without an increase in bit rate or decoding complexity. If the index mapping function is $\gamma(c)$, for 2^q quantization levels, there are q! possible ways to map a code vector c_i onto an index $\gamma(c_i)$. Index assignment is an important topic in VQ design. In recent years, some iterative index assignment algorithms like the pseudo-Gray coding [17] and the simulated annealing [18] have been reported. However, these iterative procedures are quite computa-

tionally intensive. In the next section, we consider two computationally efficient index assignment algorithms: the Growth algorithm [19] and the Natural Binary Codes (NBCs) [18] from the splitting algorithm. The performance of each is compared.

2.2 Index Assignment

Consider the design of a Ω-dimensional, q-bit VQ with the codebook $C = \{c_0, c_1, \ldots, c_{M-1}\}$, where M = 2^q. Let the index set be denoted by $I = \{0, 1, \ldots, M-1\}$. We define a one-to-one index mapping as γ: $C \rightarrow I$, which assigns the indices to the codevectors. Suppose c_i is represented by the index $i = 0, 1, \ldots, M-1$. The indices of the quantized vectors are transmitted over a noisy memoryless channel. Figure 2.2 shows the VQ transmission system without explicit channel coding.

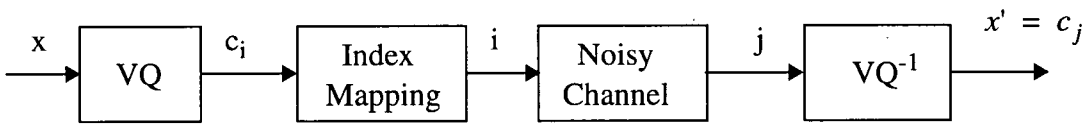


Figure 2.2: VQ data transmission system

Due to the channel noise, index i may become index j at the VQ decoder, $j = 0, 1, \ldots, M-1$. Let $p(j|i)$, i, j = 0,1,...,M-1, denote the probability that j is received given that i is actually sent over the noisy channel. The overall distortion per source sample of the communication system is as follows:

$$D(q, \gamma) = \frac{1}{\Omega} \sum_{i=0}^{M-1} \sum_{j=0}^{M-1} p(j|i) \int_{R_i} p(x) d[x, c_j] dx \qquad (2.4)$$

where d(x,y) is the distortion measure between codevector x and y given by (2.1), and R_i is the

quantization region of the codevector c_i. When the distortion is measured by mean square error (2.1) and the codevectors are the centroid of their respective quantization region, (2.4) can be written as [18]:

$$D(q,\gamma) = \frac{1}{\Omega}\sum_{i=0}^{M-1}\int_{R_i} p(x)d[x,c_i]dx + \frac{1}{\Omega}\sum_{i=0}^{M-1}\sum_{j=0}^{M-1} p(i)p(j|i)d[c_i,c_j] \tag{2.5}$$

where the first term is the average quantization distortion per source sample. We use $D_S(q)$ to denote it, and call it the *source distortion*. We also note that the value of the second term will be influenced by the source information, the index assignment and the channel characteristic. This term is denoted by $D_C(\gamma)$ and is called the *channel distortion*. Thus we have

$$D(q,\gamma) = D_S(q) + D_C(q,\gamma) \tag{2.6}$$

It has been shown that a substantial reduction in channel distortion can be obtained through an appropriate index assignment rather than a random assignment [20]. Recently, H-S Wu and J. Barba developed an efficient index scheme called the Growth Algorithm [19] which is simple computationally and which has excellent performance compared with the average random assignment. In what follows we will present this algorithm and another algorithm called the NBCs from the splitting algorithm.

2.2.1 The Growth Algorithm

The Growth index assignment technique [19] was published in 1993. The channel is assumed to be memoryless with bit error probability ε. Accordingly, $p(i|j) = \varepsilon^m(1-\varepsilon)^{q-m}$, where m is the Hamming distance of the index i and j. For simplicity, assume that no more than

one bit is in error for any one received binary index. Let h(i,j) denote the Hamming distance between index i and j. $H_1(i,j)$ represents those indices i and j whose Hamming distance between them is 1; $H_1(i,j) = \{ (i,j) \mid h(i,j)=1 \}$. Thus, the channel error transition probability can be described as

$$p(i|j) = \begin{cases} 1-q\varepsilon & for i = j \\ \varepsilon & for (i,j) \in H_1(i,j) \\ 0 & otherwise \end{cases}$$

Thus, the second term in (2.5), the channel distortion, can be rewritten as:

$$D_C(q,\gamma) = \frac{\varepsilon}{\Omega} \sum_{i=0}^{M-1} p(i) \sum_{(i,j) \in H_1(i,j)} d[c_i, c_j] \tag{2.7}$$

To reduce the channel distortion, one should try to assign the codevectors with smaller Euclidean distance onto the indices with Hamming distance equal to one. When the indices are represented by q bits, there are 2^q indices in total. The Growth algorithm is explained as follows [19]:

This algorithm starts to establish a q+1 multilevel index structure. The structure begins with the top level which contains only one node. Any indices whose Hamming distance is 1 from the index on the top node is allocated on the first level. Any indices whose Hamming distance is 1 from any index on the $(m-1)^{th}$ level, except those which have been assigned to 0^{th} ~ $(m-2)^{th}$ level, is assigned to the m^{th} level. There are $\frac{q!}{[(q-m)!m!]}$ nodes on the m^{th} level, where $0 \le m \le q$, there are 2^q nodes in the structure in total. Figure 2.3 shows an example of the multilevel structure for q=4. Any indices which are separated by Hamming distance 1, are connected by a straight line

directly. The number shown at each node is the index representing the code word at the node.

After assigning the indices on the structure nodes, we assign the codevectors from the codebook on the according nodes. To make the $D_C(\gamma)$ smaller, two codevectors with small Euclidean distance between them tend to be assigned as neighboring nodes linked by a straight line; and a centroid with large *a priori* probability tends to be assigned to the high levels so that this centroid has more freedom to be chosen as a neighbor of those codes. Thus, each node is assigned one index code and one codevector. A one-to-one mapping can be established by mapping the index in each node to the codevector in the same position corresponding to the identical multilevel structure created based on the Hamming distance of indices.

This algorithm is computationally very simple and it has to consider the Euclidean distance between the codevectors, the probability of the codevectors and the Hamming distance between the indices. Its performance is much better than the average performance attained by random index assignments. It is not the optimal assignment, but it comes close to an optimal assignment [19].

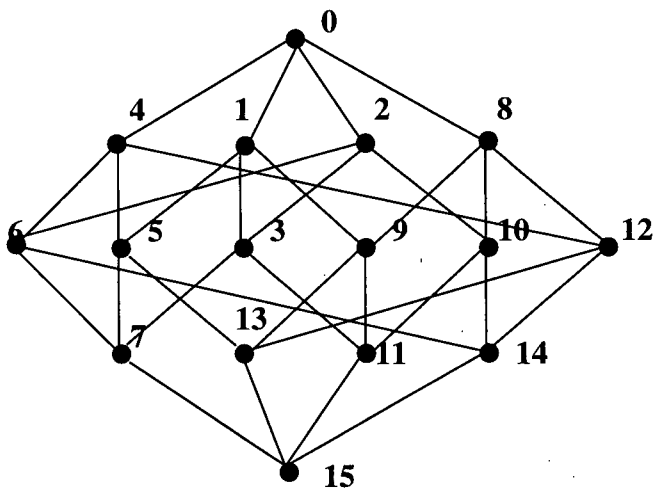


Figure 2.3: Multilevel structure for index assignment for q=4 VQ

2.2.2 Natural Binary Codes from the Splitting Algorithm

Suppose the splitting algorithm is used to initialize the vector quantization procedure. To design a Ω-dimensional vector quantizer, first we compute the average of the entire input as the output codevector of a one-level vector quantizer, say y_0. We put it on the top node of a tree. After splitting y_0 into y_0 and $y_0(1+\delta)$, we apply the LBG algorithm to get the two codevectors, called y_{10} and y_{11}, as the output of the two-level vector quantizer. We put them on the second level of the tree and label them 0 and 1 respectively. In the same manner, we will get the four output codevectors called y_{20}, y_{21}, y_{22}, y_{23}, labelled as 00, 01, 10, 11 respectively. As we continue splitting and building the tree, we are also building binary strings with the desired codevectors having binary codes called Natural Binary Codes (NBCs). Figure 2.4 shows an example of a decision tree for a 8-level VQ with the 3-bit NBC indices. The leaves of the tree are the output codevectors and the label inside each node leaf is the index corresponding to the output codevector.

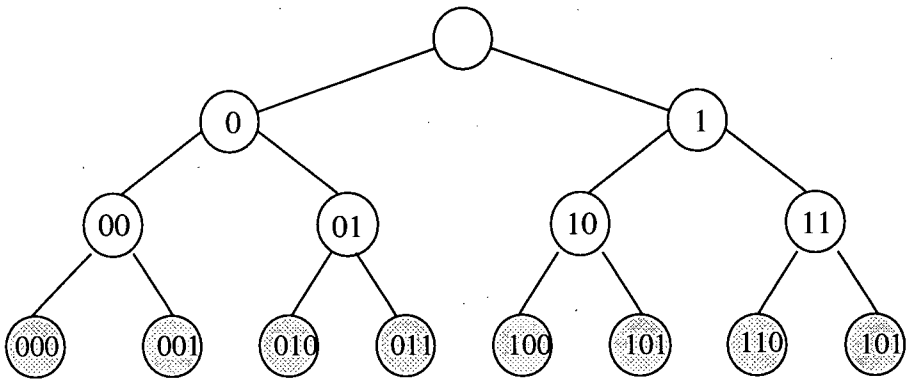


Figure 2.4: Decision tree for Natural Binary Codes, 3-bit VQ

We have applied the NBC and the Growth algorithms respectively in the system of Figure 2.2 with the 512 by 512 standard monochrome Lena image. For our simulations, in order to keep the required side information low, we use a low dimension 2 by 2, $\Omega = 4$, LBG VQ then forms a

codebook composed of 16 codevectors. After compressing the image into the indices, we transmit sends these indices over the BSC channel. The VQ decoder decodes these noise affected indices to reconstruct the image according to the codebook. In this paper, we suppose the codebook is known perfectly by the VQ decoder. The performance is measured in terms of the Reconstruction Signal-to-Noise Ratio (RSNR).

$$RSNR = 10log_{10}\frac{\sum_{r,c} x_{r,c}^{2}}{\sum_{r,c}(x_{r,c} - x'_{r,c})^2} \tag{2.8}$$

where $x_{r,c}$ is original image source value at the point of row r and column c and $x'_{r,c}$ is the reconstructed value at the same position.

The RSNR performance for 4-bit VQ using the Growth algorithm and the NBC is shown in Figure 2.5. It is interesting to find that the NBC performs about 2 dB better than the growth algorithm. We also tried other rate VQs. In most cases the NBCs from the splitting algorithm performs very well. The reason is that the codevectors will be concentrated on a thin ellipsoid in the Ω-dimensional space, and the binary partitions obtained by the splitting algorithm are such that "most" vectors in the same region are closer to one another than to those in other regions; and therefore, to a great extent, binary codewords of small Hamming distance will be assigned to the codevectors of small Euclidean distance [18]. Also due to the NBC algorithm's simplicity, in the latter, we use it as our index assignment algorithm.

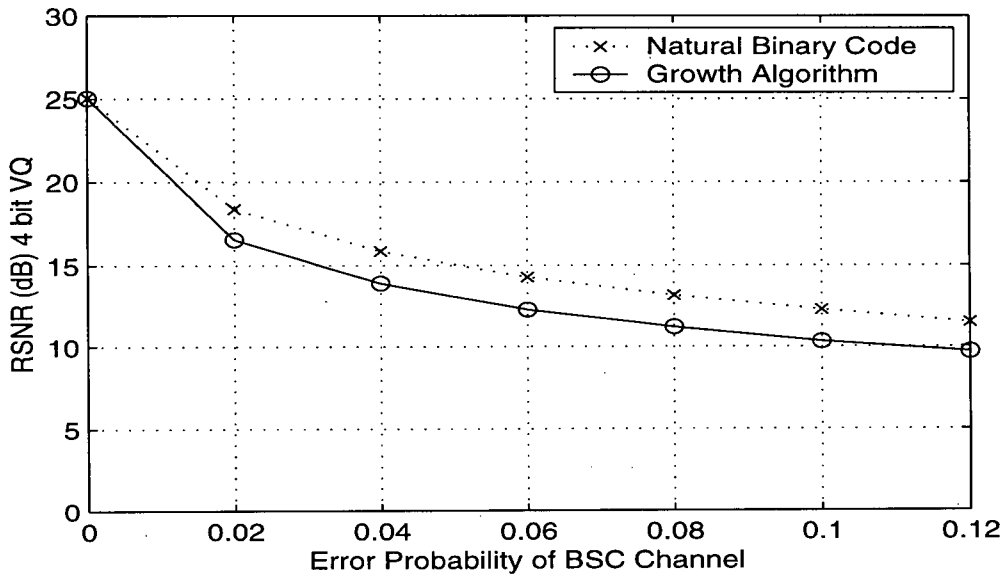


Figure 2.5: RSNR performance of different index assignment schemes

Figure 2.6 shows the original Lena image and the applied 3-bit, 4-bit, 5-bit VQ reconstructed images without noise influence. Their rates are 8 bpp, 0.75 bpp, 1.00 bpp, 1.25 bpp and 1.50 bpp respectively. Figure 2.7 shows the original 256 by 256 Sena image and 3, 4, 5-bit VQ reconstructed images. Note that fairly high quality is already achieved at the source coding rate of 1.00 bpp.

Original Image

RSNR=23.13dB using 0.75bpp

RSNR=25.04dB using 1.00bpp

RSNR=26.80dB using 1.25bpp

Figure 2.6: Original 512 by 512 Lena image and VQ reconstructed images

Original Sena image

RSNR=21.32dB using 0.75bpp

RSNR=23.40dB using 1.00bpp

RSNR=25.75dB using 1.25bpp

Figure 2.7: Original 256 by 256 Sena image and VQ reconstructed images

Chapter 3 Markov Model Aided Decoding without Explicit Channel Coding

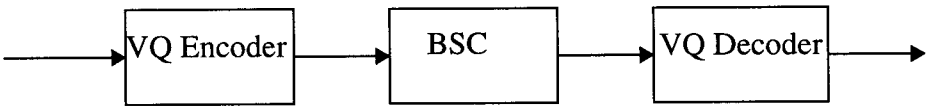
It was shown in Chapter 2 that a good index assignment can reduce the end-to-end distortion. In addition to that, we will exploit another way to protect VQ data against the channel errors without increasing the bit rate. It is known that the residual redundancy in the source encoded data can and should be exploited for channel error protection [21]. Although VQ removes much redundancy, there is still some statistical dependency existing among spatially neighboring VQ encoded symbols, especially when the dimension of the vector quantizer is low. It is known that Markov Model Aided Decoding (MMAD) techniques can make good use of residual redundancy to help error protection [4][5]. In this chapter, we will study MMAD techniques for vector quantized image data transmission without explicit channel codes.

Although MMAD techniques as described below can be used in conjunction with any fixed length source code, in this paper we consider only images encoded by a vector quantizer. In MMAD one models the source data set as a finite-state Markov sequence, and integrates it into the channel decoding process. Therefore, the MMADs require that the receiver have the model of the source. There are two ways for the receiver to obtain the source model parameters. One method is to measure the model parameters at the transmitter and then sent them as side information. Alternatively, it is possible for the receiver to recover the source model without *a priori* information, as we will discuss in the latter part of this chapter.

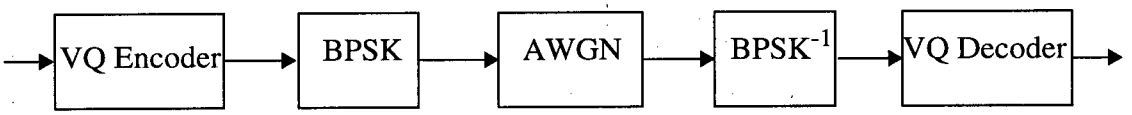
Suppose an image is quantized by a VQ with codebook size 2^q, and that the corresponding index is q-bit. These indices are represented by a symbol, $s_{r,c}$, of a Q-ary alphabet, defined at each point (r, c) of the VQ encoded image; where r is the row index and c is the column index, $Q = 2^q$.

Here we will compare the RSNR performance between the MMAD and the non-source aided decoding.

We consider the decoding of the sequence of received symbols resulting from the row-by-row transmission over the noisy memoryless channels BSC or AWGN. For the AWGN channel, each bit 0 or 1 is BPSK modulated into ± 1 before transmission over the channel. The block diagrams of the conventional systems without source aided decoding are shown in Figure 3.1.



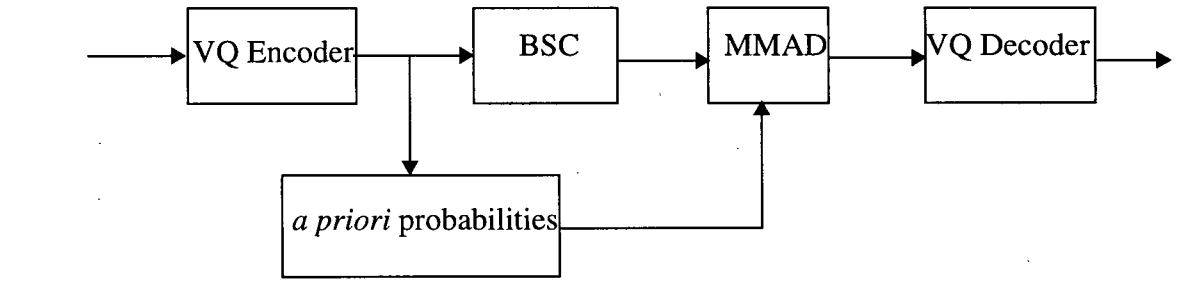
(a)



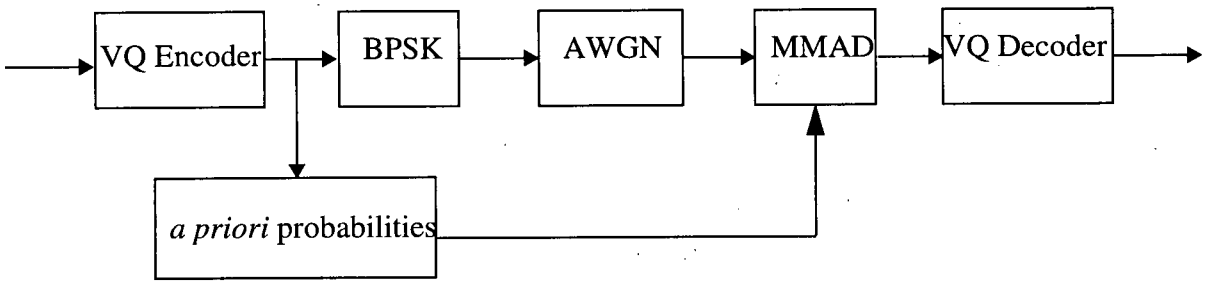
(b)

Figure 3.1: Non-source aided VQ transmission systems (a) BSC (b) AWGN

The block diagrams of the system with MMAD are shown in Figure 3.2, in which we assume for now that the knowledge of the source is known perfectly by the receiver. The sequence of q-bit symbols is modeled as a finite order Markov sequence. The RSNR performance between the two systems will be compared.



(a)



(b)

Figure 3.2: VQ/MMAD transmission systems (a) BSC (b) AWGN

In Section 3.1 and 3.2, we will review the MMAD principles which are modified based on the Viterbi algorithm or the BCJR algorithm. Two different types of Markov Model will be considered and compared. One is the first order (O(1)) Markov model, the other is the second order (O(2)) Markov model. In section 3.3, an iterative model recovery technique is applied to allow the receiver to recover the source model parameters without any *a prior* information. Here we assume that the VQ used in our simulations is a 4-dimensional LBG VQ with NBCs indices.

