UNIVERSITY OF CALIFORNIA

Los Angeles

Channel Coding for Video Transmission over Unknown Channels

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

by

Jaehyeong Kim

1996

The dissertation of Jaehyeong Kim is approved.

Kung Yao

Abeer A.H. Alwan

Kirby A. Baker

Gregory J. Pottie, Committee Chair

University of California, Los Angeles

1996

DEDICATION

This dissertation is dedicated to my parents.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

First of all, I would like to express my deepest gratitude to my advisor, Professor Gregory Pottie, for his full support, encouragement and guidance. His patience and understanding relieved me from the rigors of my research and made my school years a lot easier. I would also like to give thanks to Professor Kung Yao, Professor Abeer Alwan and Professor Kirby Baker for investing their time to be on my dissertation committee and for their helpful input.

Fortunately, I had chances to meet several good teachers in my life. Among them are my junior-high school teacher Jong Yoon Lee and Professor Byeong Gi Lee. Their guidance at my earlier stage is essential for my academic achievement. I am also indebted to Dr. Nambi Seshadri for his support and guidance when I was a summer intern at AT&T Bell Laboratories.

I am thankful to my colleagues for many helpful technical discussions and friendship. Among them are Victor Lin, Charles Wang, Chris Hansen, Ben Tang, Eldad Perahia, Heung-No Lee, Kathy Sorabi, Dennis Connors and George Kondylis. I would also like to show my appreciation to Cheon Won Choi and other Korean students for their guidance in general and friendship. My friend Soung Soo Yi and other people whom I met in Hershey Hall where I have stayed for five years, I want to thank for their friendship. They were always with me when I need people to exchange minds with.

I am deeply grateful to the people at Jesus Christ Korean Church and especially to pastor Chang Hwan Park for their spiritual guidance and prayers.

My appreciation also goes to David Sarnoff Research Center and the state of California Micro Program who provided financial support.

Most importantly, I would like to thank my parents from the bottom of my heart. Without their immeasurable love and support, I would not have been what I am. Thanks

god for bringing so many good people into my life and for giving me the chance and ability to accomplish this goal.

VITA

Jaehyeong Kim

March 22, 1965	Born, Pusan, Korea
1988	B.S., Electronics Engineering, Seoul National University, Korea
1990	M.S., Electronics Engineering, Seoul National University, Korea
1988–1990	Research Assistant, Seoul National University, Korea
1993–1996	Research Assistant, University of California at Los Angeles
1993	Teaching Assistant, University of California at Los Angeles

PUBLICATIONS AND PRESENTATIONS

B.G. Lee, M.G. Kang and J. Kim
Statistical Processing over Acoustic Signals
Seoul National University,Korea, Jan., 1989.

J. Kim and and B.G. Lee
System Recognition Using Cepstrum Coefficients
The second joint conference on Signal Processing,Seoul, Korea,Vol.2, No.1, pp 150-153, 1989.

J. Kim
Application of Cepstrum Techniques for Acoustic Signal Source Recognition
M.S. thesis, Seoul National University, Seoul, Korea, Feb., 1990.

J. Kim and G. J. Pottie
On Punctured Trellis Coded Modulation
Proc. of IEEE International Conference on Communications 1995.

J. Kim and G. J. Pottie
On Punctured Trellis Coded Modulation
IEEE Trans. on Information Theory, VOL. 42, NO. 2, pp. 627-636, March 1996.

N. Seshadri and J. Kim
Coding and Modulation for Simultaneous Voice and Data Transmission
IEEE Communication Theory Mini-Conference 1995.

ABSTRACT OF THE DISSERTATION

Channel Coding for Video Transmission over Unknown Channels

by

Jaehyeong Kim

Doctor of Philosophy in Electrical Engineering

University of California, Los Angeles, 1996

Professor Gregory J. Pottie, Chair

Punctured convolutional codes result in some savings in the complexity of Viterbi decoders, compared to other codes of the same rate. However, in general use of the punctured structure in the decoder results in a performance loss for trellis codes, due to difficulties in assigning metrics. We provide constructions for punctured rate-2/3 codes based on decomposition of the metric into orthogonal components. These show no loss in performance for trellis coded QAM and PSK. We also provide 180^o and 90^o rotationally invariant punctured TCM for QAM signalling.

We have also considered a family of unequal error protection (UEP) codes which use a four way partitioning in a one dimensional lattice with multilevel codes. These non-regular set partitionings combined with non-uniform signal constellations provide large minimum distance and small path multiplicity for the important data. However, in this case, standard code search techniques do not give us reliable information for estimation of coding gain. A new code search method is introduced for better estimation of actual coding gain. We show how to make 90^o rotationally invariant codes by using 180^o

rotationally invariant rate-1/2 convolutional code and resolving in-phase and quadrature-phase power.

The original motivation for this work was consideration of possible alternative coding methods for HDTV systems. While the coder for HDTV has subsequently been standardized, we have developed a means for improving the decoding reliability beyond what is anticipated in the standard.

Chapter 1

Introduction

In video transmission one is often confronted with the problem of a channel which is unable to transmit enough data to reproduce images of desired quality. Channel coding provides two possible solutions. It can increase the range for high quality transmission, or may be used to ensure reliable reception of lower quality images beyond the range of reliable high-quality transmission. A set of trade-offs in the relative ranges of the two levels of quality is available. In most modern transmission systems, some form of channel coding is included to increase the reliability and/or range of the service. The resulting hardware must be cost-effective for broad customer acceptance. There will be a wide variety of multimedia networks, potentially each with their own channel coding methods. The cost and development time for these systems can be reduced if critical components of the decoder design may be re-used in a variety of applications.

We have investigated new channel coding schemes which make convenient use of previous decoder designs as well as methods for variable error protection transmission that extensively re-use a common decoder engine. Techniques based on *punctured convolutional code* (PCC) appear to be very attractive. Punctured convolutional codes need

fewer arithmetic operations than ordinary convolutional codes. In addition to the complexity advantage, in a situation where not all bits require equal error protection a family of punctured codes of variable rate may be used. This allows the use of one basic decoder, reducing the area devoted to the decoder in ASIC implementations. Examples of such re-use include "pragmatic" trellis coded modulation [1] [32]. *Punctured trellis coded modulation* (PTCM) uses PCC as its component and has the advantages of PCC. However, in general use of the punctured structure in the decoder results in a performance loss for TCM, due to difficulties in assigning metrics for the decoder. We provide constructions for PTCM based on decomposition of the metric into orthogonal components. These show no loss in performance for trellis coded QAM and PSK.

One practical problem with PTCM is the difficulty of providing rotational invariance or resolving phase ambiguity. To compensate for a phase ambiguity in the receiver, there are essentially two approaches. We can estimate the phase ambiguity by sending a fixed sequence of modulation phases to initialize data communication. On the other hand, we can design trellis codes that are transparent to phase offsets at multiples of the smallest difference between two modulation angles in the signal constellation. Our concern is the latter approach and we provide 180^o and 90^o rotationally invariant PTCM for QAM signalling.

In many speech and image coding schemes, some of the coded bits are very important while some others are less important from the point of view of the perceptual quality of the reconstructed signal. In such applications, use of *unequal error protection* (UEP) which provides different error protection for different classes of information, may provide benefits. For example, HDTV (High Definition Television) broadcast allows the possibility of offering several grades of service. Customers close to the transmitter could

receive the full resolution promised by HDTV while those at a larger range would receive NTSC quality (normal TV quality) images, which can be accomplished by using multi-resolution codes. Multi-resolution codes could be a subset of UEP in the sense that NTSC quality information is important and highly protected and the additional information is not very important but can up-grade the quality of the image if correctly recovered. In UEP, important information is transmitted at a low rate and the rest at a higher rate. There are two ways of achieving UEP in a broadcast channel.

The first approach is *time sharing* or time-multiplexing method, where different rate signals are in different time slots. This scheme recovers only the signals at the time slot for low rate data if the channel capacity is low. The generalized time sharing scheme in which a code of non-zero rate specifies the multiplexing rule rather than using a fixed multiplexing rule (see Calderbank/Seshadri [3]) is more clever, but is still not the best we can do if TCM may be employed.

The second approach is *superimposing* higher rate information and lower rate information in one signal, where we recover from the signals in all time slots the low rate information if the channel capacity is low. Cover [50] showed that superimposing codes may be preferable to time sharing in that for a small reduction in capacity for the high rate, the low rate information may be better protected. The use of multilevel codes and multi-stage decoding (see Calderbank [33] and Pottie/Taylor [34]) provides a rather flexible way of allocating different levels of error protection to various classes of data.

We have designed a family of unequal error protection (UEP) codes based on superimposition, which provides good coding gains for both data classes (important data and less important data) with reasonable complexity. Furthermore, we can easily make 90^o rotationally invariant codes by using 180^o rotationally invariant rate-1/2 convolutional codes and resolving in-phase and quadrature-phase power.