3.1 MMAD Based on the Viterbi Algorithm

The Viterbi algorithm is normally a Maximum Likelihood (ML) sequence decoder [22] (refer to Section 4.1), which selects the sequence $\underline{s}$ that maximizes $\sum_r \sum_c P(\hat{s}_{r,c}|s_{r,c})$, where $P(\hat{s}_{r,c}|s_{r,c})$ is the probability of the receiving symbol $\hat{s}_{r,c}$ given that $s_{r,c}$ was transmitted; and depends only on the channel. Note that the Viterbi algorithm is not limited to convolutional codes because one can still form the source symbols into a two dimensional trellis (time forming one dimension, the states forming the other). Here the value of a state is the value of a VQ symbol. We apply the Viterbi decoding algorithm in a block decoding mode where each block is a row of VQ image data. We use the term *depth* to refer to the decoder memory in a particular direction. MMAD based on the Viterbi algorithm takes advantage of *a priori* source statistics information in such a manner that MAP sequence coding is achieved (refer to section 4.1).

Here two different types of the Markov model (O(1) and O(2)) are considered and their performance is compared. The necessary modification to the Viterbi algorithm for MMAD is shown in the following.

3.1.1 Viterbi Based O(1) MMAD

In O(1) MMAD, one models the VQ source data as a first order Markov model. This model is defined by the *a priori* probability of a symbol given that the symbol previous in the row is known, for all Q symbols in the code alphabet. It is assumed that the *a priori* conditional probabilities (Markov model) of the VQ source data are exactly known to the receiver. Since the rows are independent, each may be decoded separately. The horizontal depth of the decoder is the row size, the vertical depth is 0. It is shown that using Viterbi based O(1) MMAD to decode the

received sequence optimally in the sense of minimizing the probability of sequence error, the receiver must choose the sequence $\underline{s}$ which maximizes [4][5]:

$$\prod_{c} P(\hat{s}_{r,c}|s_{r,c})P(s_{r,c}|s_{r,c-1}) \tag{3.1}$$

where $P(s_{r,c}|s_{r,c-1})$ is the *priori* probability of $s_{r,c}$ given that $s_{r,c-1}$ was previous symbol, which depends only on the source and defines the Markov model.

The most computationally efficient way to decode this sequence is to form a trellis, by indexing the states with both a state index and a time index, and apply the Viterbi algorithm to decode one row (with fixed row index r) at a time. The example of a 4-state trellis is shown in Figure 3.3.

Assume that the number of symbols per row is L, of that there are Q states at each time c, corresponding to the symbols $s_{r,c}$ of the Q-array symbol alphabet. There is a *branch metric* calculated between every pair of states consecutive in time whose value is given by:

$$\log P(\hat{s}_{r,c}|s_{r,c}) + \log P(s_{r,c}|s_{r,c-1}) \tag{3.2}$$

Among the paths terminating on a given state, the one having the largest cumulative metric is stored, resulting in Q paths at any given time. Of these, the path with the largest cumulative metric is the decoder's best estimate of the transmitted sequence of symbols up to the current time.

When the channel is the BSC, the channel transition probability is:

$$P(\hat{s}_{r,c}|s_{r,c}) = \varepsilon^{d}(1-\varepsilon)^{q-d} \tag{3.3}$$

where ε is the channel bit error probability, d is the hamming distance between the transmitted symbol $s_{r,c}$ and the received symbol $\hat{s}_{r,c}$.

When the channel is the AWGN, define the channel SNR = E_b/N_0, where E_b is the received energy of per bit, N_0 is noise spectral density, then we have:

$$P(s_{r,c}|\hat{s}_{r,c}) = \sum_{i=0}^{q-1} P(s^i_{r,c}|\hat{s}^i_{r,c}) \tag{3.4}$$

$$P(s^i_{r,c}|\hat{s}^i_{r,c}) = \frac{1}{\sqrt{\pi \cdot N_0}} \cdot \exp\left\{-\frac{(s^i_{r,c} - \hat{s}^i_{r,c})^2}{N_0}\right\} \tag{3.5}$$

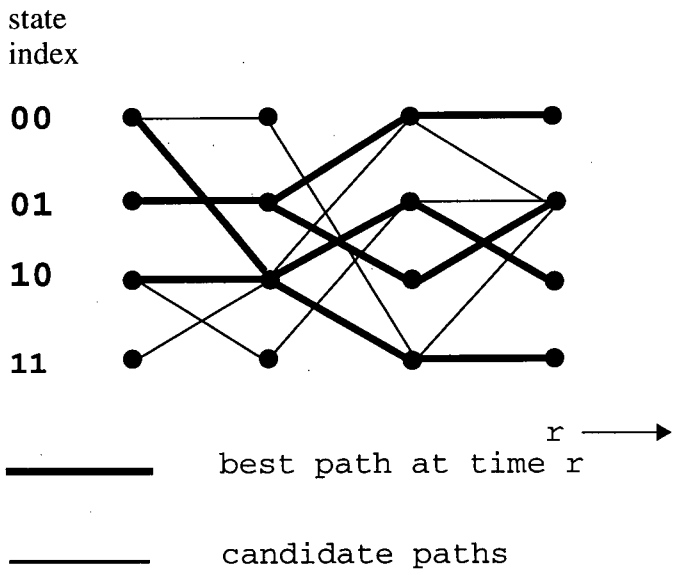


Figure 3.3: 4 state trellis for 2-bit VQ

3.1.2 Viterbi Based O(2) MMAD

For optimal decoding when source is modeled as a O(2) Markov sequence, the decoder must choose the sequence $\underline{s}$ that maximizes [4]:

$$\sum_r \sum_c P(\hat{s}_{r,c}|s_{r,c})P(s_{r,c}|s_{r,c-1}, s_{r-1,c}) \tag{3.6}$$

where $P(s_{r,c}|s_{r,c-1}, s_{r-1,c})$ is the probability of $s_{r,c}$ given that $s_{r,c-1}$ was previous in the row and $s_{r-1,c}$ was previous in the column. These Q^3 second order transition probabilities define the O(2) model at the receiver. Since the rows are no longer independent, they can no longer be decoded separately. In principle, for optimality with respect to the second order Markov model, all the rows must be decoded simultaneously.

Here we only consider the case that a decoded row is used to aid the decoding of the subsequent row. This is referred to as *vertical depth 1 sheet decoding* in reference [4]. The decoding of a row proceeds just the same as the Viterbi O(1) MMAD decoding, except that the decoder selects the sequence which maximizes:

$$\prod_c P(\hat{s}_{r,c}|s_{r,c})P(s_{r,c}|s_{r,c-1}, \bar{s}_{r-1,c}) \tag{3.7}$$

where $\bar{s}_{r-1,c}$ are the decoded symbols for the previous row and are referred to as the Hard-Decision-Feedback (HDF). The log likelihood function for the branch metric:

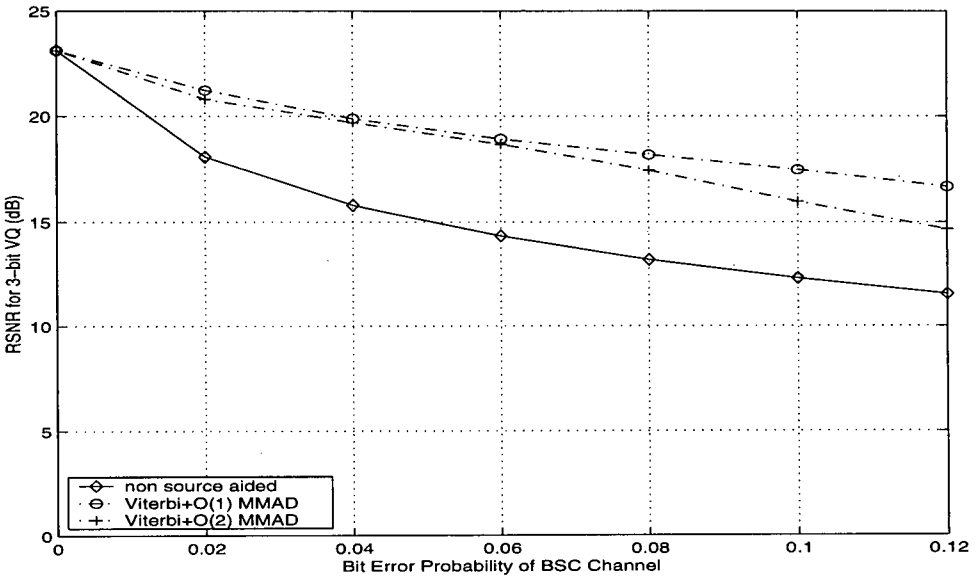
$$\log P(\hat{s}_{r,c}|s_{r,c}) + \log P(s_{r,c}|s_{r,c-1}, \bar{s}_{r-1,c}) \tag{3.8}$$

depends on the previously decoded row via the decoded symbols $\bar{s}_{r-1,c}$. Obviously, the perfor-

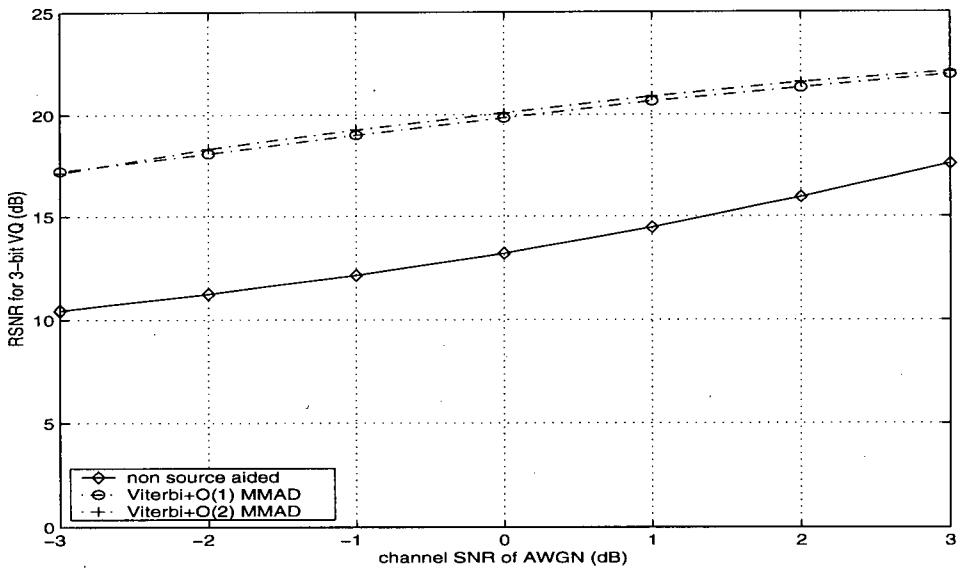
mance depends on the correctness of the previously decoded sequence.

The RSNR performance of the different decoders for 3-bit VQ transmission over BSC and AWGN is shown in Figure 3.4. We use the 512 by 512 Lena image. Compared with the non-MMAD decoders which are labelled non-source aided, the Viterbi based MMAD decoders which are labelled Viterbi + O(1) MMAD or Viterbi + O(2) MMAD, give significant improvement for VQ transmission for high channel error rates – about 5dB for the BSC channel, and 6dB for the AWGN channel. The two curves converge when channels become cleaner.

The reconstructed Lena images of the non-source aided decoding and the Viterbi O(1) MMAD for 3-bit VQ transmission over the BSC and AWGN channels are shown on in Figure 3.7. These images are obtained using the same channel Bit Error Rate (BER). The BSC channel cross-over error probability is 0.0768; while the AWGN channel SNR which results in the same bit error probability or BER of 0.0768 for the system without MMAD is 0dB. For the BPSK signal over the AWGN channel, the BER is calculated by $BER_{BPSK} = Q\left(\sqrt{\frac{2E_b}{N_0}}\right)$, where $Q(x) = \frac{1}{\sqrt{2\pi}}\int_x^\infty e^{-\frac{y^2}{2}}dy$.



(a) 3-bit VQ transmission over the BSC channel, 512 Lena



(b) 3-bit VQ transmission over the AWGN channel, 512 Lena

Figure 3.4: Viterbi based MMAD decoders without explicit channel coding

For the same MMAD system, we find that VQ using MMAD for the AWGN channel gives higher MMAD gain than for the BSC channel. Because MMAD for the AWGN channel uses the

unquantized signal level representing the transmitted bit; this unquantized signal has more channel information than the BSC quantized signal, thus allowing the source statistics to weight the decision more accurately.

Because of that, for the AWGN channel, the O(2) MMAD is a little better than the O(1) MMAD, but opposite for the BSC channel. This poor performance of the O(2) MMAD decoder relative to the O(1) MMAD decoder is due to error propagation from row to row. We find in the next section that this error propagation can be mitigated by using Soft-Decision-Feedback (SDF) information from the previously decoded row.

3.2 MMAD Based on MAP Algorithms with SDF

The O(2) MMAD based on the Viterbi algorithm uses HDF values. Here we use SDF instead of HDF to give the O(2) MMAD more correct knowledge [23]. The Viterbi based MMAD algorithm is a MAP sequence decoding method which minimizes the probability of the sequence error. However, this algorithm does not minimize the probability of symbol error. We apply two other MAP decoders which minimize the probability of symbol error. One uses a symbol-by-symbol Modified MAP (MMAP) decoding algorithm and the other uses a sequence MAP decoding algorithm known as the BCJR algorithm using the modified numerically stable vision [8][24]. These decoders provide the *a posterior probability* (*app*) which can be exploited by MMAD decoders as SDF. We will see that MMAD based on the BCJR algorithm with SDF achieves a better performance than the Viterbi based O(2) MMAD.

3.2.1 Symbol-by-Symbol MMAD

The MMAP receiver proposed [4], for data obeying a O(1) Markov model, is a symbol-

by-symbol decoder whose decoding rule is to choose the symbol $s_{r,c}$ which maximizes

$$P(\hat{s}_{r,c}|s_{r,c})P(s_{r,c}|\bar{s}_{r,c-1}) \tag{3.9}$$

where $\bar{s}_{r,c-1}$ is the previously decoded symbol in the row.

Similarly, for data obeying a O(2) Markov model, the MMAP decoding rule is to select the symbol $s_{r,c}$ which maximizes

$$P(\hat{s}_{r,c}|s_{r,c})P(s_{r,c}|\bar{s}_{r,c-1}, \bar{s}_{r-1,c}) \tag{3.10}$$

where $\bar{s}_{r-1,c}$ is the previously decoded symbol in the column. $\bar{s}_{r,c-1}$ and $\bar{s}_{r-1,c}$ are the HDF symbols. The performance of a decoder using the decision variables in (3.9) (3.10) will depend on these feedback values. A error in the HDF value might cause further errors for the present decoding. For 3-bit VQ transmission over the BSC channel, from Figure 3.5 (a), we can see that the O(1) MMAP with HDF is not better than the non-source aided decoding; and the performance of O(2) MMAP with HDF decreases quickly when the channel becomes worse.

Consider using $\pi(s_{r,c-1})$, $\pi(s_{r-1,c})$ instead of $\bar{s}_{r,c-1}$, $\bar{s}_{r-1,c}$. where $\pi(s_{r,c})$ is the *a posteriori* probability of $s_{r,c}$. The *a posteriori* probabilities of the previously decoded symbols are used as Soft-Decision-Feedback for decoding the current symbol. The MMAP decision variables for the O(1) SDF are [23]:

$$\lambda_{r,c}(s) = P(\hat{s}_{r,c}|s_{r,c} = s)\sum_{s'} P(s_{r,c} = s|s_{r,c-1} = s')\pi_{r,c-1}(s') \tag{3.11}$$

where s is the symbol state and $\pi_{r,c-1}(s) = P_{app}(s_{r,c-1} = s)$ is the *a posteriori* probability that

the state at position (r, c-1) is equal to s. The computation of $\pi_{r,c}(s)$ is given by the normalized product:

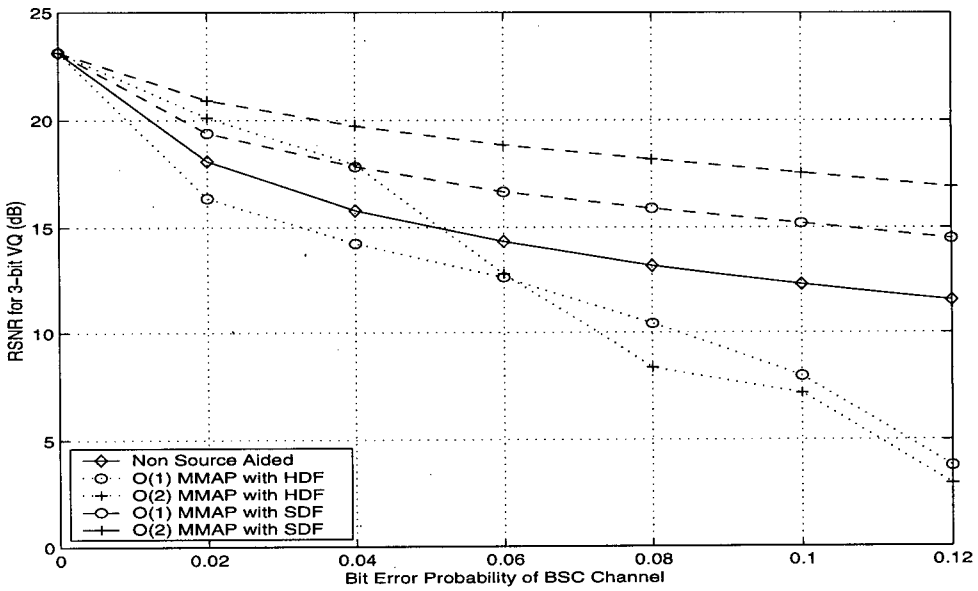
$$\pi_{r,c}(s) = \frac{\lambda_{r,c}(s)}{\sum_{s'} \lambda_{r,c}(s')} \tag{3.12}$$

The optimal decision rule is to choose the symbol $s_{r,c} = s$ that maximizes $\lambda_{r,c}(s)$. Note for the first symbol, the decoding variable is $\lambda_{r,c}(s) = p(\hat{s}_{r,c} | s_{r,c} = s) P(s_{r,c} = s)$. Similarly for MMAP with O(2) SDF, the decision variables are [23]:

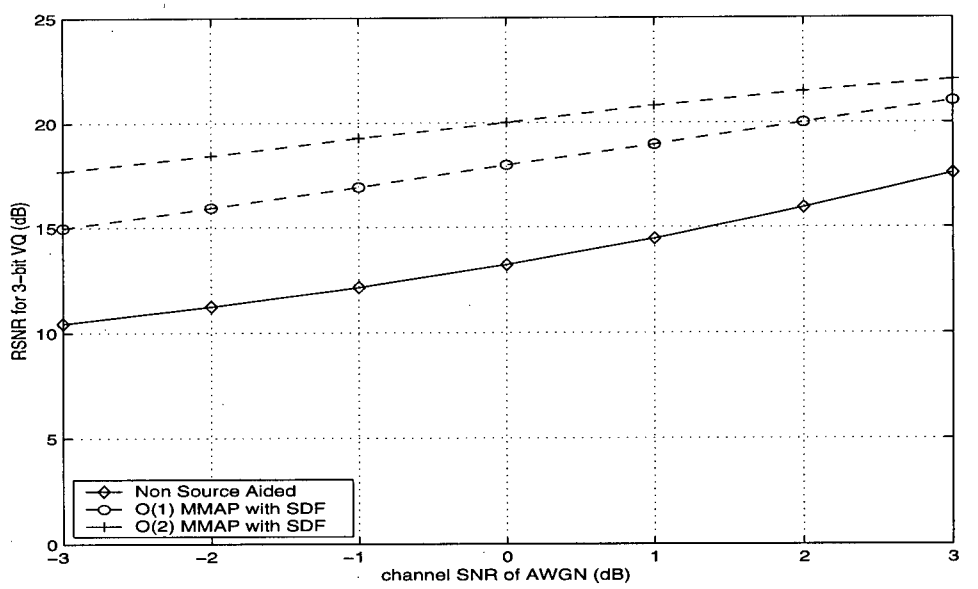
$$\lambda_{r,c}(s) = P(\hat{s}_{r,c} | s_{r,c} = s) \sum_{s''} \sum_{s'} P(s_{r,c} = s | s_{r-1,c} = s'', s_{r,c-1} = s') \pi_{r-1,c}(s'') \pi_{r,c-1}(s') \tag{3.13}$$

where $\pi_{r,c}(s)$ has the same normalization equation as (3.12).

The performance of the symbol-by-symbol MMADs with SDF is shown in Figure 3.5. We find that for 3-bit VQ transmission over the BSC channel, the O(1) MMAP with SDF gives about 3dB improvement over the non-source aided decoder; and the O(2) MMAP with SDF gives about 3dB gain over the O(1) MMAP with SDF. Compared with Figure 3.4, the O(2) MMAP with SDF has a similar MMAD gain as the Viterbi based O(1) MMAD. This is because that the former is a symbol-by-symbol decoder, while the latter is a sequence decoder. We conclude that using SDF or *a posteriori* symbol probabilities, is far superior than using HDF in terms of performance for the symbol-by-symbol decoding. In the next section, we will find that the same is true for sequence decoding.



(a) 3-bit VQ transmission over the BSC channel



(b) 3-bit VQ transmission over the AWGN channel

Figure 3.5: MMAP decoders with SDF without explicit channel coding

3.2.2 Sequence MMAD Based on the BCJR Algorithm

In this part, we develop a row-by-row MAP decoding method based on the BCJR algorithm [25] with SDF. The BCJR algorithm was derived for decoding discrete time finite state Markov sequence over noisy memoryless channels. As in the case of the Viterbi algorithm, the BCJR algorithm can be used for any linear code by forming a trellis [25]. Unlike the Viterbi algorithm, the BCJR algorithm minimizes the probability of symbol error. The decoder estimates the a *posteriori* probabilities of the states which we exploit for SDF.

Based on the algorithm [25], we develop a row-by-row BCJR decoding method with SDF using the modified numerically stable version [8][24]. Suppose the number of source symbols in a row is τ; then the input VQ data sequence for each row to a noisy memory-less channel is denoted by $s_1^\tau = s_1 s_2 \ldots s_\tau$; and the received sequence is denoted by $\hat{s}_1^\tau = \hat{s}_1 \hat{s}_{s\ldots} \hat{s}_\tau$. Define the state of the Markov sequence at one time as the value of the corresponding source symbol. Considering one row, the optimal decision rule is to choose the symbol $s_i = s$, that maximizes the *a posteriori* probability $P(s_i = s | \hat{s}_1^\tau)$ which is given by a normalized product:

$$\pi_i(s) = \frac{\alpha_i(s)\beta_i(s)}{\sum_{s'} \alpha_i(s')} \tag{3.14}$$

where $\alpha_i(s)$ is the joint probability of state at time i, $\alpha_i(s) = P(s_i = s, \hat{s}_1^\tau)$; while $\beta_i(s)$ is the probability of the received sequence from time i+1 to time τ given that the state is s at time i, $\beta_i(s) = P(\hat{s}_{i+1}^\tau | s_i = s)$.

To calculate α and β, define a probability function: $\gamma_i(s', s) = P(s_i = s; \hat{s}_i | s_{i-1} = s')$, $0 \le s, s' < Q$, $Q = 2^q$, then as shown in [25], α is given by the forward recursion [25] of the modified numerically stable version [8][24]:

$$\alpha_i(s) = \frac{\sum_{s'} \alpha_{i-1}(s') \cdot \gamma_i(s', s)}{\sum_{s''} \sum_{s'} \alpha_{i-1}(s') \cdot \gamma_i(s', s'')} \tag{3.15}$$

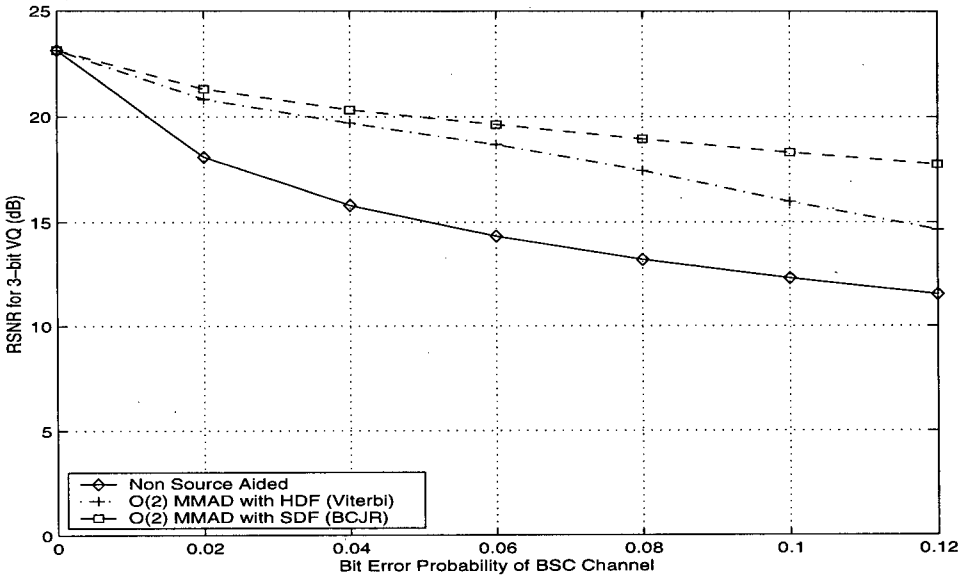
with boundary condition $\alpha_1(s) = \gamma_1(0, s)$. Similarly, β is determined by the backward recursion:

$$\beta_i(s) = \frac{\sum_{s'} \beta_{i+1}(s') \cdot \gamma_{i+1}(s, s')}{\sum_{s''} \sum_{s'} \beta_i(s') \cdot \gamma_{i+1}(s', s'')} \tag{3.16}$$

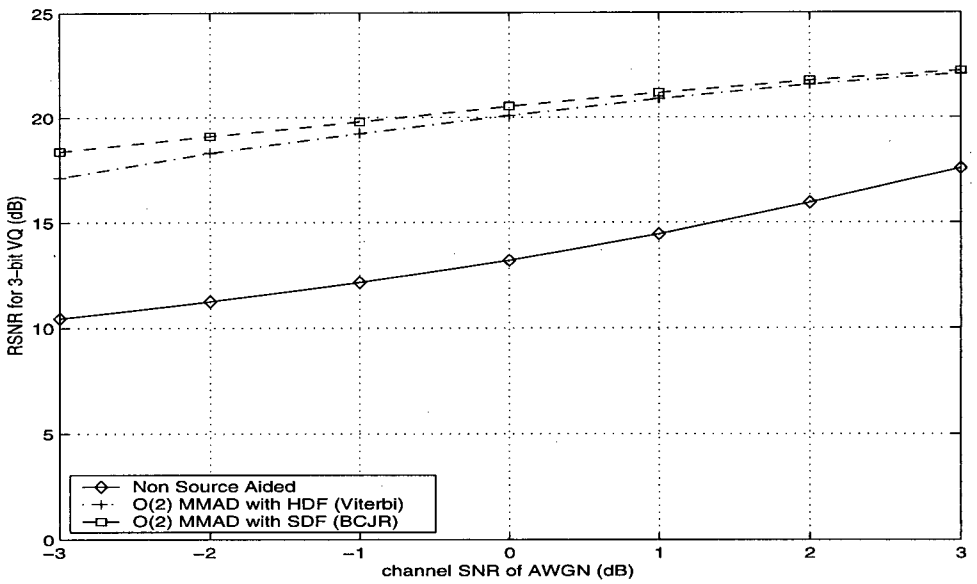
with the boundary condition $\beta_\tau(s) = \frac{1}{Q}$. Where γ depends on the channel transition probability $P(\hat{s}_{r,c} | s_{r,c})$, the *a priori* source transition probabilities of the second order Markov model $P(s_{r,c} | s_{r,c-1}, s_{r-1,c})$ and the *posteriori* probability of the previously decoded row (SDF) $\pi_{r-1,c}(s)$:

$$\gamma_{r,c}(s', s) = P(\hat{s}_{r,c} | s_{r,c} = s) \sum_{s''} P(s_{r,c} = s | s_{r,c-1} = s', s_{r-1,c} = s'') \pi_{r-1,c}(s'') \tag{3.17}$$

Figure 3.6 shows that the O(2) sequence MMAD with SDF based on the BCJR algorithm is better than the O(2) sequence MMAD with HDF based on the Viterbi algorithm for 3-bit VQ data transmission over the BSC or AWGN channel.



(a) 3-bit VQ transmission over the BSC channel



(b) 3-bit VQ transmission over the AWGN channel

Figure 3.6: O(2) sequence MMAD with HDF and SDF

Figure 3.7 also shows the reconstructed images decoded by the O(2) MMAD with SDF (BCJR) for the 3-bit VQ 512 by 512 Lena image over an AWGN channel of 0dB SNR, which

corresponds to the BSC channel of cross-over probability 0.0786. Although the advantage of the O(2) MMAD with SDF (BCJR) over the O(1) MMAD (Viterbi) is relatively small by image RSNR measure, the image decoded with the O(2) MMAD with SDF (BCJR) is visually much better in the high detail regions of the image.

Non Source Aided
RSNR=13.17dB

O(1) MMAD (Viterbi)
RSNR=18.24dB

O(2) MMAD with SDF (BCJR)
RSNR=19.00dB

(a) 3-bit VQ over BSC

Non Source Aided
RSNR =13.19dB

O(1) MMAD (Viterbi)
RSNR =19.84dB

O(2) MMAD with SDF (BCJR)
RSNR=20.53dB

(b) 3-bit VQ over AWGN

Figure 3.7: MMADs at channel SNR=0dB for AWGN, ε = 0.0786 for BSC

3.3 Iterative Source Model Recovery

In the above we assumed that the *a priori* source information is perfectly known by the receiver as the side information. Obviously this will increase the transmission overhead. Also, the

transmitter must be modified to measure and encode the source model. If the images are time variant, using *a prior* knowledge may lead to inaccuracy. We will consider a recently published iterative decoding method to recover the source model form the noise corrupted channel data [26]. The MMAD will first decode the received data without *a priori* knowledge; it then measures the statistics of the decoded data as the source model for the next decoding iteration. The convergence to the ideal MMAD is achieved after repeating the above steps a few times.

We apply this method to the 3-bit VQ data transmission over the AWGN channel with the Viterbi based O(1) MMAD. The curves in Figure 3.8 show that the process converges the remarkably quickly to the result that a receiver which has perfect *a priori* knowledge of the source model. The source model is recovered well enough at the end of 2nd iteration. Further iterations yield minor or negligible improvement. This means that by using the iterative source model recovery algorithm, we can reconstruct the noisy damaged image without *a prior* source information. In the iterative MMAD system structure of the conventional transmitter is unchanged; but there is the drawback of a processing time delay at the receiver.

In this chapter, we have found substantial performance gains by using the various kinds of MMADs for the VQ data transmission over the memoryless noisy channels. In the next chapter, we will continue to investigate MMADs for systems with channel convolutional codes. Among the various MMADs, we choose the O(1) sequence MMAD for our latter research because the following reasons: The sequence MMADs have superior performance than the symbol-by-symbol decoders. Although the symbol-by-symbol O(2) with SDF MMAD performs as well as the O(1) sequence MMAD, the O(1) sequence MMAD is easier to incorporate into the convolutional codes. The performance gain attained by using the O(2) sequence MMAD instead of the O(1)

sequence MMAD is small. Moreover, the former is significantly more complex than the latter.

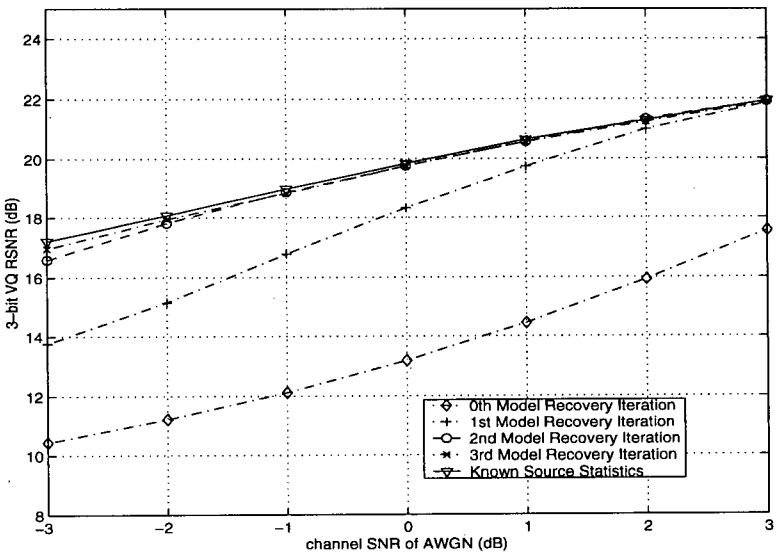


Figure 3.8: Iterative Source Model Recovery, 3-bit VQ over AWGN

Chapter 4 Markov Model Aided Convolutional Decoding

In Chapter 3 we found that MMAD without explicit channel coding gives significant coding gain for VQ transmission at high channel error rates, but provides little improvement at low channel error rates. On the other hand, conventional Forward Error Correcting (FEC) codes provide excellent error protection at low error rates, so that the combination of conventional FEC codes with MMAD can give good performance at all channel error rates. In this chapter, we will continue the development by incorporating Convolutional Codes (CCs) for VQ data transmission over the AWGN channel. Convolutional codes are selected as our FEC codes due to their pervasiveness in existing systems. We call source aided channel decoding, which uses the source statistics information in the form of a Markov Model to aid the convolutional channel decoding, Markov Model Aided Convolutional Decoding (MMACD).

Convolutional Codes were first introduced by Eli [27] in 1955. He proved that redundancy introduced into a data stream through the use of a linear shift register [22] can give substantial error protection ability. He also showed that the resulting codes were very good even when randomly chosen. This result correlated to Shannon's theory that there exist randomly selected codes that, on average, can provide high levels of error protection ability, given data transmission at a rate less than the channel capacity [28].

Let us start with the standard definition of a convolutional code [22][29][30]. A rate *k/n* convolutional encoder takes *k* input bits and generates *n* output bits. The input bit stream is fed into a shift register circuit consisting of a series of memory elements. Normally, an input data stream is fed into a shift register with memory m_i. The *total memory* is $M = \sum_i m_i$,

$i = 0, \ldots, k-1$. The *constraint length K* of a convolutional code is the maximum number of bits in a single output stream that can be affected by any input bit. The *maximal memory m* of the convolutional code is the length of the longest input shift register. In practice, the constraint length is usually taken to be the length of the longest input register plus one.

$$K = 1 + m \tag{4.1}$$

A (n, k) convolutional code with memory m is said to be an (n, k, m) CC. The *minimum free distance*, denoted d_{free} is the minimum Hamming distance between all pairs of complete convolutional code words. The convolution operation can be described by the *delay transform,* or *D-transform*,

$$X(D) = U(D) \times G(D) \tag{4.2}$$

The *transfer-function matrix* G(D) is a $k \times n$ matrix with polynomial entries for an (n, k, m) CC. U(D) is the D-transform of the input code words. X(D) is the D-transform of the output code words of the convolutional code. For example:

$$\boldsymbol{x} = (x_0, x_1, x_2, \ldots) \leftrightarrow X(D) = x_0 + x_1 D + x_2 D^2 + \ldots$$

G(D) elements are the generator polynomials $g_{i,j}(D) = \sum_{m=0}^{m_i} g_{ij}^m D^m$, where i = 1, ..., k; j = 1, ..., n. It is simple to use octal codes to represent $g_{i,j}(D)$. For example, Figure 4.1 shows an (2, 1, 2) CC with $G(D) = (1+D+D^2, 1+D^2)$. We can represent it by the generator polynomials $g_1 = 7_8$, $g_2 = 4_8$.

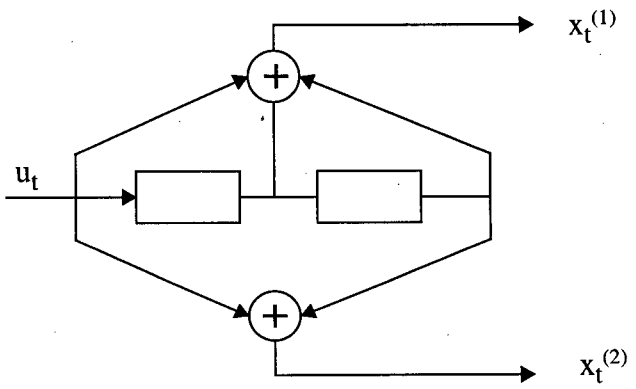


Figure 4.1: Rate 1/2, memory 2 CC

With a CC, the block diagram of the end-to-end system for q-bit VQ transmission over the AWGN channel is shown in Figure 4.2.

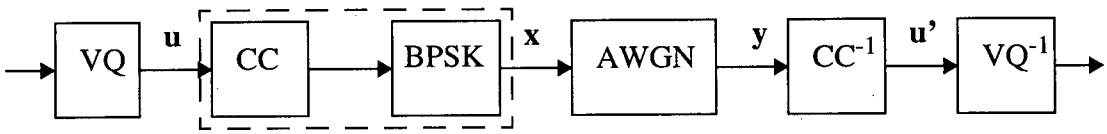


Figure 4.2: VQ with CC over AWGN

In this chapter, we will develop two different kinds of MMACD techniques. One is based on the Viterbi algorithm; the other is based on the BCJR algorithm. Their performance and computation complexity is compared and analyzed. In our simulations, we use a (3,1,5) convolutional code with the generator polynomials $g_1 = 74_8$, $g_2 = 53_8$, $g_3 = 37_8$ for the 3-bit VQ encoded Lena image transmitted over the AWGN channel.

As mentioned previously, the source model can be made known to the receiver in several ways. One is by transmitting the model parameters as side information with stronger error protec-

tive codes. Another is by estimating the model information by the iterative source model recovering technique used in Section 3.3. In the simulations discussed in the following, we assume that the source information is known perfectly by the receiver.

4.1 MMACD Based on the Viterbi Algorithm

4.1.1 The Viterbi Algorithm for Convolutional Codes

In 1976, Viterbi discovered a practical decoding algorithm for convolutional codes [28]. The Viterbi decoding algorithm has been known for at least ten years in various forms in the field of operations research due to its simplicity and low decoding complexity.

Consider the decoding problem presented in Figure 4.2. An information VQ sequence **u** is encoded by a CC and modulated into the sequence **x**, which is transmitted over an AWGN channel. The Viterbi decoder takes the received sequence **y** and generates an estimated **u'**. The decision rule for the conventional Viterbi decoder is to select the sequence **u'** that maximizes the probability $P(\mathbf{y} \mid \mathbf{u})$.

We know that MAP decoder maximizes $P(\boldsymbol{u}|\boldsymbol{y})$; and ML decoder maximizes $P(\boldsymbol{y}|\boldsymbol{u})$. If the distribution of the source words **u** is uniform, then according to Baye's rule, we have:

$$P(\boldsymbol{u}|\boldsymbol{y}) \cdot P(\boldsymbol{y}) = P(\boldsymbol{y}|\boldsymbol{u}) \cdot P(\boldsymbol{u}) \tag{4.3}$$

then the two decoders are identical. In the conventional Viterbi algorithm, the term p(**u**), the *a priori* probability of the information sequence, is not known and is assumed the same for every sequence **u**. Thus the Viterbi decoder is a ML decoder.