This dissertation is organized as follows. A brief tutorial description of channel coding including convolutional codes and TCM is presented in chapter 2. In chapter 3, we discuss PTCM for QAM and PSK. We describe how to construct the punctured trellis and signal constellation, and describe the decoding methods. 180^o and 90^o rotational invariance of PTCM is discussed in chapter 4. In chapter 5, we propose a family of unequal error protection code having a rotationally invariant structure. The original motivation for this work was consideration of possible alternative coding methods for HDTV systems. While the coder for HDTV has subsequently been standardized, we have developed a means for improving the decoding reliability beyond what is anticipated in the standard. A description and the results of the HDTV systems development project is given in chapter 6. Concluding remarks are given in chapter 7.

Chapter 2

Channel Coding

A communication system connects an information source to a user through a channel as shown in Figure 2.1. The information sequence is first processed by a source encoder designed to represent the information in more compact symbols called *source codewords*. The source codewords might be represented by a group of bits. The *channel encoder* transforms a binary information sequence into another sequence called the *channel codeword*. The channel codeword is a new, longer sequence that has more redundancy than the binary source codeword. This enables correction of errors introduced by a noisy channel. The *modulator* converts the channel codeword into a corresponding analog symbol from a finite set of possible analog symbols. The sequence of analog symbol is transmitted through the channel. Because the channel is subject to various types of noise, distortion and interference, the channel output differs from the channel input. The *demodulator* converts each received channel output signal sequence into one of the channel codeword symbols. Each demodulated symbol is the best estimate of the transmitted symbol, but the demodulator makes some errors because of the channel noise. The channel decoder uses the redundancy in a channel codeword to correct

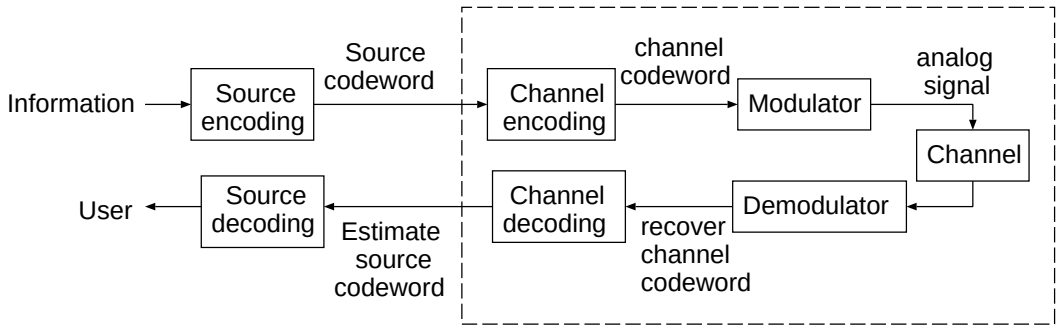


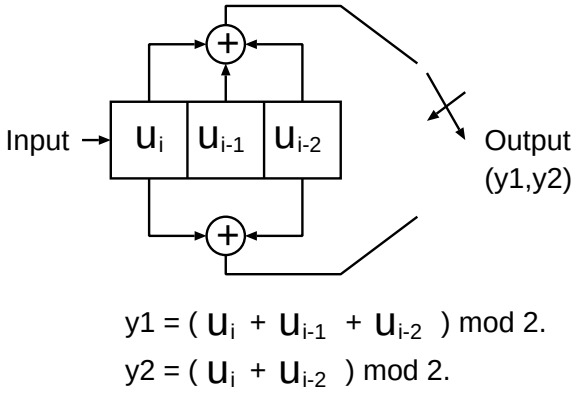
Figure 2.1: Block diagram of a digital communication system.

the errors in the received words and then produces an estimate of the source codeword. The source decoder performs the inverse operation of the source encoder and delivers its output to the user. Since our main concern is channel coding, we will consider the components in the dashed block in Figure 2.1.

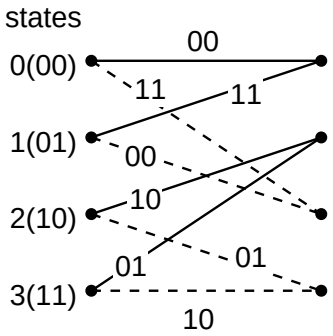
The code rate R_c is defined as $\frac{k}{n}$ when the number of encoder output bits per each k bit input sequence is n. There are two categories of channel coding. *Block codes* have a strong algebraic structure, where a finite length information sequence is encoded into a finite length encoded sequence. *Trellis codes* of infinite length can be represented by a tree and can be decoded by tree searching algorithms. One of the most useful classes of trellis codes are *convolutional codes* [49], which can be generated by a linear shift-register circuit that performs a convolution operation on the information sequence. The Viterbi algorithm(VA) [18] [47] has gained widespread popularity for decoding convolutional codes.