Suppose that we have an input sequence **u** composed of L q-bit symbols $u_0, u_1, \ldots u_{L-1}$; Lq

is the decoding block length. Then the source bit stream to the encoder is,

$$\mathbf{u} = (u_0^0, u_0^1, ..., u_0^{(q-1)}, ..., u_{L-1}^0, u_{L-1}^1, ..., u_{L-1}^{(q-1)})$$

where u_i^j represents the j^{th} bit of the i^{th} source symbol. For a rate 1/n code, the encoder output bit sequence **x** and received bit sequence **y** will consist of Lqn bits:

$$\mathbf{x}=(x_0^{0(0)}, ..., x_0^{0(n-1)}, ..., x_{L-1}^{(q-1)(0)}, ..., x_{L-1}^{(q-1)(n-1)})$$

$$\mathbf{y}=(y_0^{0(0)}, ..., y_0^{0(n-1)}, ..., y_{L-1}^{(q-1)(0)}, ..., y_{L-1}^{(q-1)(n-1)})$$

where $x_i^{j(k)} \in \pm 1$ is the k^{th} bit of the channel encoded word which corresponds to the j^{th} bit of the i^{th} VQ source input symbol, and $y_i^{k(m)}$ is the corresponding received bit. Because the AWGN channel is a memoryless channel, that the noise process affecting a given bit in the received word **y** is independent of the noise process affecting all of the other received bits. The decision rule is to choose the **u** with the largest *path metric* computed by:

$$P(\boldsymbol{y}|\boldsymbol{u}) = \prod_{i=0}^{L-1} \prod_{j=0}^{q-1} P(y_i^j | u_i^j) \tag{4.4}$$

Then the log likelihood function known as the *branch metric* is calculated by:

$$\log P(y_i^j | u_i^j) = \sum_{k=0}^{n-1} \log P(y_i^{j(k)} | x_i^{j(k)}) \tag{4.5}$$

For the AWGN channel, we define E_s as the received energy per channel codeword bit, and the channel SNR = E_s/N_0. We have:

$$P(y_i^{j(k)} \mid x_i^{j(k)}) = \frac{1}{\sqrt{\pi \cdot N_0}} \cdot \exp\left\{-\frac{(y_i^{j(k)} - x_i^{j(k)})^2}{N_0}\right\} \tag{4.6}$$

then (4.5) can be expressed as:

$$\log P(y_i^j \mid u_i^j) = -\frac{1}{N_0} \cdot \sum_{k=0}^{n-1} (y_i^{j(k)} - x_i^{j(k)})^2 + C \tag{4.7}$$

where C is an irrelevant constant value.

In the Viterbi algorithm, the code trellis is used for the computation of $P(\mathbf{y} \mid \mathbf{u})$. A trellis diagram shows the encoder states as a function of the time. Each stage in the trellis corresponds an input bit. The state of the encoder is simply the contents of its shift registers. For an encoder with total memory m, the number of states is 2^m. For CCs, the initial state is normally chosen to be 0. After inputting the bit sequence of length L, m 0s are input to cause the last state of the sequence to be 0. The branches of the trellis diagram are labeled with the output bits corresponding to the associated state transitions. Among the paths terminating on a given node, the one which has the largest cumulative metric is stored, where the cumulative branch metric for a path is the sum of the branch metrics along a path. Therefore, at each time there will be 2^m survivor paths. After calculating for all of the stages, we trace back from the last node (zero state) to find the one path with the maximum-likelihood probability called *ML path*. According to this path, we can get the input bit on every stage. Then we obtain the decoded source symbols **u'** by combining consecutive q decoded bits into symbols. Figure 4.3 shows the trellis diagram for the rate 1/2 encoder of Figure 4.1.

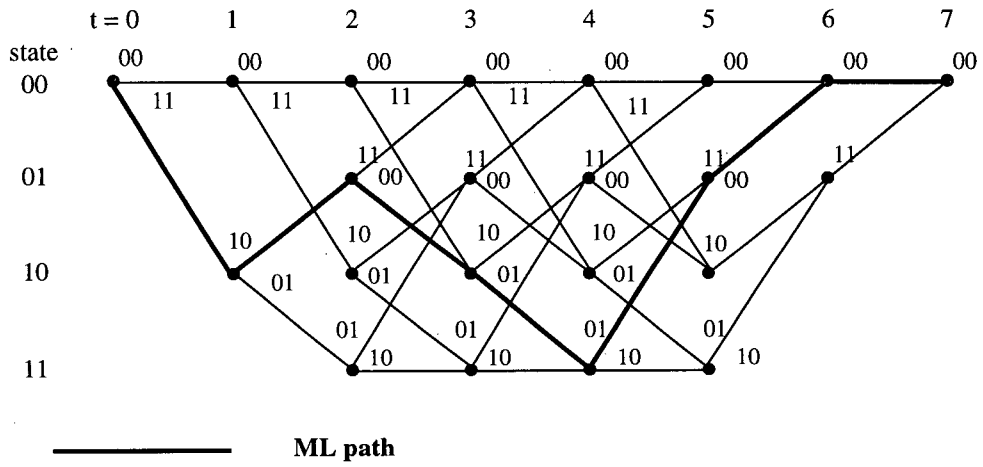


Figure 4.3: Rate 1/2, memory 2 CC trellis diagram

4.1.2 Trellis Merging MMACD Based on the Viterbi Algorithm

Suppose that we use a q-bit VQ source and its bit stream is encoded with a convolutional encoder, modulated and sent over an AWGN channel. Here we will apply a technique called *trellis merging* [26] which allows the correlation in the source codeword stream to be utilized when decoding a trellis-based channel code. The system block diagram is shown in Figure 4.4.

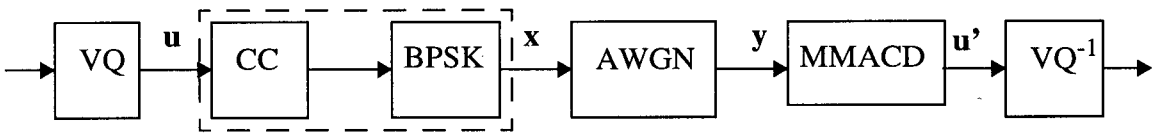


Figure 4.4: VQ /MMACD using trellis merging

Let the length of the source symbol sequence block be L. Each q-bit source symbol will be encoded into $q \cdot n$ channel bits by a rate 1/n convolutional code with memory m. Each block of n bits corresponding to one source information bit is called a *channel codeword.* If we block q

channel codewords into a single block, the block will correspond to a single source codeword (symbol). The trellis merging algorithm is shown in the following [26]:

We model the source by a first order, Q state Markov model, with $Q = 2^q$. The model is defined by the Q^2 *a priori* transition probabilities $P(u_i|u_{i-1})$. For a O(1) Markov sequence, it can be shown that

$$P(\boldsymbol{u}) = \prod_i P(u_i|u_{i-1}) \tag{4.8}$$

$i = 0, \ldots, L-1$. Since the channel is memoryless, we have

$$P(\boldsymbol{y}|\boldsymbol{u}) = \prod_{i=0}^{L-1} P(y_i|u_i) \tag{4.9}$$

where y_i is the block of q channel codewords which corresponds to the source symbol u_i. Therefore, using the source statistics to aid the conventional channel decoding, MMACD selects the estimate **u** that maximizes:

$$\prod_i P(y_i|u_i) \cdot P(u_i|u_{i-1}) \tag{4.10}$$

This decoding algorithm proceeds in the same manner as the conventional Viterbi algorithm except that it has a modified *branch metric*:

$$\log P(y_i|u_i) + \log P(u_i|u_{i-1}) \tag{4.11}$$

where

$$P(y_i|u_i) = \prod_{j=0}^{q-1} \prod_{k=0}^{n-1} P(y_i^{j(k)}|x_i^{j(k)}) \tag{4.12}$$

The first term in (4.11) is the usual term for ML decoding, and depends only upon the channel, while the second term depends on the source statistics. Inclusion of the source term in the branch metric causes the decoding to become MAP sequence decoding.

In the trellis diagram, this (blocking q channel codewords into one) corresponds to merging q branches into one which is connected by the two remaining end nodes directly. Then every stage of the new trellis corresponds to an input source symbol rather than an input bit. The decoding proceeds as the conventional Viterbi algorithm except that the trellis length will be reduced by a factor q, but the number of branches is increased by the factor 2^{q-1} and the branch metric will be modified. Therefore, the computational complexity of decoding in a merged trellis is in proportional to $2^{q-1}/q$. When q is increased from 1 to 2, the computational complexity does not increase. An example of trellis merging for the CC with memory m=2 and 2-bit VQ is shown in Figure 4.5.

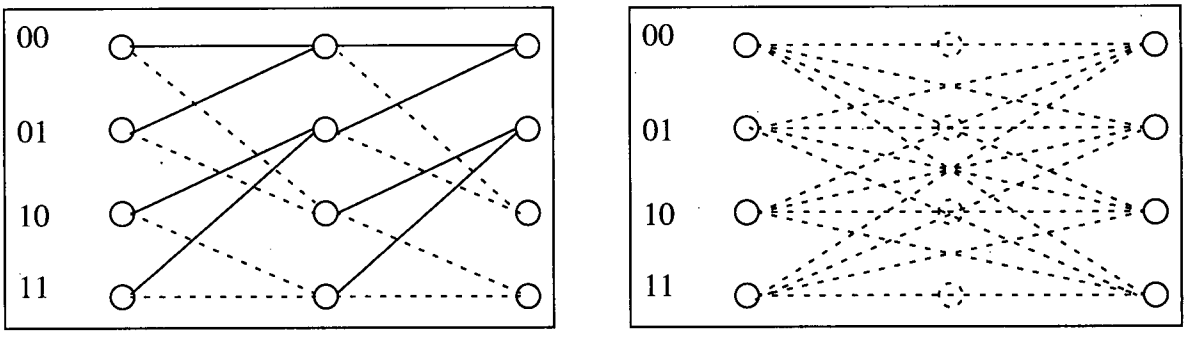


Figure 4.5: Regular Trellis and Two-Stage Merged Trellis of memory 2 CC

In Figure 4.6 we show the simulation result for transmission of 512 monochrome Lena encoded with a 3-bit VQ and a (3, 1, 5) convolutional code. The result shown in the figure tells us that Viterbi based trellis merging gives significant Markov model coding gain compared with conventional Viterbi decoding for the transmission of VQ image data over a AWGN channel. At the BER of 10^{-2} the coding gain due to use of the Markov Model is approximately 1.6dB. Similar to the MMAD without explicit channel coding, MMAD for the convolutional channel codes or MMACD gives less coding gain for the low error rate channel. Because in a clean channel, the channel term of the branch metric is strong enough to correct almost all channel errors; and the source term has a relatively weak influence on biasing the metric. But for a noisy channel, the source model is more effective at biasing the branch metric. We can see that the two curves will converge when the channel SNR becomes high. The comparison of the reconstructed Lena images at the channel SNR=-6dB and -4dB is shown in Figure 4.7. The improvement obtained by MMACD is about 9.44dB in RSNR when the channel SNR is -6dB.

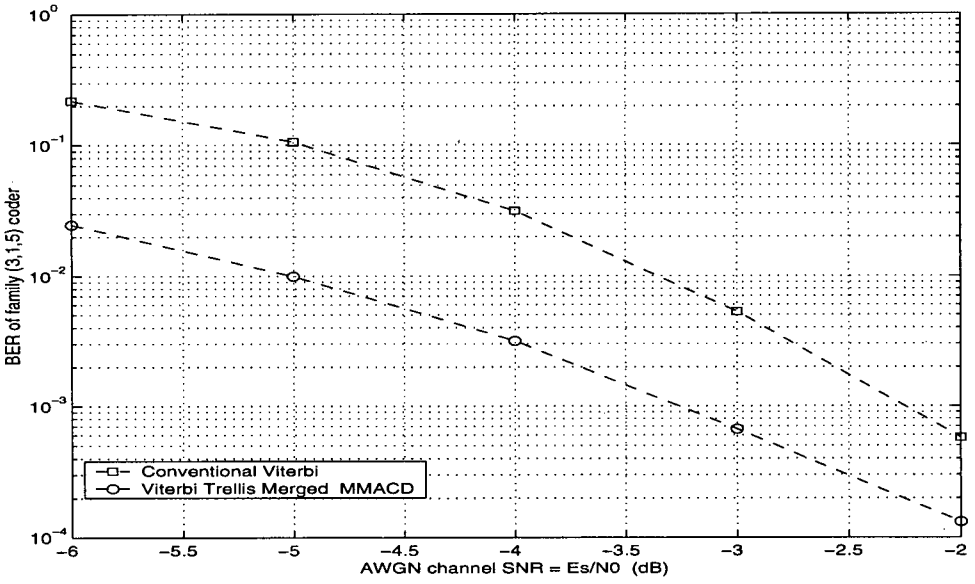


Figure 4.6: Viterbi decoding with and without an 8-State Markov Model for 3-bit VQ

Viterbi only at SNR = -6dB

RSNR = 9.21dB

Viterbi MMACD at SNR = -6dB

RSNR = 18.65dB

Viterbi only at SNR = -4dB

RSNR = 16.65dB

Viterbi MMACD at SNR = -4dB

RSNR = 22.20dB

Figure 4.7: Viterbi decoding with or without MMAD for 3-bit VQ over AWGN

Trellis merging is an efficient algorithm to make use of the VQ source information to aid the channel decoding, but it is necessary that the memory m of the convolutional code be greater or equal to the number q of bits per VQ symbol.

4.2 Concatenated MMACD Based on the BCJR Algorithm

In this section, we will be considering an alternate technique to trellis merging that does not have the restriction $q \leq m$. This would be useful for systems in which the source encoder produces long source words or in which a low memory channel code is used. This proposed decoding technique is called the concatenated MMACD. The conventional non-source aided channel decoder based on the BCJR algorithm decodes the received sequence in the first stage, then passes the *a posteriori* source bit probabilities to a cascaded MMAD without explicit channel coding which decodes the noise effected source bit stream using the Markov model in the second stage. Here we will discuss the BCJR algorithm for convolutional codes first.

4.2.1 The BCJR Algorithm for Convolutional Codes

In Chapter 3, we developed the BCJR decoding algorithm with SDF for the system without an explicit channel code. Here we will develop the BCJR [25] decoding algorithm for a convolutional code using the modified numerically stable version [8][24].

Assume we use a 1/n CC with memory m. The BCJR algorithm considers the source as a discrete-time finite-state Markov process. Suppose the q-bit VQ source is encoded bit-by-bit, and that the data bit stream is $\boldsymbol{u} = (u_1, u_2, \ldots, u_\tau)$; here u_i is a bit 0 or 1, τ is the decoding block length $\tau = L \cdot q + m$ and L is the total number of VQ source symbols in one block. Let $\boldsymbol{x} = (x_1, x_2, \ldots, x_\tau)$ be the encoded channel code word sequence, and let $\boldsymbol{y} = (y_1, y_2, \ldots, y_\tau)$ be the corresponding received sequence of a noisy channel transmission, $x_i = (x_i^{(0)}, x_i^{(1)}, \ldots, x_i^{(n-1)})$, $y_i = (y_i^{(0)}, y_i^{(1)}, \ldots, y_i^{(n-1)})$. Let S refer to the state of the Markov process. Unlike the BCJR decoding without explicit channel coding, the state here refers to the state of the convolutional

code instead of the VQ symbol value, which is the content of the shift registers. For a code with memory m, the total number of the states is 2^m.

We also know that the state of a convolutional code normally starts in the initial state $S_0 = 0$, and that the last state is forced to be $S_\tau = 0$ by adding a tail sequence of zeros. The objective of the BCJR convolutional decoding is to examine **y** and estimate the *a posteriori* bit probabilities i.e. $P(u_i|\mathbf{y})$, $0 \le i \le \tau$, which can be obtained from the *a posteriori* probabilities of the state P(S|**y**),

$$P(S_i|\mathbf{y}) = \frac{P(S_i, \mathbf{y})}{P(\mathbf{y})} = \frac{\lambda_i(S_i)}{\lambda_\tau(0)} \tag{4.13}$$

where $\lambda_i(s) = P(S_i = s, \mathbf{y})$. P(**y**) is a constant $P(\mathbf{y}) = \lambda_\tau(0)$. Let $S_i = (s_i^0, s_i^1, ..., s_i^{(m-1)})$, so the input bit $u_i = s_i^0$ at time i. Define A_i be the set of states S_i such that $s_i^0 = 0$ which means that the input bit at time i is 0. After getting *a posteriori* probabilities of the state P(S|**y**), calculate [25]:

$$P(u_i = 0|\mathbf{y}) = \sum_{S_i \in A_i} P(S_i|\mathbf{y}) = \frac{1}{\lambda_\tau(0)} \sum_{S_i \in A_i} \lambda_i(s) \tag{4.14}$$

Then the decoder can decode $u_i = 0$ if $P(u_i = 0|\mathbf{y}) \ge 0.5$; otherwise $u_i = 1$.

Similar to Chapter 3, λ is calculated by:

$$\lambda_i(s) = \alpha_i(s) \cdot \beta_i(s) \tag{4.15}$$

The calculation of α and β uses the same equations as (3.15) and (3.16), except that s or s' is now

the state of the convolutional code, $0 \le s, s' < 2^m$ instead of the VQ symbol value. For the BCJR convolutional decoding, the initial boundary conditions for α become $\alpha_0(0) = 1$ and $\alpha_0(s \neq 0) = 0$, corresponding to the encoder initial state 0; and the final boundary conditions for β are $\beta_\tau(0) = 1$ and $\beta_\tau(s \neq 0) = 0$, corresponding to the encoder ending in state 0.

The transition probability $\gamma_i(s', s)$ is defined as $\gamma_i(s', s) = P(s_i = s, y_i | s_{i-1} = s')$ and is easily shown to be

$$\gamma_i(s', s) = P(u_i)P(y_i|x_i) \tag{4.16}$$

The first term $P(u_i)$ is the *a priori* probability of the bit u_i that causes the transition $(s_{i-1} = s') \rightarrow (s_i = s)$ which results in the encoder output x_i. Now it assumes all input sequences equally likely for $i \le Lq$. For $i > Lq$, the input to the CC is m 0s, so $P(u_i)$ is equal to 1. The second term $P(y_i|x_i)$ is the probability that code word y_i is received given that code word x_i is sent and this depends on the channel. Because the input bit $u_i \in 0, 1$, then we have:

$$\gamma_i(s', s) = \begin{cases} \frac{1}{2} \cdot P(y_i|x_i) & s' \overset{0/1}{\rightarrow} s, i \le Lq \\ P(y_i|x_i) & s' \overset{0}{\rightarrow} s, i > Lq \\ 0 & else \end{cases} \tag{4.17}$$

where $s' \overset{u}{\rightarrow} s$ means that the transition $(s_{i-1} = s') \overset{u_i}{\rightarrow} (s_i = s)$ exists, because s_i is decided by s_{i-1} and input bit u_i. For a memoryless AWGN channel:

$$P(y_i|x_i) = \prod_{k=0}^{n-1} P(y_i^{(k)}|x_i^{(k)}) \tag{4.18}$$

For the bit transition probability of AWGN channel see (4.6).

Using the stable renormalization version, we can see that the complexity of the computation will increase with the state number. Due to the relationship between the previous state, the current state and the current input bit, we can simplify the calculation of α and β in (3.15) and (3.16) by:

$$\alpha_i(s) = \frac{\sum_b \alpha_{i-1}(t'(s,b)) \cdot \gamma_i(t'(s,b), s)}{\sum_b \sum_{s'} \alpha_{i-1}(s') \cdot \gamma_i(s', t(s',b))} \tag{4.19}$$

$$\beta_i(s) = \frac{\sum_b \beta_{i+1}(t(s,b)) \cdot \gamma_{i+1}(s, t(s,b))}{\sum_b \sum_{s'} \beta_i(s') \cdot \gamma_{i+1}(s', t(s',b))} \tag{4.20}$$

$t'(s, b)$ is the function to get the previous state value if we know the current state value s and the current input bit b. $t(s', b)$ is the function to get the current state value given the previous state value and the current input bit b = 0 or 1. There are 2^m states in total. Thus, using (4.19) and (4.20) we can reduce the computation by the factor of 2^{m-1}.