2.1 Convolutional codes

A four state rate-$\frac{1}{2}$ convolutional encoder and the corresponding code *trellis* is shown in Figure 2.2. As shown in Figure 2.2 (a), the binary input data to the encoder is



(a) Convolutional encoder.



(b) Four state code trellis.

Figure 2.2: Four state rate-1/2 convolutional encoder and corresponding code trellis.

shifted into the shift register which has three stages (one has the present input u_i and the other two have the past inputs (u_{i-1}, u_{i-2}) representing the state). If the current state is (u_{i-1}, u_{i-2}), the next state is (u_i, u_{i-1}). There are four possible states (0,0), (0,1), (1,0) and (1,1) or equivalently, 0, 1, 2 and 3 in decimal notation. The output of the encoder is determined by the input and the state of the encoder. The code trellis in Figure 2.2 (b) shows their relation. Each state in the trellis has two branches leaving and entering it. A solid line denotes the output generated by the input bit 0 and a dotted line denotes the output generated by the input bit 1. All the possible paths in the trellis could be codewords and the minimum distance between them increases as the number of states increases for well-constructed codes.

The Viterbi algorithm considers the code to be represented by a trellis, which is a periodically repeating structure with nodes or states connected by edges or branches which are labeled by the encoder outputs corresponding to the state transitions. The VA finds the connected path through the trellis that is closest to the received sequence of bits or symbols according to the metric or distance measure. For additive Gaussian noise

as the only channel impairment, the metric is the squared Euclidean distance between the symbol corresponding to the branch and the received symbol for that time interval. The VA accumulates the metric along each path, then selects only the branch entering a state with the lowest accumulated metric, killing the others. These decisions are stored in memory. After the decoder has proceeded a depth of L branches, it initiates traceback, searching back along one surviving path to determine the branch decided upon at the beginning. The main computational blocks of a Viterbi decoder are metric computation, the add/compare/selection(ACS) process and path traceback. Among them, in general the ACS process is more computationally demanding since we require one such operation for each state.

One simple example of the operation of the VA is illustrated in figure 2.3. We assume we BPSK (binary phase shift keying) signalling, i.e. we transmit +1.0 when the encoded output is 1 and we transmit -1.0 when the encoded output is 0. The received signal is noise corrupted. At time 0, the sequence (0.9,1.2) is received. The decoder computes the squared Euclidean distance (squared Euclidean distance is an optimal distance measure under Gaussian noise environment) of the received sequence to the modulated code words assigned to the two branches out of state 0. This step is shown in Figure 2.3 (b). The numbers in the branches denote the squared Euclidean distance between the received signal and modulated signal of the corresponding branches, and are called the *branch metric*. For example, From Figure 2.2 (b), the codeword of the branch connecting state 0 to state 0 is 00 and the modulated signal is (-1,-1). The squared Euclidean metric is $(0.9-(-1))^2+(1.2-(-1))^2=8.45$. In the same way the metric of the branch connecting state 0 to state 2 is $(0.9-1)^2+(1.2-1)^2=0.05$. The numbers in the parentheses denote the *accumulated metric* of the two paths. The accumulated metric