The BER of this algorithm is shown in Figure 4.9. Compared with Figure 4.6, we see that the BCJR decoding algorithm has the same error protection ability as the Viterbi algorithm. The computation of the algorithm becomes considerable when the constraint length of the convolutional code becomes large. But the advantage of this algorithm is that it can provide the soft

output of *a posteriori* probability (*app*); we will exploit this character to propose a concatenated MMACD which does not have the restriction $q \leq m$ in next section.

On the other hand, we can apply the trellis merging method of Section 4.1.2 to a soft-output trellis merged (BCJR trellis merged) MMACD. It would be useful for constructing source-aided turbo decoding systems [31][8], and we used it to check the concatenated MMACD that we will propose. We find that the BCJR trellis merged decoder also has the same BER as the Viterbi trellis merged decoder. The detailed information about the BCJR trellis merged decoding algorithm is shown in the Appendix.

4.2.2 A Concatenated MMACD Based on the BCJR Algorithm

It is true that these trellis merging techniques combine the channel transition term and the source statistics term together in the calculation of the branch metric. However, these trellis merged MMACDs have to satisfy the condition that $q \leq m$. Can we calculate the channel term first and combine the source term later? Thus, in the case that $q \geq m$, we could have an alternative way to take advantage of the Markov model.

Consequently we propose a *concatenated* MMACD which is composed of two decoders. The first decoder is a conventional channel decoder and the second decoder is the Viterbi based MMAD without explicit channel coding. First, the conventional channel decoder decodes the received sequence **y**, then passes its output into the MMAD which can bias the decision using the source information. The output of the first decoder can be hard output or soft output. Our analysis and simulations showed that the second decoder can not give help if the first decoder gives the hard output decision. Therefore, we would like to use a decoder which can generate soft output

information.

The BCJR algorithm is optimum in the sense that it estimates the exact value for a soft output (*app* of source bit) $P(u_i^j|\mathbf{y})$, where u_i^j is the j^{th} bit of the i^{th} source symbol. Because of this, we would like to use the BCJR decoder as the first decoder in the concatenated MMACD. The structure of a concatenated MMACD is in Figure 4.8.

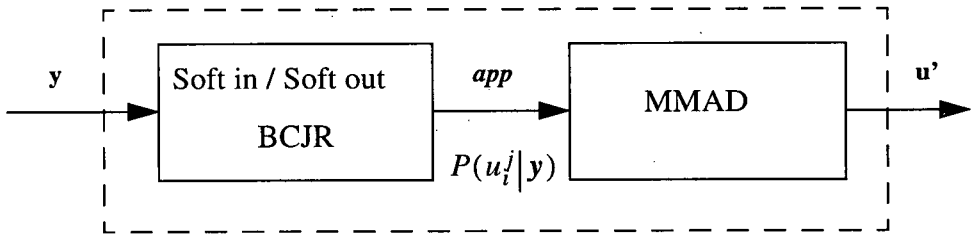


Figure 4.8: Concatenated MMACD

For MMAD without explicit channel coding, the branch metric is calculated by $\log p(u_i|\hat{u}_i) + \log P(u_i|u_{i-1})$, where u_i is the VQ source symbol, and $\hat{u}_i$ is the received symbol. For q-bit VQ transmission with a convolutional code over the memoryless AWGN channel, the BCJR decoder gives the *app* of a input bit $P(u_i^j|\mathbf{y})$, j=0, 1, ..., q-1, and **y** is the received sequence.

The MMAD without explicit channel coding in the proposed concatenated MMACD decoder uses the channel transition probabilities for the branch metric.

$$P(u_i|\hat{u}_i) = \prod_{j=0}^{q-1} P(u_i^j|\mathbf{y}) \tag{4.21}$$

which is based on $P((u_t^j|\hat{u}_t^j) = P(u_t^j|y))$, and that the q conditional bit probabilities $P(u_t^j|\hat{u}_t^j)$ are independent.

The simulation result shown in Figure 4.9 shows that this decoding gives some improvement over the conventional BCJR or Viterbi decoding, but much less improvement over the trellis merged decoding. To investigate this further, we use the BCJR trellis merged decoder (see the Appendix) without using the source statistics in the γ calculation to generate the *app* of the input symbol $P(u_t|y)$ instead of the *app* of the input bit $P(u_t^j|y)$; and we pass this soft output to the MMAD. From the simulation result in Figure 4.9, we find that the performance using the *app* of the input bit $P(u_t^j|y)$ is different from the one using the *app* of the input symbol $P(u_t|y)$, which means that conditional bit probabilities $P(u_t^j|\hat{u}_t^j)$ are not quite independent. Also, the BCJR concatenated decoder using the *app* of symbol cannot outperform the trellis merging MMAD.

Although there is no loss of information when we pass the soft output of the convolutional decoder to the MMAD, we do find a performance penalty with respect to the trellis merged decoder. This is because in the trellis merged decoder the source information is used to bias the decision based on the received channel data. In the concatenated decoder, the convolutional decoder increases the reliability of the received channel data. The source information in the second decoding stage cannot bias a decision based on this more reliable information as effectively.

The BCJR is not computationally efficient for large memory convolutional codes, also that the computational complexity of the MMAD without explicit channel codes grows exponentially

with the number q of bits per VQ symbol. Therefore this decoding method is practical only for short constraint lengths and long VQ symbols. It should only be used when the restriction $q \leq m$ is not satisfied so that the trellis merging cannot be applied.

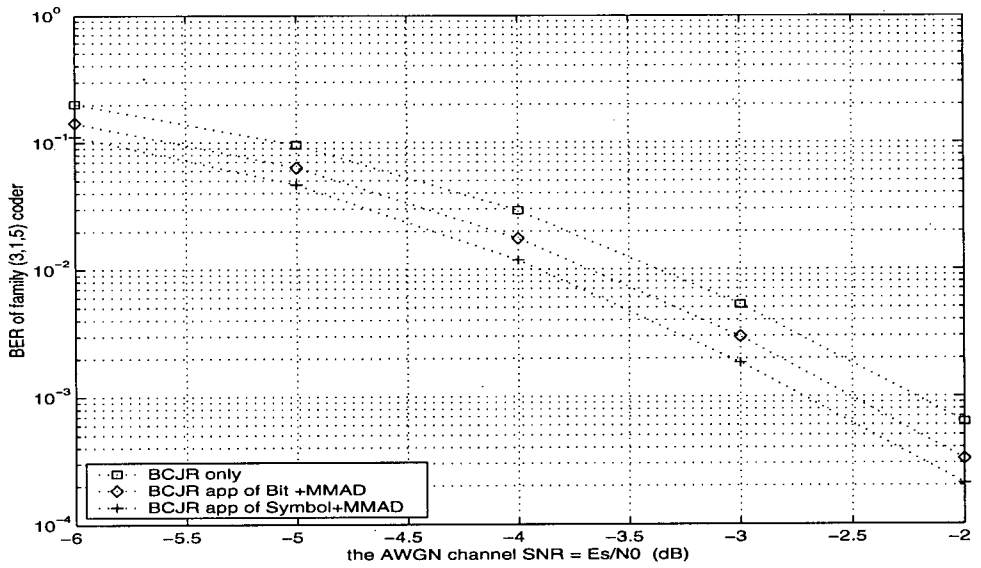


Figure 4.9: BCJR and Concatenated MMACDs

From the above discussion, it can be concluded that the Viterbi based trellis merging is the most efficient method with which to take advantage of the Markov source model. In the following, we will use the Viterbi based decoder with trellis merging to construct in the rate allocated systems.

Chapter 5 Rate Allocated Image Transmission Systems

In this chapter, we will continue by investigating the problem of rate allocation between the source encoder and the channel encoder for VQ image transmission over the AWGN channel, given that the system information rate and channel transmission rate are fixed. Here we will use Punctured Convolutional Codes (PCCs) as the channel codes. We have shown that MMAD convolutional decoders give little improvement at low error rates. Some authors including the present, found that at most times the bit error probability becomes very low when the source and channel rate is optimally allocated [3]. That would indicate that MMAD may be not useful for the rate allocated systems. However, MMAD does extend the SNR rage over which a channel code corrects most of the channel errors; and we will find that this does impact the rate allocation.

In this chapter, we will compare the two rate allocated systems: system A is the conventional rate allocated system without MMAD; system B is the proposed rate allocated system with MMAD. We will apply the model simulation scheme to determine the optimal rate allocation for the two systems and examine the difference of the optimal rate allocation between the two systems. Performance comparisons are made in terms of the end-to-end RSNR as a function of channel bit SNR. We conclude the study with comparative analysis of the sensitivity to channel mismatch of the two systems.

First, we will discuss the system model and look for good PCCs for our systems. The distortion calculation scheme for determining the optimal rate allocation is discussed as well.

5.1 System Model

Figure 5.1 shows the block diagram of the end-to-end communication system.

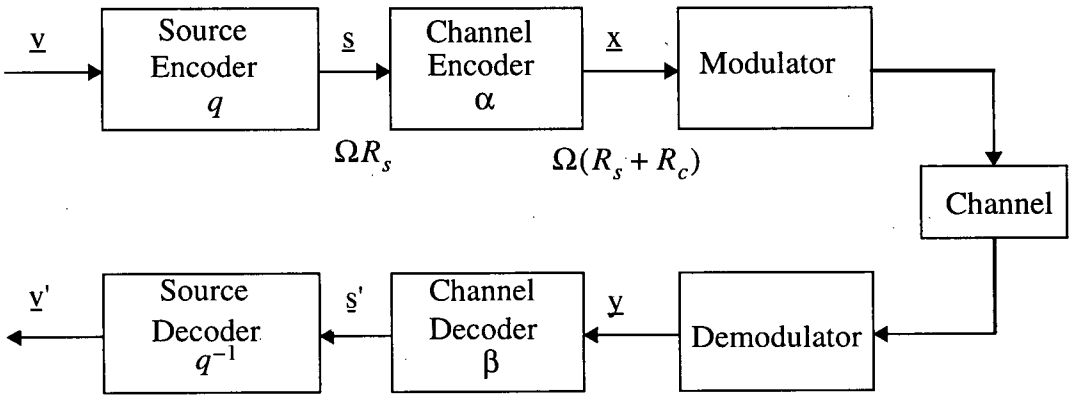


Figure 5.1: System model

Let R_s be the *source rate*, which means *bits per pixel* (bpp), for the Ω-dimension fixed rate VQ – each encoded symbol is composed of ΩR_s bits. The channel encoder will add the controlled redundant bits for these source encoded bits, which will result in $\Omega(R_s + R_c)$ bpp. Define the *total rate* $R = R_s + R_c$, where R_c is the *channel rate*. Then, the *channel coding rate* r_c is given by:

$$r_c = \frac{R_s}{R_s + R_c} \tag{5.1}$$

The channel coded bits (channel symbols) are then modulated into the channel symbols and transmitted over an AWGN channel at a rate of one channel symbol per T second, where T is the *channel symbol time*. Define the *information transmission rate* I_r of the system as the number of the source symbols (pixels) transmitted per second. Given the *channel symbol transmission rate* 1/T, we have:

$$I_r = \frac{1}{T \cdot (R_s + R_c)} \tag{5.2}$$

Thus given a fixed source information rate I_r and a fixed channel symbol transmission rate $1/T$, the total rate $R = R_s + R_c$ is fixed too. In that case the rate allocation problem is reduced to finding the best combination of R_s and R_c for a given fixed R.

We suppose a discrete-time, real-valued, stationary source encoded with the VQ source coder and the PCC channel coder. Each possible Ω-dimensional source vector $\underline{v}$ is mapped to a binary code (index) $\underline{s}$ with fixed length. After the channel coding, the binary code $\underline{s}$ becomes the channel symbol vector $\underline{x}$. These channel symbols are modulated by BPSK and are sent to the AWGN channel. Due to the channel noise, the received channel symbol vector $\underline{y}$ might not be the same as the transmitted symbol vector $\underline{x}$. After passing into the channel decoder, which may correct some of channel errors, one obtains the decoded binary code $\underline{s}'$ corresponding to the original binary code $\underline{s}$. Finally, the noisy reproduction $\underline{v}'$ is obtained after $\underline{s}'$ is passed through the source decoder.

We calculate the total end-to-end distortion of the system as the squared error between the source vector $\underline{v}$ and the source reconstructed vector $\underline{v}'$: $d(\underline{v}, \underline{v}') = \|\underline{v} - \underline{v}'\|^2$. The total distortion is normally composed of the channel errors or distortion and the source quantization errors or distortion. Given a fixed *total rate* R, if one uses the high R_s, then there are fewer bits used for R_c. This will occur at the high channel code rate (large r_c) at the expense of the high probability of bit error. In that case, the source coding or quantization distortion is small, but since the channel decoding error probability is relatively high, the end-to-end distortion may also be unacceptably high. On the other hand, if we give more bits for the channel rate R_c, the low channel coding rate r_c occurs at the cost of low source coding resolution. In this case, the channel decoding error prob-

ability is small, but the source coding distortion is relatively high; thus again possibly yielding a large total distortion.

Between those two extremes there should exist the optimum combination of the source rate R_s and the channel rate R_c which minimizes the distortion. The optimal rate allocation for the conventional system depends on the channel SNR = E_s/N_0, the source statistics, channel coding and modulation [3]. Usually, R_s corresponding to the optimal bit allocation, will increase when the channel SNR is increased. Because the less noise there is in the channel, the less bits are needed for error protection, and the more bits can be used by the source code.

For the various source rates R_s, the system needs the various rate channel codes r_c. We choose a family of PCCs as the variable rate channel codes that are generated from the same convolutional code. This allows relatively simple implementation of the transmitter and the receiver.

For our systems, we use a 4-dimensional VQ, and consider the total rate R = 2bpp over the range of channel SNRs $-3dB \le E_s/N_0 \le 3dB$. We choose these values for the parameters because, we will see from the following numerical results, that there is little distortion reduction for R_s > 2bpp or E_0/N_0 > 3dB. For each R_s $0 \le R_s \le R$, there is a corresponding $R_c = R - R_s$. We assume all the redundant bits are used for the channel code, and that the codes are equal weight channel codes, which means that there is the same error protection for each bit of the source encoded data. Previous research has found that there is not much difference between using equal weight channel codes and using unequal weight channel codes when the system is optimally rate allocated [3][32].

For example, if we use a 3-bit 4-dimensional VQ, the source rate R_s is 0.75 bpp, and using (5.1), the channel coding rate required r_c is 3/8. All the variable rate codes that our systems require are shown in Table 5.1. In the next section we will discuss PCCs and the way to search for the best PCCs we need.

Table 5.1: Combination of R_s and R_c

R_s	q-bit VQ	R_c	r_c
0.25	1	1.75	1/8
0.50	2	1.50	2/8
0.75	3	1.25	3/8
1.00	4	1.00	4/8
1.25	5	0.75	5/8
1.50	6	0.50	6/8
1.75	7	0.25	7/8
2	8	0	1

5.2 Punctured Convolutional Codes (PCCs)

5.2.1 PCC Encoding and Decoding

PCCs [31]-[37] have drawn considerable attention in recent years, because one would like to build a system with more flexible rate channel codes when the channel states or characteristics change very often. PCCs were first introduced by Cain, Clark, and Geist [34]. They used a fixed CC (n, k, m) to generate high-rate PCCs by periodically deleting (puncturing) bits with period p from one or more of the encoder output streams. The resulting high-rate codes are defined by the low-rate code used, called the *original code*, and by the $n \times p$ *perforation matrixes* which show the pattern (the number and specific positions) of the punctured bits. For example, for the rate 1/2 code (2, 1, 2) in Figure 3.2, the *perforation matrix* of rate 2/3 punctured code is given by

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

where 0 means the corresponding bit symbol will not be transmitted, and 1 means the corresponding bit symbol will be kept to transmit. The trellis of the rate 2/3 punctured code from the 1/2 original code is shown in Figure 5.2, where the symbol x indicates the punctured bit. Every period, with 2 bits input, the fourth coded bit is not transmitted and 3 bits are transmitted instead of 4 bits, so the rate of this PCC is 2/3.

The decoding of PCC is similar to the Viterbi decoding algorithm for the original code. The decoding is performed on the trellis of the original low-rate code by inserting a dummy data bit into the position corresponding to the punctured bit symbol. In the decoding process this dummy data bit is discarded by assigning it the same metric value (usually zero) regardless of the code bit symbol, 0 or 1 [33].

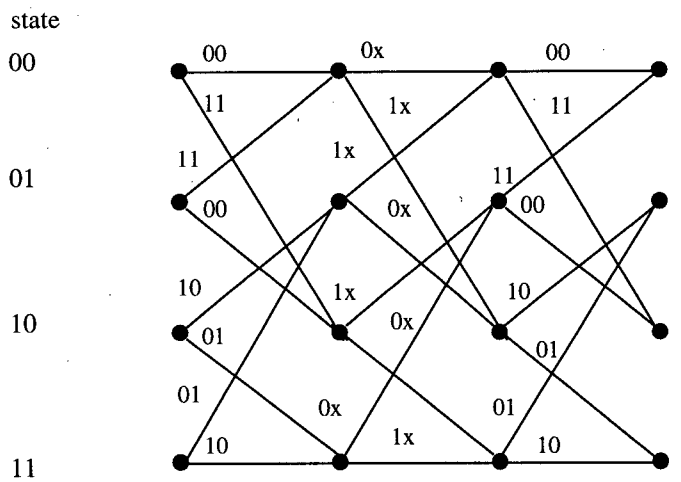


Figure 5.2: Trellis for R = 2/3 PCC based on (2, 1, 2) code

5.2.2 Search for Good PCCs

Let the number of ones in the perforation matrix be *l*. The rate of generated PCC from a *1/n* original code with puncturing period p will be $r_c = \frac{p}{l}$. By changing the value of *p* or *l*, variable-rate codes of all punctured rates of interest are readily obtained from the same low-rate encoder.

Unlike the search for the usual convolutional codes, the search for punctured codes is often based on intuition and trial rather than on a strict mathematical construction [36]. We choose the (3, 1, 8) convolutional code as the original code with the generator polynomial $g_1=557_8$, $g_2=663_8$, and $g_3=771_8$ because it has powerful error protection ability. Our research is based on the intuition that "good codes generate good codes". We apply the following process to look for the best variable rate PCC from this 1/3 rate original code:

In practice we only consider PCCs at the reasonable allocation $0.75 \le R_s \le 2$ due to the corresponding VQs' high source distortion when R_s is small. For $R_s = 2$ or $r_c = 1$, all bits are allocated to the source encoder, which means no channel coding. So the variable channel coding rates required for our system are 3/8, 4/8, 5/8, 6/8, and 7/8. To generate a punctured code with rate r_c = j/8, $3 \le j \le 7$, we choose the *perforation matrix* period p = j. With j input bits in a period, the encoder should have 8 output bits for the rate j/8 code. Thus in the pattern of the *perforation matrix* $3 \times j$, $3 \le j \le 7$, 8 positions are assigned to 1, and the other positions are assigned to 0. Consequently there are $C^8_{3\times j}$ possible *perforation matrixes* for the rate j/8 codes, where *C* is the combination function; in another words, we have $C^8_{3\times j}$ possible codes with rate j/8.

Then, among these same rate codes, we will select the best code which has the best performance over the AWGN channel. It is known that an upper bound on the bit error probability P_b of a PCC with a period p is given by [34] [37]:

$$P_b \leq \frac{1}{p} \sum_{i=d_{free}}^{\infty} c_i \cdot P_i \tag{5.3}$$

where d_{free} is the free distance of the code and c_i is the total number of bit errors in all the incorrect paths of Hamming weight i, $i \geq d_{free}$. P_i is the probability that one such incorrect path is selected in the Viterbi decoding process. For the AWGN channel and BPSK modulation,

$$P_i = Q\left(\sqrt{2 i r_c \frac{E_b}{N_0}}\right) \tag{5.4}$$

where r_c is the channel coding rate, and E_b / N_0 is the energy per bit-to-noise density ratio, $E_b = \frac{E_s}{r_c}$.