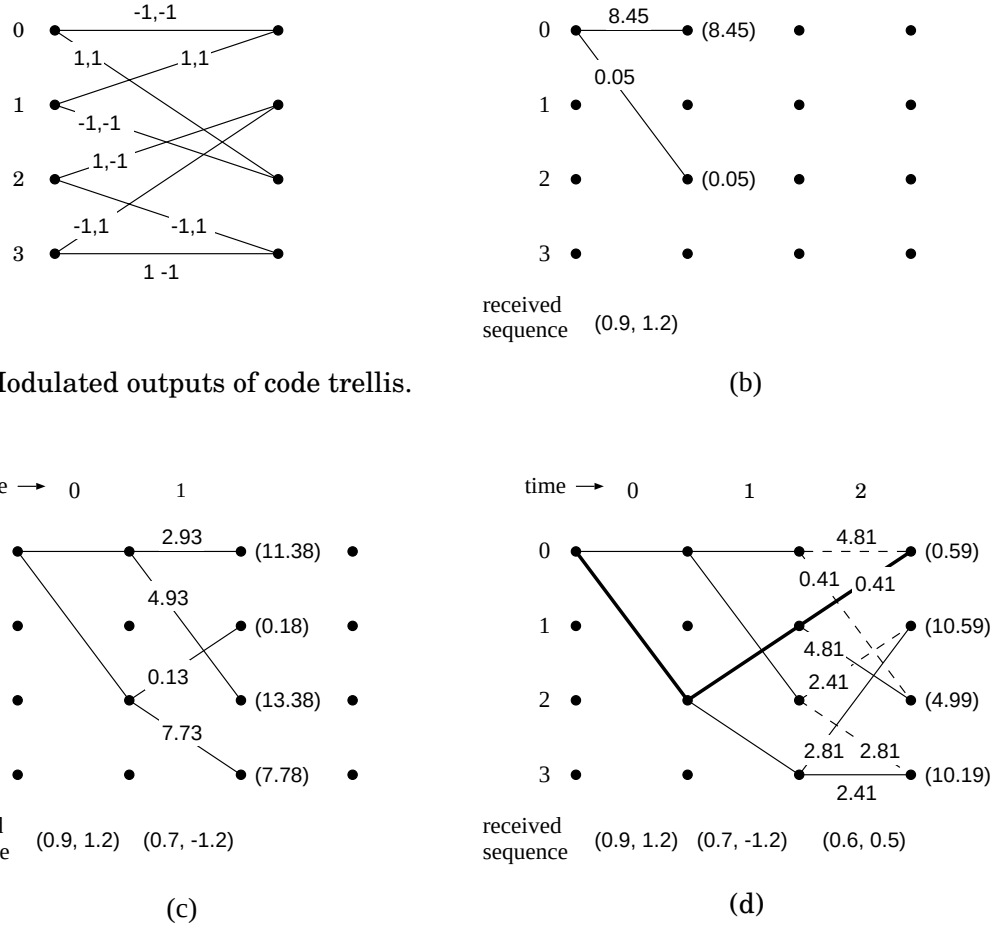
is defined as the sum of metrics of all branches on that path. Next, the sequence (0.7,-1.2) arrives. The decoder now computes four branch metrics as shown in Figure 2.3 (c). The accumulated metrics of the four paths are again shown in parentheses. Next the decoder computes eight branch metrics corresponding to the received sequence (0.6,0.5) at time 2 (see Figure 2.3 (d)). Two paths enter each state in this figure, of which the Viterbi algorithm retains the one with the smaller accumulated metric. The other path is discarded from further consideration. The discarded paths are noted as a dotted line.

At the end of the time 2, state 0 has the smallest accumulated metric. The history of the state 0 will be chosen as a correct path which is noted as a thick line. As a result, the correct encoded sequence at time 2 is (1,0,0).

Decoding a rate-$\frac{m}{n}$ convolutional code is similar to decoding a rate-$\frac{1}{n}$ code, with the difference that there are 2^m branches entering each trellis state. The Viterbi algorithm must therefore select one of these 2^m branches. The number of comparisons can be reduced with the use of *punctured* convolutional codes (PCC) [13]-[15]. Decoding a rate-$\frac{m}{m+1}$ convolutional code requires $2^m - 1$ comparisons per state, while a PCC with the same rate needs only m comparisons in each state. This difference becomes large as the code rate increases.

The principle behind punctured codes is easily explained using a four state rate-1/2 code and its trellis [19]. If we delete, or *puncture*, every fourth bit provided by the encoder, the resulting code produces three output bits for every two input bits and hence has rate-2/3. The trellis for this code is shown in Figure 2.4(a), where an **x** indicates a punctured output bit. The trellis for the rate-2/3 code shown in Figure 2.4(b) is equivalent to the trellis in Figure 2.4(a), although one stage of the former corresponds to two stages of the latter.

In addition to the complexity advantage, in a situation where not all bits require



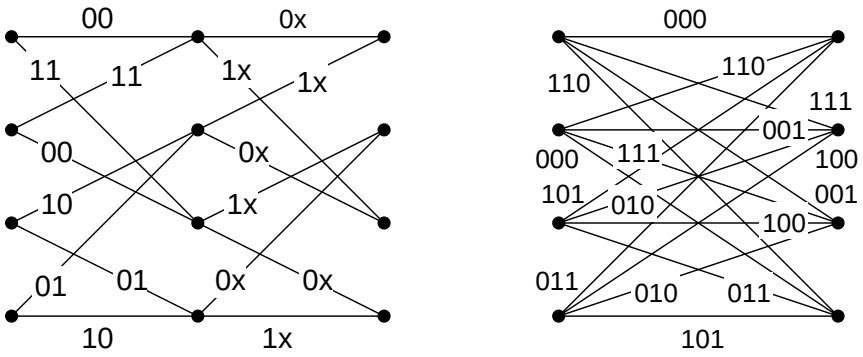
(a) Modulated outputs of code trellis.

(b)

(c)

(d)

Figure 2.3: Example illustrating the Viterbi algorithm (Numbers on branch indicate branch metrics; accumulated metrics of surviving paths are given in parentheses).



(a) Punctured trellis for rate-1/2. (b) Rate-2/3 trellis.

Figure 2.4: Rate-2/3 trellis and equivalent punctured trellis.

equal error protection (e.g. with voice and video coding) a family of punctured codes of variable rate may be used. This allows the use of one basic decoder, reducing the area devoted to the decoder in ASIC implementations.

2.2 Trellis coded modulation

Trellis coded modulation (TCM) is a combined coding and modulation technique for digital transmission over band-limited channels. The basic principles of TCM were published in 1982 [31], and some further developments are documented [24]-[30]. In a bandwidth-limited environment, increased efficiency in frequency can be obtained by using a larger size signal constellation, but a larger signal power would be needed to maintain the same signal separation and the same error probability. TCM combines a multilevel modulation scheme with a convolutional code, while the receiver, instead of performing demodulation and decoding in two separate steps, combines the two operations into one.

In classical digital communication systems, the function of modulation and error-correction coding are separated. Modulators and demodulators convert an analog waveform channel into a discrete channel, whereas encoders and decoders correct errors that occur on the discrete channel. In conventional multilevel (amplitude and phase) modulation, the modulator maps m binary bits into one of M (=2^m) possible transmit signals, and the demodulator recovers the m bits by making an independent M-ary nearest-neighbor decision on each signal received. M-AM, M-PSK and M-QAM are examples of the multilevel modulation.

Conventional channel codes operate on binary symbols transmitted over a discrete channel, and Hamming distance is the measure of distance for decoding. When we use maximum likelihood decoding, the optimal distance metric in additive white Gaussian channel is squared Euclidean distance. Thus, in decoding it is desirable to also use Euclidean distance; This results in nearly a 3 dB gain over use of Hamming distance. However, when we use BPSK or QPSK signalling, maximizing the Hamming distance also maximizes the squared Euclidean distance, and thus codes with large Hamming distance are also effective for the Gaussian channel.

If the channel is band limited, we enlarge the signal set of the modulation system, i.e. use multilevel modulation. In this case, independent hard signal decisions prior to decoding may cause a large loss of information, because the Hamming distance between signal labels can not be made proportional to squared Euclidean distance between signals. Thus, maximizing the Hamming distance does not necessarily maximize the squared Euclidean distance of the code. Even if it does, we need soft decisions for decoding and want to combine the coding and modulation process.

Consider one example. Four bit codewords are mapped into 16-QAM signal set as shown in Figure 2.5. As a mapping rule, we use the Gray code, where the labels of the

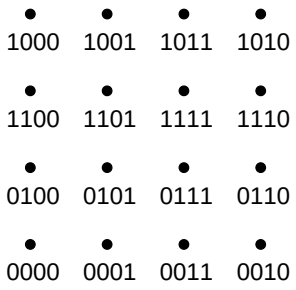


Figure 2.5: Mapping of binary codeword into 16-QAM constellation.

nearest neighbors differ in only one bit. As we can see in the Figure 2.5, the Hamming distances and Euclidean distances of codeword pairs are not always the same. The remedy for this problem is soft-decision decoding, where the decoder operates directly on the unquantized received signal. The idea of using redundant signal sets and directly optimizing the encoder to get the best Euclidean minimum distance were presented by Ungerboeck [31]. He proposed a new way of mapping known as mapping by *set partitioning*. Assume we map m encoded bits into an $M(=2^m)$ point signal set, and label the encoded bits as $(z_{m-1}, z_{m-2}, ..., z_1, z_0)$. Set partitioning divides a signal set into disjoint subsets, called *cosets*, with maximally increasing intra-subset (or intra-coset) distances $\delta_i,\ i = 0, 1, 2, ..., m$. Each partition is two-way and the partition is repeated k times until δ_k is equal to or larger than the desired minimum distance d_{min} of the TCM scheme to be designed. The least significant (LS) bit is assigned in the first partition and the next LS bit is assigned in the second partition. The signal points whose label differs in only the k-th LS bit are at least a distance of δ_{k-1} apart. Therefore, the labels of the signals contain useful information about how far apart the signal points are. This is illustrated in Figure 2.6. In this example the squared intra-coset distances are doubled in each two-way partitioning.