Thus, the criterion for searching the best punctured code is to select one with the maximum d_{free} and the minimum c_i. After obtaining all the possible *perforation matrixes*, we calculate the corresponding d_{free} and c_i of each code. The codes with maximum d_{free} are picked out first, then among these codes, we select one with the minimum c_i as the best code. Of course, we have to check if the resulting code is not *catastrophic code* which may translate a small number of errors in the received codeword to an unlimited number of data error.

In the above way, we obtain the required best variable-rate PCCs, which have the follow-

ing *perforation matrixes*:

$$\text{3/8 PCC: } P = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}, \text{4/8 PCC: } P = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 \end{bmatrix}, \text{5/8 PCC: } P = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$$\text{6/8 PCC: } P = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 \end{bmatrix}, \text{7/8 PCC: } P = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

Their BER performance is shown in Figure 4.3.

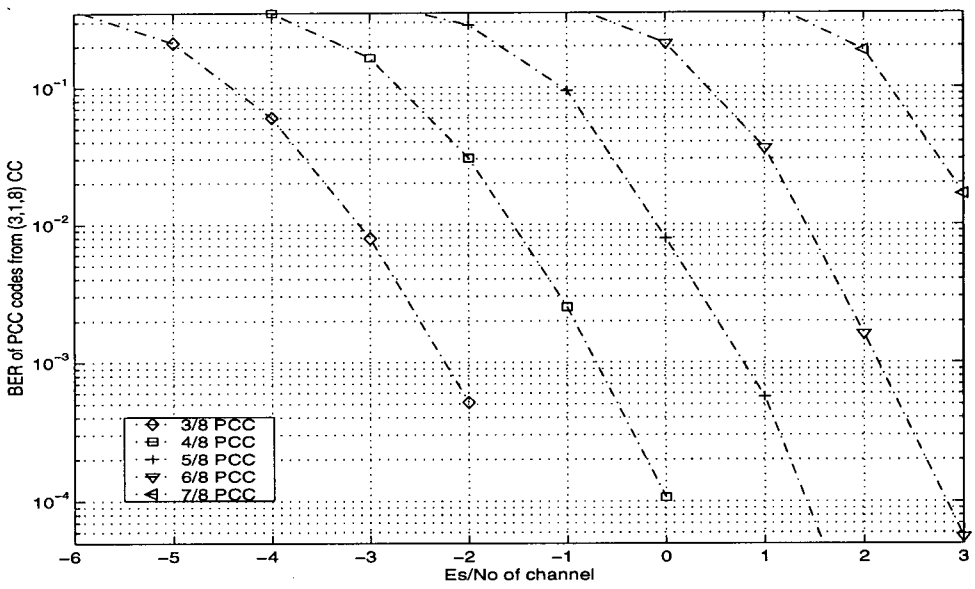


Figure 5.3: BER performance of the various rate PCCs based on (3, 1, 8) code

In a practical encoding implementation, the original channel encoder is followed by a puncturing table, which is used to store the *perforation matrixes* corresponding to the different rates. The output streams of the original coder are punctured according to the puncturing table

associated with the selected rate.

5.2.3 MMAD for PCCs

After obtaining the variable rate codes we need, and their BER performance, similarly as in Chapter 4 for CC decoding, we apply the MMAD technique to the PCC decoding. In this chapter, we only consider the Viterbi based trellis merging MMAD technique for our rate allocated systems because the memory of our channel codes is greater than the longest source code word length. The BER performance of the 3/8 PCC decoding with and without MMAD for the 3-bit VQ encoded 512 by 512 Lena image is shown in Figure 5.4. We see that MMAD improves the conventional PCC Viterbi decoding at noisy channel by taking advantage of the source statistics. When the channel SNR becomes high, the performance of two decoders converges.

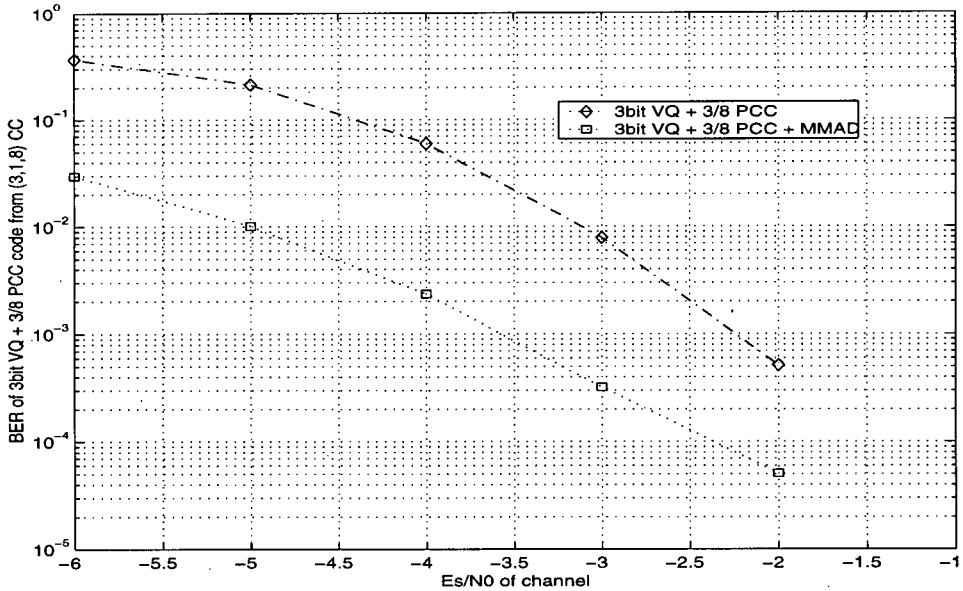


Figure 5.4: BER performance of PCC decoding with or without MMAD, 512 Lena

For the latter rate allocated systems, we simulate the BER of the various rate PCCs using

MMAD with the corresponding source rate VQ for the Lena image. The results for the R=2 family are shown in Figure 5.5.

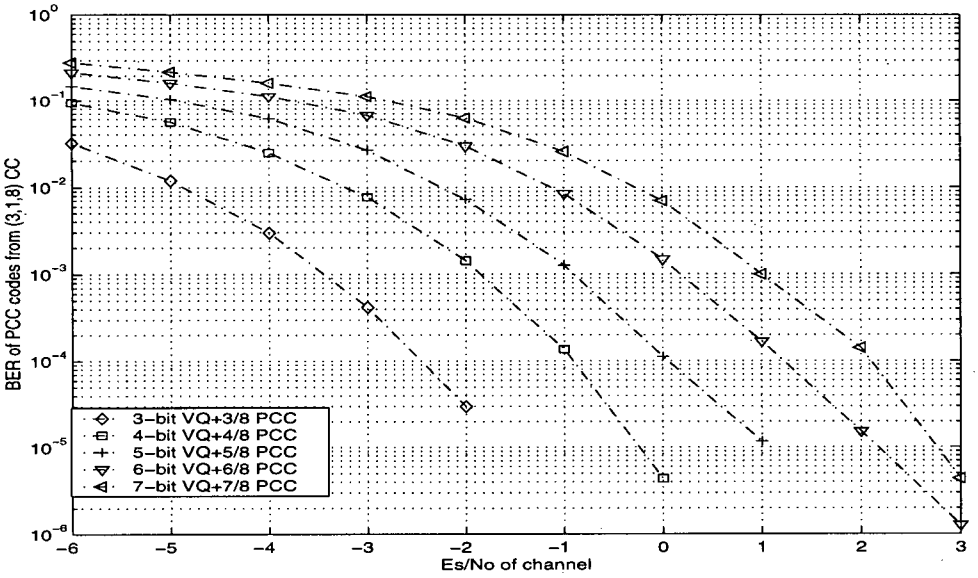


Figure 5.5: BER performance of VQ/PCC family at R=2 with MMAD, 512 Lena

Because MMAD uses the source statistics to aid the channel decoding, the BER performance of PCC using MMAD should be a function of the source statistics which will be related to the source characteristics and the source encoder. We use a discreet event simulation to obtain the BER of the 3-bit VQ with 3/8 PCC using MMAD for the 512 by 512 Lena image and the 256 by 256 Sena image. In Figure 5.6, as expected, we see that PCC plus MMAD for the Sena image is different from that for the Lena image.

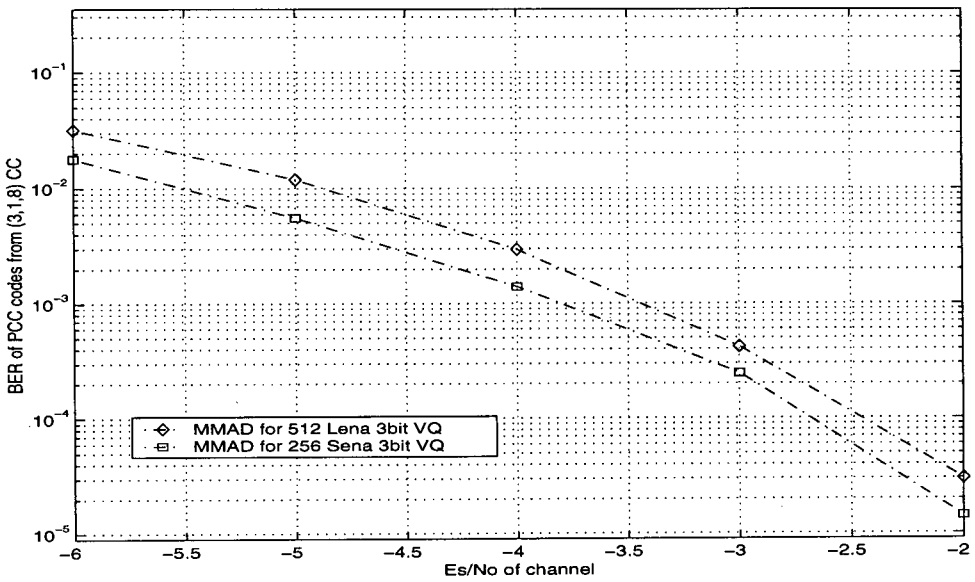


Figure 5.6: BER performance of PCC with MMAD for different images

5.3 Distortion Calculation for Rate Allocated Systems

The calculation of the end-to-end distortion is used to determine the optimal rate allocation for rate allocated systems [3][38], which select the rate with the minimum distortion as the optimal rate.

For the system in Figure 5.1, we define the total distortion D_t of the system as $D_t = E[d(\underline{V}, \underline{V}')]$, where $\underline{V}$ is the source input vector, and its reproduction $\underline{V}'$ is the output of the receiver. When the channel errors are independent of the source, and the source encoder is designed to meet the centroid condition, which means the reproduction codevector corresponding to the binary code $\underline{s}$ is the centroid of the quantization region $R_{\underline{s}}$: $\eta(\underline{s}) = E[\underline{V}|\underline{v} \in R_{\underline{s}}]$, the total distortion D_t is the sum of source distortion D_s and channel distortion D_c [3][38]; therefore:

$$D_t = D_s + D_c \tag{5.5}$$

$$D_s = E_v[\|\underline{V} - \eta(q(\underline{V}))\|^2] \tag{5.6}$$

$$D_c = E_{\underline{s},\underline{s}'}[\|\eta(\underline{s}) - q^{-1}(\underline{s}')\|^2] \tag{5.7}$$

where D_s is the distortion-rate performance of the vector quantizer and is known or precomputed for the chosen VQ. D_c is determined by the channel encoder, decoder and source statistics. We have:

$$D_c = \sum_{\underline{s}} \sum_{\underline{s}'} P(\underline{s}) P(\underline{s}'|\underline{s}) \|\eta(\underline{s}) - q^{-1}(\underline{s}')\|^2 \tag{5.8}$$

where P(s) is the source coded data statistics which can be precomputed if the source encoder is fixed. $P(\underline{s}'|\underline{s})$ is called the *index crossover probability*, which is a function of the channel encoding, modulation, channel decoding and the channel character. When the bit errors of the channel decoder output are independent, the bit error probability P_b, the index crossover probability is calculated as

$$P(\underline{s}'|\underline{s}) = P_b^n \cdot (1 - P_b)^{\Omega R_s - n} \tag{5.9}$$

where n is the Hamming distance between index $\underline{s}$ and index $\underline{s}'$.

In addition to $P(\underline{s}'|\underline{s})$, we need to know the source encoder codebook; using (5.8), D_c is easily calculated. Thus, having D_s and D_c, the total distortion D_t is obtained staightforwardly. After calculating all possible D_t for each possible rate, we choose the R_s with the minimum D_t as the optimal rate which is sent to the transmitter.

We can see that the bit error probability P_b of the channel coding is an essential term which may be more complicated for the calculation of the total distortion (5.5)-(5.9) than the other terms.

For the system using PCCs, the bit error probability P_b of the PCC Viterbi decoding is not related to the source statistics and can be efficiently computed, so this scheme was used to determine the optimal rate allocation for the rate allocated systems using the Viterbi decoding only [3][38].

When considering the system with MMAD, this method can be also used to determine the rate allocation for a MMAD system. However since the decoding decision of PCC with MMAD is influenced by the source statistics and source coders, it is difficult to get the bit error probability P_b. Because of this, it will be computationally complicated to determine the optimal rate allocation for the MMAD system.

Because we do not want to worry about the calculation of P_b and its preciseness, we choose an alternative scheme, model simulation, to determine the optimal rate allocation for our systems. Our optimality criterion is to maximize the average end-to-end RSNR. At the same channel SNR, we simulate every model in the system and calculate the end-to-end RSNR for each R_s; the rate with the maximum RSNR is selected as the optimal rate.

It is true that at the rate with the smallest distortion, the $RSNR = 10 \cdot \log\left(\frac{\sigma^2}{D}\right)$ will be the largest, where D is the total end-to-end distortion, and σ^2 is the image variance. Therefore, the two schemes are actually same for calculating the optimal rate allocation. Compared with the dis-

tortion calculation, the model simulation scheme can be done off-line, too, with the similar simulation complexity. The result of the model simulation scheme is more straightforward. In the next sections we will use this scheme to determine the optimal rate allocation for non-MMAD and MMAD systems. First we will simulate the non-MMAD system.

5.4 Optimal Rate Allocation without MMAD

According to Figure 5.1, we simulate each model in the following way: with all the required PCCs, we combine the VQ source encoder, the PCC channel encoder, the AWGN channel with the corresponding PCC decoder and VQ decoder. Given a certain channel SNR, if one selects R_s as the VQ source encoding rate, the corresponding PCC channel coding rate should be $r_c = R_s/R$. At the receiver, the final RSNR is calculated. Then, according to the result of RSNRs, we select the R_s with the largest RSNR and decree it to be the optimal rate for this channel SNR. This process will be repeated for each R_s $0.75 \le R_s \le 2$. Then we repeat the above process for other channel SNRs. Then we plot these results on one graph.

The result of the optimal rate allocation for the 512 by 512 Lena is shown in Figure 5.7. It is seen from this figure that the improper choice of R_s can reduce the RSNR by 10~20 dB. As expected, with the channel SNR increasing, the value of the optimal rate R_s is increased, too, since less bits are needed for the channel coding when the channel becomes cleaner. The optimal rate result obtained by the model simulation scheme is the same as that obtained by the distortion calculation in [3].

The solid line with the symbol "+" shows the simulation result of the transmission of the rate R_s VQ encoded data only over the noiseless channel. We will use it to compare with the per-

formance of VQ with PCC over the AWGN channel at the different channel SNRs. We see that RSNR increases monotonically as the channel SNR increases. It is a fact that with the certain rate R_S VQ, when all the channel errors are corrected by the corresponding rate PCC, the final RSNR value should be equal to the RSNR value corresponding to the same rate R_S on the solid line. In these cases, the total distortion is contributed by the source distortion.

We choose the rate with the highest RSNR as the optimal rate for the certain channel SNR. The figure shows that the optimal bit rate R_S is often at the point that the RSNR is approximately the same as the value on the solid line associated with the same rate R_S, except in the worst channel situation SNR = -3dB. This also means that most of the total distortion at the optimal rate allocation (R_S, R_C) is determined by the source distortion. It is due to the fact that at the optimal rate allocation, the channel bit error-probabilities of the PCC decoding are approximately zero, that channel errors are almost all corrected. Refer to the bit error probability or bit error rate (BER) of the PCC decoders based on the Viterbi algorithm in Figure 5.3.

Our simulation found the deviation, σ, of the RSNR result to be less than 0.2 dB, by using 10 different seed numbers to retransmit the same image. So to save simulation time, the reported RSNRs for the curves of this chapter are obtained by using only one seed number.

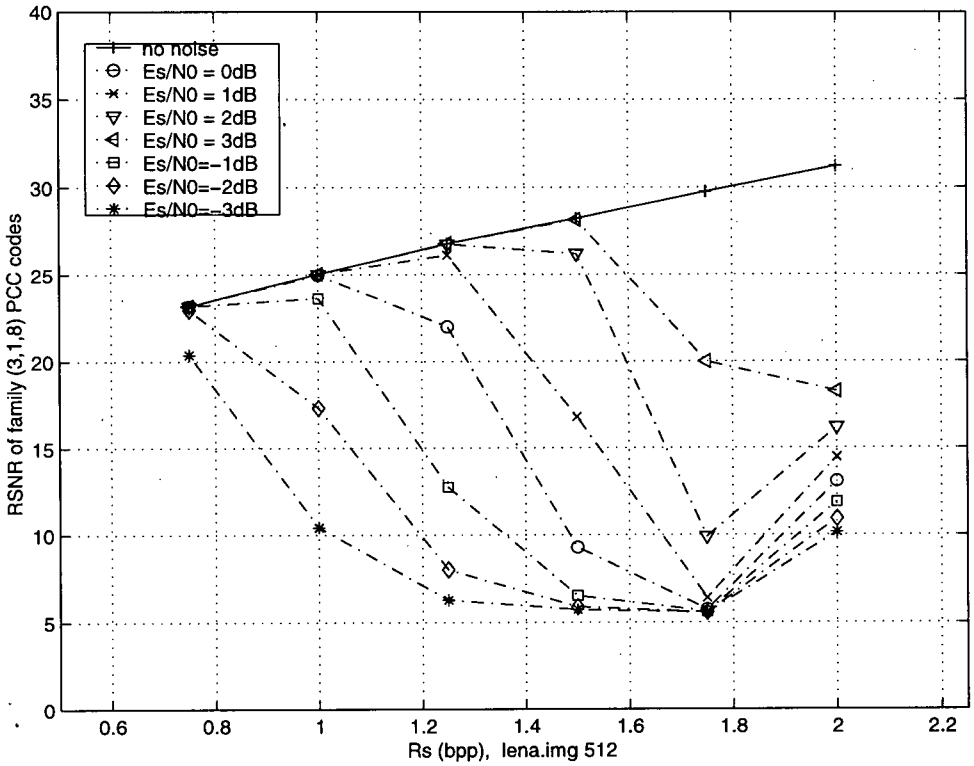


Figure 5.7: Optimal rate allocation for the conventional system without MMAD

Following this, we select other images with different variance, correlation and size, and apply the non-MMAD system to see if the optimal rate allocation is changed with image. From our results in Table 5.2, the optimal rate allocation turns out to be fairly independent of the image. We find that in the few cases, when the optimal allocation differs from our reported image independent allocation, that the price in RSNR is very small (always less than 0.5 dB). The point of optimal rate allocation always is the allocation where most the channel errors are corrected and this does not depend on the image. It is known that the BER performance of the PCC decoding without MMAD does not relate to the source statistics, but to the channel codes' character and channel SNR.

Table 5.2: Optimal R_s for non-MMAD systems for the various images

E_s/N_0 (dB)	Lena 512	Pepper512	Couple 512	Lena 256	Sena 256	Couple 256
-3	0.75	0.75	0.75	0.75	0.75	0.75
-2	0.75	0.75	0.75	0.75	0.75	0.75
-1	1.00	1.00	1.00	1.00	1.00	1.00
0	1.00	1.00	1.00	1.00	1.00	1.00
1	1.25	1.25	1.25	1.25	1.25	1.25
2	1.25	1.25	1.25	1.50	1.25	1.50
3	1.50	1.50	1.50	1.50	1.50	1.50

This result is very useful for the design of the rate allocation for the conventional non-MMAD system. The optimal rate allocation non-MMAD system associated with the channel SNR can be calculated off-line only once. If we store the result into a mapping table, the system can use a table look-up to determine the rate.

5.5 Optimal Rate Allocation with MMAD

Here we continue to use the model simulation to study the optimal rate allocation for the system with MMAD. The simulation process is the same as the above, except it uses the MMAD PCC decoders instead of PCC conventional decoders. We obtain the optimal rate allocation of the MMAD system with the 512 by 512 Lena image using the curves shown in Figure 5.8.

Comparing the optimal rate allocation of the conventional non-MMAD system with that of the MMAD system, there are some similarities and some differences. One similarity is that the optimal rate R_s is increased as the channel SNR is increased. Similarly, the improper choice of the

optimal rate R_s can reduce the system RSNR by 4~10 dB; this value is less than that of the conventional non-MMAD system. It is also true for the MMAD system that at the optimal rate, the total distortion is mostly determined by the source distortion; the BER of PCC with MMAD is approximately zero.

However, an interesting result is found in that the optimal rate allocation for VQ using PCC with MMAD is not always the same as that for VQ using PCC only. For the same channel SNR, the increased channel error protection capability offered by including MMAD results in an optimal rate allocation which gives more rate to the source coding operation.

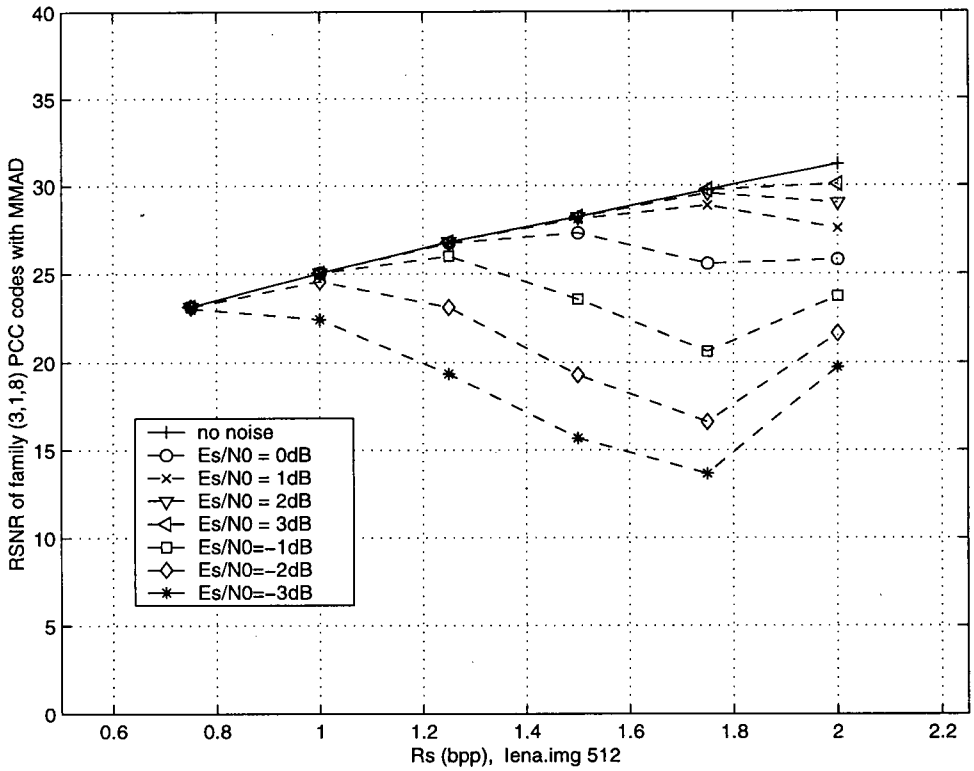


Figure 5.8: Optimal rate allocation for the system with MMAD

We continue to investigate the optimal rate allocation of the MMAD system for various

other images. For the same channel SNR, the optimal rate of the MMAD system is often higher than that of the non-MMAD system. Unlike the conventional system, the optimal rate allocation is not always the same for different images. See Table 5.3.

Table 5.3: Optimal R_s for MMAD systems for the various images

E_s/N_0 (dB)	Lena 512	Pepper 512	Couple 512	Lena 256	Sena 256	Couple 256
-3	0.75	1.00	0.75	0.75	1.00	0.75
-2	1.00	1.00	1.00	1.00	1.25	1.25
-1	1.25	1.50	1.00	1.25	1.75	1.25
0	1.50	1.50	1.25	1.75	1.75	1.75
1	1.75	1.75	1.25	1.75	1.75	1.75
2	1.75	1.75	1.75	2.00	2.00	2.00
3	2.00	2.00	1.75	2.00	2.00	2.00

To make this result clearer, we define ΔR_s as the difference of the optimal rate allocation between the two systems

$$\Delta R_s = R_s^M - R_s^N \tag{5.10}$$

where the R_s^M is the optimal rate allocation R_s for the system using MMAD, and R_s^N is the optimal rate allocation R_s for the non-MMAD system. From Table 5.4, we see that ΔR_s s are not always same for the different images, though ΔR_s s are almost always greater than zero.

Because the BER of the PCC using MMAD is a function of image statistics and the source VQ encoder and channel character, and that the optimal rate is often occurred at the BER equal to zero, the optimal rate allocation for the system with MMAD will depend on the image variance,

VQ encoder and the channel SNR. That is why ΔR_s s in Table 5.4 is not often the same for the different images at the same channel SNR.

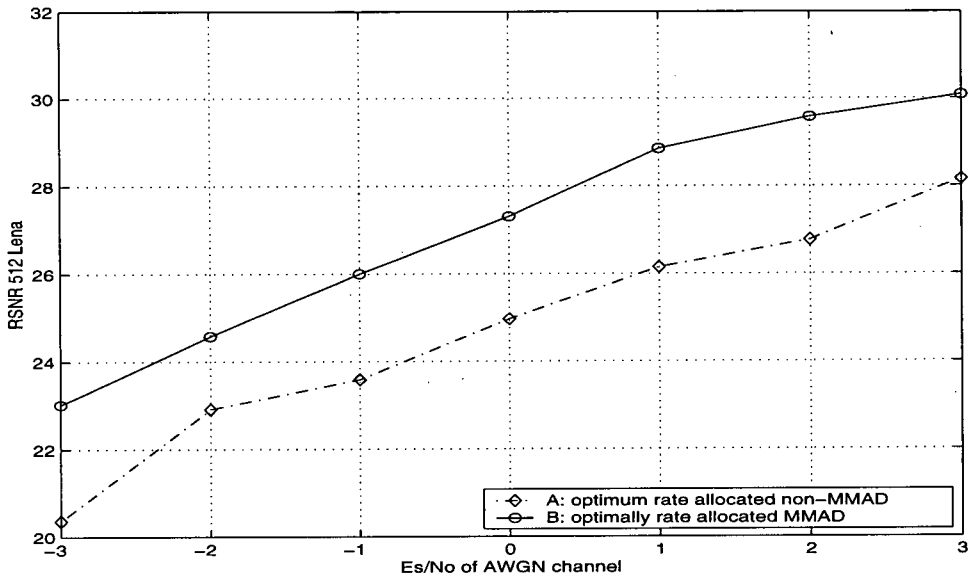
Table 5.4: ΔR_s between the two systems for the various images

E_s/N_0 (dB)	512 Lena	512 Pepper	512 Couple	256 Lena	256 Sena	256 Couple
-3	0.00	0.25	0.00	0.00	0.25	0.00
-2	0.25	0.25	0.25	0.25	0.50	0.50
-1	0.25	0.50	0.00	0.25	0.75	0.75
0	0.50	0.50	0.25	0.75	0.75	0.75
1	0.50	0.50	0.00	0.50	0.50	0.50
2	0.50	0.50	0.50	0.50	0.75	0.50
3	0.50	0.50	0.25	0.50	0.50	0.50

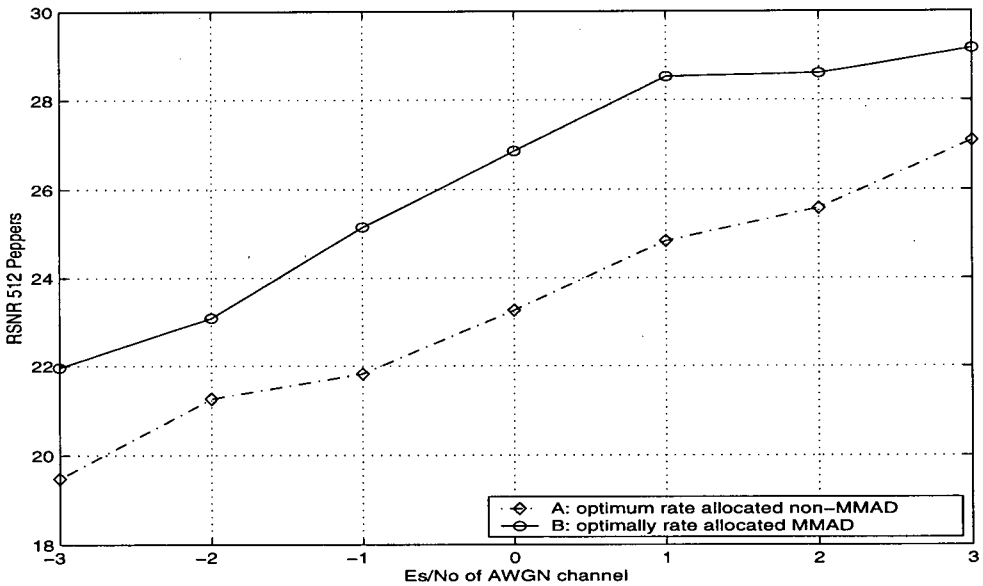
After determining the optimal rate allocation for the VQ with PCC transmission systems, here we propose a rate-allocated MMAD system B, which always uses the optimal rate allocation for the system applying MMAD. System B will be compared with the conventional rate-allocated non-MMAD system A, which always uses the optimal rate allocation for the system without MMAD.

We apply the two systems for the following images: 512 by 512 Lena, Peppers, Couple and 256 by 256 Sena. Figure 5.9. shows that when the source and the channel coding are optimally rate allocated, system B often gives a superior performance to the conventional rate allocated system A. The average improvement in RSNR of the rate allocated MMAD system is about 2.4 dB for the 512 Lena image, 2.85 dB for the 512 Peppers image, 4.5 dB for the 512 Couple image, and 4.1 dB for the 256 Sena image.

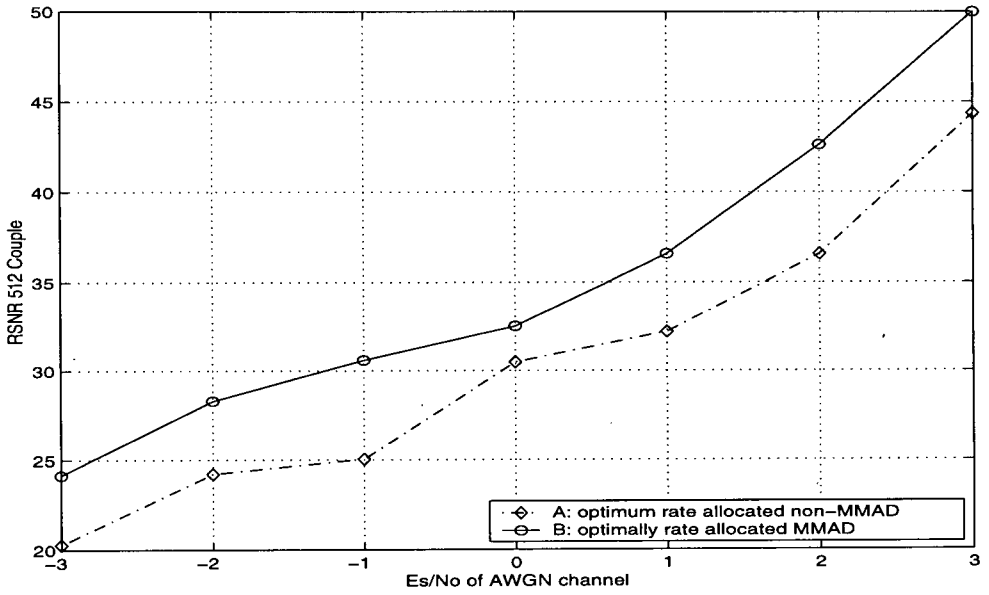
This significant improvement is because that for the same channel SNR, system B often selects the larger optimal rate R_s than that system A, which will make the source distortion smaller. Even when it chooses the same R_s, MMAD has stronger error protection ability than the conventional PCC decoding which will make the channel distortion smaller.



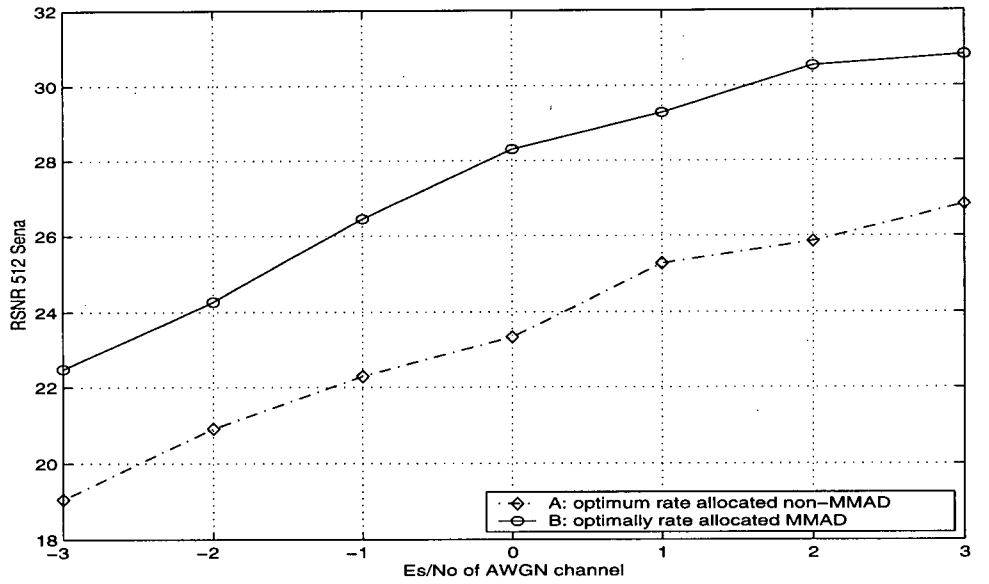
(a) 512 by 512 Lena



(b) 512 by 512 Peppers



(c) 512 by 512 couple



(d) 256 by 256 Sena

Figure 5.9: Optimally rate allocated systems with and without MMAD

The reconstructed images are shown in Figures 5.10 and Figure 5.11. They compare the performance of the two systems at the channel SNR 0dB for the 512 Lena image and that of the

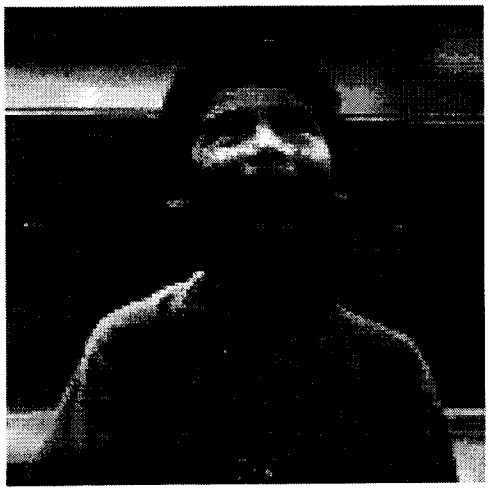
channel SNR -2dB for 256 for the Sena image. The original Lena and Sena images are shown in Chapter 2. The difference in the clarity of these images is much more evident on a high-resolution computer monitor than the shown low-resolution printer output.

For the Lena image when the channel SNR = 0dB, system A chooses the higher optimal rate R_s = 1.0bpp, and system B applies R_s = 1.50bpp. Their respective RSNRs are 24.96 dB, and 27.30 dB. The gain of the optimally rate allocated MMAD system is 2.46 dB. For Sena at channel SNR = -2dB, the gain of the optimally rate allocated MMAD system is 3.34 dB. These results show that the improvement of optimally rate allocated MMAD is very significant.

(a) System A, RSNR = 24.96 dB
R_s = 1.00bpp

(b) System B, RSNR = 27.30 dB
R_s = 1.50bpp

Figure 5.10: Performance of the two rate allocated systems at channel SNR = 0 dB



(a) System A, RSNR = 20.93 dB
R_s = 0.75bpp

(b) System B, RSNR = 24.27dB
R_s = 1.25bpp

Figure 5.11: Performance of the two rate allocated systems at channel SNR = -2dB

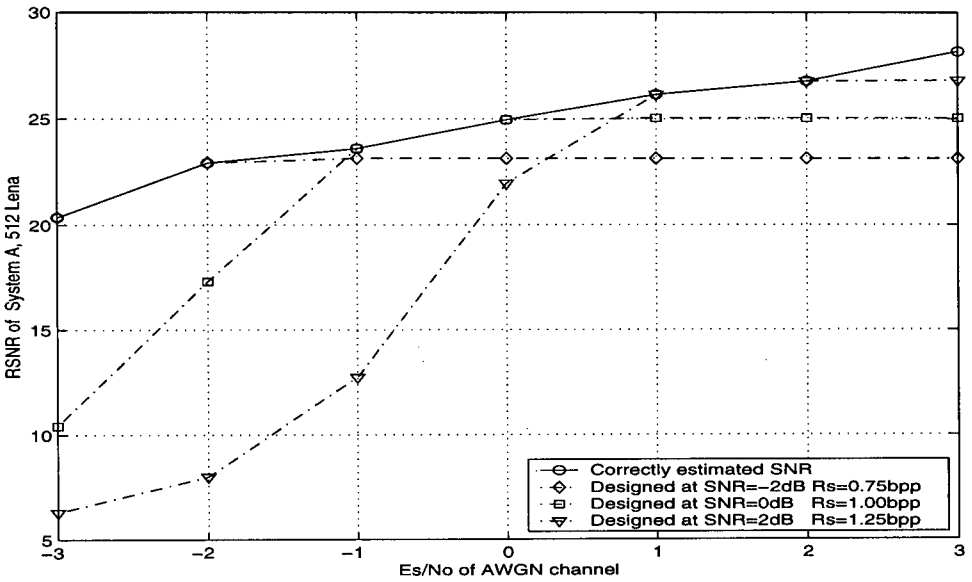
5.6 Sensitivity to Channel Mismatch

Because the optimal rate allocation depends on the channel SNR E_s/N_0, the performance of the rate allocated systems will suffer if the channel SNR is estimated in error. Here we will make a channel mismatch sensitivity analysis. Suppose we calculate the optimal allocation when the channel SNR is estimated as, for example, 2 dB, but when we send the data the channel SNR is actually 1.5 dB. How much performance will we lose?