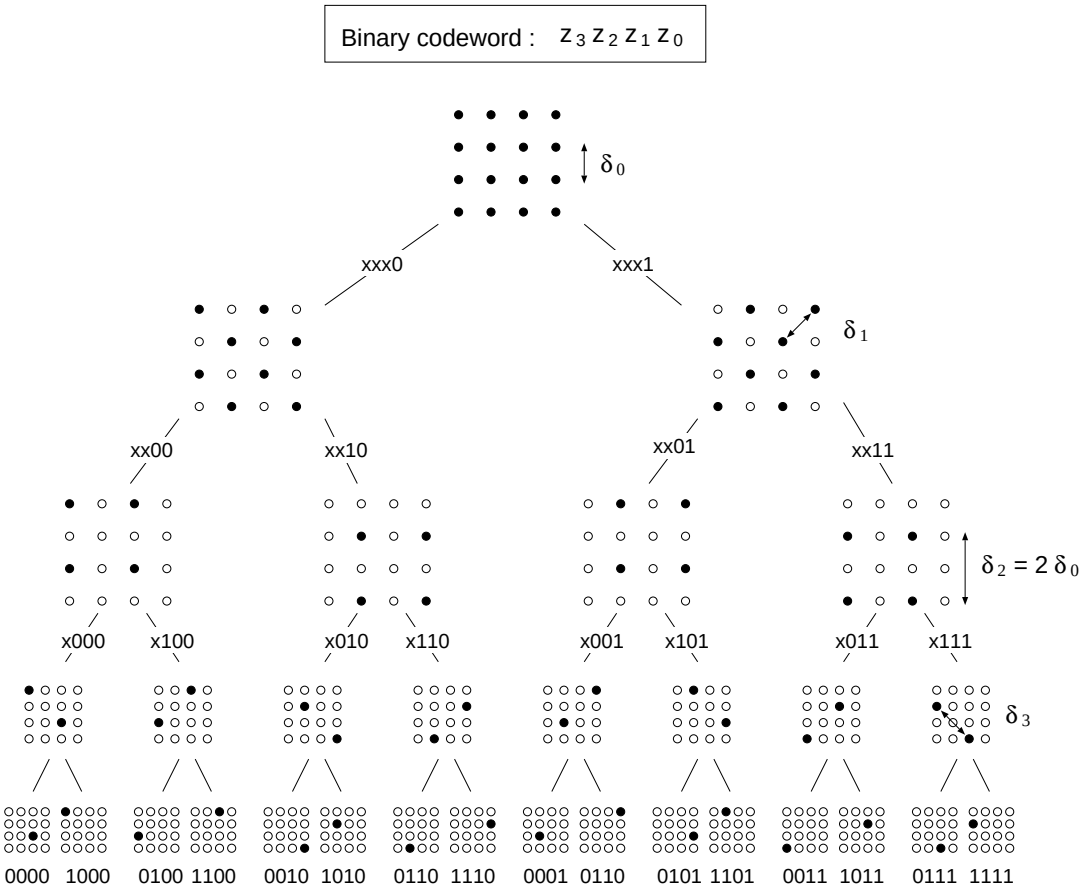


Figure 2.6: Set partition of a 16-QAM constellation [31].