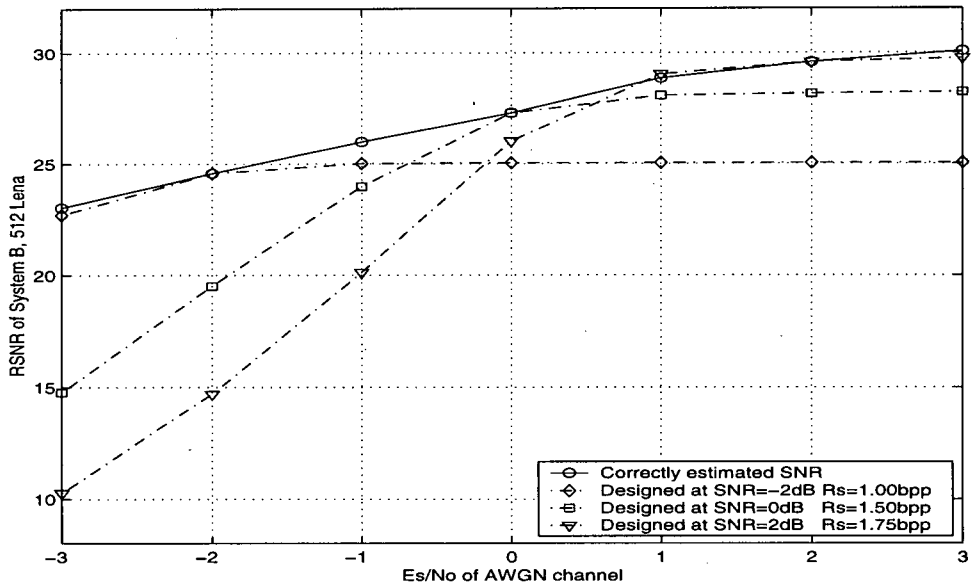
Figures 5.12 shows the effect of channel mismatch for the systems A and B. In each of these figures, we show the RSNR versus channel SNR curve for the system designed with perfect knowledge of the channel SNR, as well as three curves, each of which is obtained by using the allocation determined to be the best at a particular SNR, and using this allocation for transmission at any channel SNR. For example, the curve labeled "designed at SNR = 0 dB Rs=1.50 bpp"

shows the performance attained over the entire range of channel SNRs when we transmit using the allocation which is only optimal at the one channel SNR of 0 dB. The difference between this curve and the ideal design curve at the channel SNR 1 dB (-1 dB) gives the performance loss when the channel SNR has been under-estimated (over-estimated) by 1 dB. For system A a 1 dB mismatch can cause a 1.2 dB loss in RSNR; while for system B a 1 dB mismatch can cause a 2.0 dB loss in RSNR. However, system B still achieves the higher absolute RSNR value than system A in the same case of channel mismatch. The slightly greater noise sensitivity of system B is due to the fact that Markov-model aided decoding requires an estimate of the channel SNR, while non-Markov-model aided decoding does not.

The curves designed for a particular channel SNR fall off more rapidly from the ideal result at decreasing channel SNR, indicating that the loss is greater when the channel SNR is over estimated. Therefore, the main conclusion to draw from the noise sensitivity analysis curves is that for both systems, when the channel SNR is imperfectly known, it is better to design the source and channel code for a pessimistic estimate of the channel SNR.



(a) Non-MMAD rate allocated system A



(b) MMAD rate allocated system B

Figure 5.12: Effect of channel mismatch for the two rate allocated systems

Chapter 6 Conclusions and Future Work

6.1 Summary

In this thesis, we have investigated joint source-channel code design for Vector Quantized image transmission systems. We reviewed the LBG algorithm for VQ codebook design, and have considered two methods for assigning binary codewords (indices) to the resulting quantizer output points in the codebook. Of the two index assignment schemes we have found the natural binary code from the splitting algorithm to be the more noise robust, and have used it in the remainder of our study.

A noise robust source coder such as the one that we use has substantial residual redundancy in its output stream. In our approach, the channel decoder makes use of this residual redundancy by the technique known as Markov Model Aided Decoding. MMADs can be symbol-by-symbol decoders, or can be sequence decoders. As well, they can use a Markov Model of any finite order. In order to choose among the various MMAD options, we first considered MMAD without explicit channel coding. We found the O(1) sequence decoders to significantly outperform the O(1) symbol-by-symbol decoders. However, the O(2) symbol-by-symbol decoder with Soft Decision Feedback performs almost as well as the O(2) sequence decoder with Soft Decision Feedback. For the symbol-by-symbol decoders with SDF, increasing the order of the Markov model from one to two significantly improves the performance. For the sequence decoders, the performance gain attained by increasing the order of the Markov model from one to two is slight by RSNR measure. Because the O(1) sequence MMAD method is easily incorporated into the convolutional codes, in the ensuing investigation we considered only the O(1) sequence decoders.

We went on to consider MMAD with explicit channel coding, taking the channel codes to

be convolutional codes. To make optimal use of the source statistics in the decoding process the technique called trellis merging was found to be necessary. However, this technique is restricted to the cases in which the source codewords are shorter in length than the memory of the channel code. We proposed a concatenated system in which a pure (non-model aided) channel decoder based on the BCJR algorithm passed the a posteriori source bit probabilities to a cascaded MMAD which sequence decodes the noise corrupted source bit stream using the Markov model (i.e. the second stage consists of an MMAD without channel coding). Even though the passing of soft-information insures that there is no information loss incurred by breaking up the decoding into two stages, we find that the performance of the concatenated decoder is inferior to that of the integrated, merged trellis decoder. This is because in the concatenated system the reliability of the channel information is increased before the source information is applied to bias the decoding decision. In the integrated decoder, the source information biases a decision based on the raw channel data, and provides a more effective bias at this point in the decoding process.

We found very large gains – up to 9 dB in RSNR – by using MMAD. However, these large gains are attained in a region of low channel SNR where the channel code is no longer effective. To make a more fair evaluation of MMAD, we considered applying MMAD in a rate allocated system. Given a fixed information transmission rate (in pixels per second), and a fixed transmission channel bandwidth, the total distortion of the image transmission is significantly reduced by optimally allocating the overall coding rate between the source coding and the channel coding operations. Authors, including the present, have found that in an optimally rate allocated system the bit error probability is usually very low. Since the MMAD techniques give little improvement when the error rate is low, this would indicate that MMAD is not useful for rate-allocated systems. However, by comparing with the analogous the conventional rate allocated non-MMAD system,

we found the interesting result that indeed MMAD can improve a conventional rate-allocated system; and in fact the coding gain is typically around 2 dB in channel SNR. This is because MMAD does increase the SNR range over which the channel code provides a low probability of bit error. This increased strength in the channel code results in a rate allocation that gives more of the overall fixed rate to the source coder, thus resulting in a higher quality image even when the bit error rate is low.

Because the optimal rate allocation is always at the point where the channel code is just strong enough to correct almost all the errors, for non-MMAD the optimal rate allocation is almost independent of the image. This means that a system can be simply implemented by employing a table look-up for the rate allocation to use at a particular channel SNR. However, for MMAD the BER depends on the image, and therefore the rate allocation must be determined for each image by an off-line calculation before the image can be formatted for transmission. We did the calculation by employing a discreet event simulation. This calculation could be done more efficiently if one could find an analytic expression for the BER of a Markov-model aided channel code. Our results give strong motivation to solve this open problem.

Because the point of optimal rate allocation depends on the channel SNR, inaccurate channel SNR estimation generally decreases the performance of the rate allocated systems whether or not they employ MMAD. Our experiments indicate that MMAD and non-MMAD rate allocated systems have similar sensitivity to channel SNR mismatch, and both are fairly robust. Based on the performance results, one is better off selecting the optimal rate for a pessimistic channel SNR if the value cannot be known precisely.

The MMAD applied to the rate allocated systems is the O(1) sequence MMAD. Because

the O(2) sequence MMAD with SDF gives better performance than the O(1) sequence MMAD, we could also improve the performance of the rate-allocated systems by using the O(2) sequence MMAD with SDF. However, this would increase the computational complexity significantly because the channel decoder would have to be changed from Viterbi-based to BCJR based.

6.2 Proposed Future Work

Suggestions for future study of MMAD for VQ image transmission are as follows:

- Find an analytic expression for the BER of MMAD, so the rate allocation can be calculated more efficiently.
- Evaluate the MMAD rate-allocated system performance over fading channels.
- Consider other VQ techniques, channel codes, and modulation methods.
- Study ways to obtain an accurate channel SNR estimate.

Glossary

app	a posteriori probability
AWGN	Additive White Gaussian Noise
BER	Bit Error Rate
BSC	Binary Symmetric Channel
CC	Convolutional Code
bpp	bits per pixel
GLA	Generalized Lloyd Algorithm
HDF	Hard Dicision Feedback
LBG	Linde-Buzo-Gray
JSCC	Joint Souce/Channel Coding
MMAD	Markov Model Aided Decoding or Markov Model Aided Decoder
MMACD	Markov Model Aided Convolutional Decoding or Markov Model Aided Convolutional Decoder
NBC	Natural Binary Code
O(1)	first order
O(2)	second order
PCC	Punctured Convolutional Code
RSNR	Reconstruction Signal-to-Noise Ratio
SNR	Signal-to-Noise Ratio
VQ	Vector Quantization

Bibliography

[1] C.E. Shannon. "Coding Theorems for a Discrete Source with a Fidelity Criterion.", *IRE National Convention Record*, Part 4, pp. 142-163, 1959.

[2] C.E.Shannon, Collected papers, N. J. A. Sloane and A. Wyner, Eds. New York, *IEEE Press*, pp. 40, 1993.

[3] Andrea J. Goldsmith, Michelle Effros, "Joint Design of Fixed-Rate Source Codes and Multiresolution Channel Codes", *IEEE Transaction on Communication*, vol. 46, pp. 1301-1312, Oct. 1998.

[4] S. Emami and S. Miller, "DPCM Picture Transmission over Noisy Channels with the Aid of a Markov Models,", *IEEE Transaction on Image Processing*, vol. 4, no. 11, pp. 1473-1481, Nov. 1995.

[5] Robert Link and Samir Kallel, "Optimal Use of Markov Models for DPCM Picture Transmission over Noisy Channels", submitted to *IEEE Transaction on Communication,* Feb. 1999.

[6] K. Sayhood, F. Liu and J.D. Gibson, "A Constrained Joint Source/Channel Coder Design", *IEEE Journal Selected Areas Communication*, vol.12, no. 9, pp. 1584-1592, Dec. 1994.

[7] W. Xu, J. Hagenauer, and J. Hollman, "Joint Source-channel Decoding Using the Residual Redundancy in Compressed Image", *ICC/SUPERCOMM'96*, pp.142-148, June 1996.

[8] Rober. Link, Norman C. Lo and Samir Kallel, "Source-Aided Turbo Decoding of Serially Concatenated Codes", submitted to *IEEE Transaction on Communication*, May 1999.

[9] Retrand Hochwald, "Trade-off Between Source and Channel Coding on a Gaussian Channel", *IEEE Transaction on Information. Theory*, vol. 44, pp. 3044-3055, Nov. 1998.

[10] Khalid Sayhood, "Introduction to Data Compression", Morgan Kaufman Publishers, Inc., 1992.

[11] T.M. Cover and J.A. Thomas, "Elements of Information Theory", *IEEE Transaction on Information Theory*, vol. 29, pp. 820-824, Nov. 1983.

[12] T. Berger, "Rate Distortion Theory: A Mathematical Basis for Data Compression", Englewood Cliffs, NJ: Prentice Hall, 1971.

[13] R.M. Gray, "Entropy and Information Theory", *New York: Springer-Verlag,* 1990.

[14] LINDE Y., BUZO A., and GRAY R.M., "An Algorithm for Vector Quantizer Design", *IEEE Transaction on Communication*, vol. 28, pp.84-95, Jan. 1980.

[15] Robert M. Gray, David L. Neuhoff, "Quantization", *IEEE Transaction on Information Theory*, vol. 44, no. 6, pp. 2325-2383, Oct. 1998.

[16] "Global Convergence and Empirical Consistency of the generalized Lloyd algorithm", *IEEE Transaction on Information Theory*, vol. 32, pp. 148-155, Mar. 1986.

[17] K. Zeger and A. Gersho, "Pseudo-Gray Coding.", *IEEE Transaction on Communication*, vol. 38, no. 12, pp. 2147-2158, Dec. 1990.

[18] Farvardin, N., "A Study of Vector Quantization for Noisy Channels", *IEEE Transaction on Information Theory*, vol. 36, no. 4, pp. 799-809, 1990.

[19] H-S. Wu and J. Barba, "Index Allocation in Vector Quantization for Noisy Channels", *Electron Letter*, vol. 29, no. 15, pp. 1317-1319, July 1993.

[20] J. R. B. De Marca and N. S. Jayant, "An Algorithm for Assigning Binary Indices to The Codevectors of a Multidimensional Quantizer", *IEEE International. Communication Conf*erence, Seattle, WA, pp.1128-1132, June 1987.

[21] J. Hagenauer, "Source-Controlled Channel Decoding", *IEEE Transaction on Communication,* vol. 43, no. 9, pp. 2449-2457, Sept. 1995.

[22] Stephen Bryant Wicker, "Convolutional Codes", chapter in the book of Error Control Coding, pp. 268-271.

[23] Robert Link and Samir Kallel, "Markov Model Aided Decoding for Image Transmission using Soft-Decision-Feedback", submitted to *IEEE Transaction on Communication*, Mar. 1999.

[24] W.E. Ryan. "A Turbo Code Tutorial", http://telsat.nmsu.edu/~wryan/turbo2c.ps.

[25] L.R.Bahl, J.Cocke, F.Jelinek, and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate", *IEEE Transaction on Information Theory*, vol. 20, no. 2, pp. 284-287, Mar. 1974.

[26] Samir Kallel, Norman C. Lo, and Robert Link, "Markov Model Aided Channel Decoding", report, Nov. 1998.

[27] P.Elia, "Coding for Noisy Channels", *IRE Convention Record*, Part 4, pp. 37-47, 1995.

[28] A.J. Viterbi, "Error Bounds for Convolutional Codes and an Asymptotically Optimum Decoding Algorithm", *IEEE Transaction on Information Theory*, vol. 13, pp. 260-269, Apr. 1967.

[29] G. D. Forney, Jr. "Convolutional Codes: Algebraic structure", *IEEE Transaction on Information Theory*, vol. 16, no.6, pp. 268-278, Nov. 1970.

[30] "The Algebraic Theory of Convolutional Codes", Handbook of Coding Theory, in preparation.

[31] Norman C. Lo, "The Application of Source-aided Turbo Decoding to IS-95 CDMA Systems", master thesis, University of British Columbia, Dept. E.E., Apr. 1999.

[32] Amin Alavi, "An Adaptive Subband-Based Joint Source-Channel Coder for Image Transmission", master thesis, University of British Columbia, Dept. E.E., June 1999.

[33] David Haccoun, Guy Begin, "High-Rate Punctured Convolutional Codes for Viterbi and Sequential Decoding", *IEEE Transaction on Communication*, vol. 37, no. 11, pp. 1113-1125,. Nov. 1989.

[34] J. Bibb Cain and George C. Clark, JR., "Punctured Convolutional Codes of Rate (n-1)/n and Simplified Maximum Likelihood Decoding", *IEEE, Transaction on Information Theory*, vol. 25, no. 1, pp. 97-100, Jan. 1979.

[35] Joachim Hagenauer, "Rate-Compatible Punctured Convolutional Codes and Their Applications", *IEEE Transaction on Communication*, vol. 36, no. 4, pp. 389-400, April 1988.

[36] G. Begin and D. Haccoun, "High Rate Punctured Convolutional Codes: Structure Properties and Construction Techniques", *IEEE Transaction on Communication*, vol. 37, no. 12, pp. 1381-1200, Dec. 1989.

[37] Yutaka Yasuda, Kanshiro Kashiki and Yasuo Hirata, "High Rate Punctured Convolutional Codes for Soft Decision Viterbi Decoding", *IEEE Transaction on Communication*, vol. 32, no. 3, pp. 315-319, March 1984.

[38] Hamid Jafarkhami, Paschalis Ligddas, and Nariman Farvardin, "Adaptive Rate Allocation in a Joint Source/Channel Coding Framework for Wireless Channels", *IEEE Transaction on Communication*, pp 492-495, 1996.

Appendix

A.1. BCJR Based Trellis Merging MMACD

As in the Viterbi based MMACD, trellis merging techniques can be used in the BCJR algorithm. Let L be the number of VQ source symbols in a block, an input sequence **u** composed of Lq-bit symbols and a few q-bit zeros $\boldsymbol{u} = (u_1, \ldots, u_L, \ldots, u_\tau)$, $\tau = L + \left\lceil \frac{m}{q} \right\rceil$, where $u_i = (u_i^0, u_i^1, \ldots, u_0^{(q-1)})$, composed of q bits represents one symbol instead of one bit, where u_i^j represents j^{th} bit in the i^{th} VQ source symbol. If we use a qn-bit channel codeword instead of a n-bit channel codeword, then the encoded channel codeword sequence will be $\boldsymbol{x} = (x_1, \ldots, x_L, \ldots, x_\tau)$, where $x_i = (x_i^0, x_i^1 \ldots, x_i^{q-1})$ corresponding to the source symbol u_i, $x_i^j = (x_i^{j(0)}, \ldots, x_i^{j(n-1)})$ refers to the n-bit channel codeword corresponding to the source bit u_i^j and $x_i^{j(k)} \in \pm 1$. From the trellis diagram, the output of a branch will be qn bits codeword. Similarly, the received qn-bit codeword sequence $\boldsymbol{y} = (y_1, \ldots, y_L, \ldots, y_\tau)$, $y_i = (y_i^0, y_i^1, \ldots, y_i^{q-1})$, $y_i^j = (y_i^{j(0)}, y_i^{j(1)}, \ldots, y_i^{j(n-1)})$. The equations in Section 4.21 needn't change except that the value of u_i is from 0 to 2^q instead of 0,1. The transition probability (4.16) becomes:

$$\gamma_i(s', s) = P(u_i | u_{i-1}) P(y_i | x_i) \tag{A.1}$$

where $P(u_i|u_{i-1})$ is the *a priori* probability of the source symbol u_i which cause the transition $(s_{i-1} = s') \rightarrow (s_i = s)$ which results in the encoded qn-bit codeword x_i; and $P(y_i|x_i)$ is the channel transition probability of the qn-bit channel codeword:

$$P(y_i|x_i) = \prod_{j=0}^{q-1} \prod_{k=0}^{n-1} P(y_i^{j(k)}|x_i^{j(k)}) \tag{A.2}$$

$$\gamma_i(s', s) = \begin{cases} P(u_i|u_{i-1}) \cdot P(y_i|x_i) & s' \to s \\ 0 & else \end{cases} \tag{A.3}$$

$s' \to s$ means that the transition $(s_{i-1} = s') \to (s_i = s)$ exists because both encoder inputs u_i and u_{i-1} have to be determined by the states s_{i-1}, s_i at times i-1 and i. This algorithm requires that q should be less or equal to the memory m of the code, too, because of using the merged trellis.

For decoding, define A_i^j as the set of state S_i corresponding to the current input $u_i = j$, j= 0. 1, ..., 2^q-1, which means that from $S_i = (s_i^0, s_i^1, ..., s_i^{(m-1)})$, we can see that $u_i^{q-1}=s_i^0$, ..., $u_i^0=s_i^{q-1}$. The decoder to choose u_i=j which maximizes:

$$P(u_i = j|y) = \frac{1}{\lambda_\tau(0)} \sum_{S_i \in A_i^j} \lambda_i(s) \tag{A.4}$$

where j = 0, 1, ..., 2^q-1. In practice, we use (4.19) and (4.20) to calculate α and β except that b represents a symbol input, b = 0, 1, ..., 2^q-1 instead of a bit input 0 or 1. If q< m, we can see that (4.18) (4.18) still can reduce the complexity of the computation.

Our simulation result shows that the BCJR based trellis merging MMACD gives essentially the same BER performance as the Viterbi algorithm with the Viterbi based trellis merging MMACD. On the whole, the trellis merging MMACD techniques can give better error protection when the channel SNR is low. Trellis merging reduces the overall length of the trellis by a factor of q, but increases the number of branches by a factor of 2^{q-1}. When q is increased

from 1 to 2, the computation complexity can be ignored.

The BCJR based trellis merging algorithm is more complicated than the Viterbi based trellis merging algorithm. It suggests that one had better use the Viterbi trellis merging algorithm if $q \leq m$. However the BCJR based trellis merging is useful for constructing source-aided turbo decoding systems[8] [31] or will be useful to other concatenated source-aided systems in the future. In this paper, we used the BCJR trellis merging without Markov model to generate the *a posteriori* symbol probabilities in order to compare with the concatenated decoding using the *a posteriori* bit probabilities.