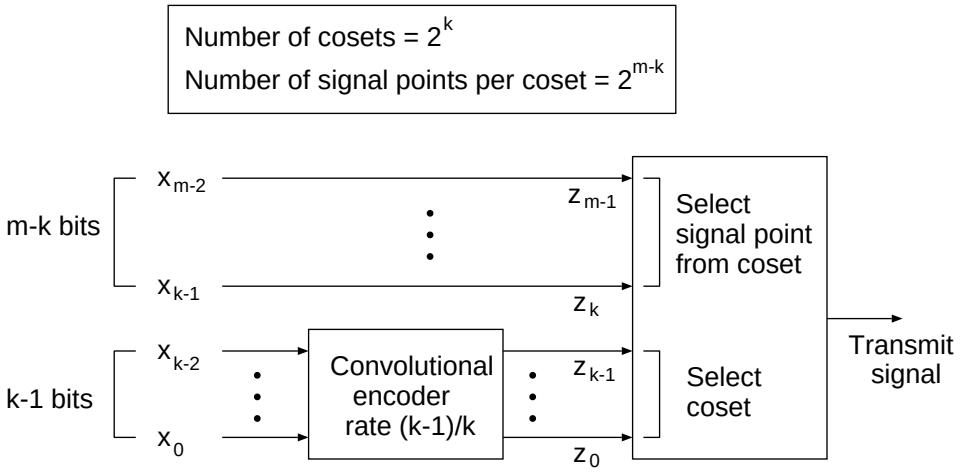
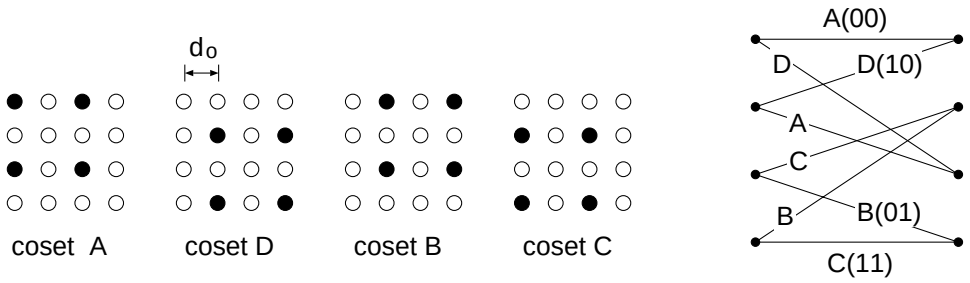


Figure 2.7: Rate-$(k-1)/k$ TCM encoder.

If δ_{k-1} is larger than or equal to d_{min}, the $m-k$ most significant bits do not need protection and can be transmitted without encoding. Rate-$(k-1)/k$ convolutional coding is used for the protection of the remaining bits, where the encoder output determines which of the 2^k cosets are to be transmitted. The remaining $m-k$ bits decide one signal point of the chosen coset. The encoder structure of TCM is illustrated in Figure 2.7. In decoding TCM, the Viterbi decoder decides upon the sequence of cosets, and the uncoded bits are then recovered from the decoded cosets. The branch metric calculation is different from convolutional codes because k output bits are grouped to represent cosets. Thus, the branch metric must be calculated per coset (in a convolutional code, the branch metric is obtained per bit). An example of rate-1/2 TCM on 16-QAM signalling is illustrated in Figure 2.8. There are four cosets $A,\ B,\ C$, and D labeled by two LS bits (z_1, z_0). The output label of the code trellis in TCM is not the binary value but the label of the cosets. The branch metric of coset A is the minimum squared distance between the received signal and the 4 signal points in coset A. The minimum distance d_f among different



(a) Set partition of 16-QAM. (b) Code trellis.

Figure 2.8: Example of rate-1/2 TCM on 16-QAM signalling.

paths in the trellis can be calculated from the code trellis and *inter-coset distance* relations. From Figure 2.8 (a), we obtain the inter-coset distances and intra-coset distance d_c. For example, the squared inter-coset distance of coset pairs (A, B) and (C, D) is $2d_o^2$ and the squared distance of coset pairs (A, C) and (B, D) is d_o^2. The squared intra-coset distance of all the cosets is $4d_o^2$. At the decoder, first we decide upon sequences of transmitted cosets using Viterbi decoding and then we recover the uncoded bits by picking up one of the four members of the chosen coset. Thus, the minimum distance of TCM is the minimum of d_f and d_c. Because we may adds any number of uncoded bits and keep the same basic TCM structure, we shall use the shorthand that rate-k/n TCM implies use of a rate-k/n convolutional code.

There have been many investigations of signal sets defined in more than two dimensions [24] [35]-[38]. The advantages of multi-dimensional signalling are as follows. First, we can pack the signal points more efficiently when the dimensionality is large, and as a result larger d_{min} can potentially be obtained. It also provides a great degree of flexibility in achieving various information rates and in designing rotationally invariant codes [36]. Practically, multi-dimensional signals can be transmitted as sequences of one or two dimensional (1-D or 2-D) signals. Assume we use an uncoded 2^m point constellation. If we want to use TCM, we have to increase the constellation size to

accommodate one more redundant bit. The signal size will be increased from 2^{Nm} to 2^{Nm+1} when we use 2N-D signalling and the signal constellation size per two dimensions is $2^{m+1/N}$. Thus, the signal size expansion per two dimensions decreases as N increases. There is 3 dB constellation size expansion loss when we use 2-D signalling ($N = 1$) and the loss decreases to 1.5 dB or 0.75 dB when we use 4-D or 8-D signalling, respectively.

The need for rotationally invariant trellis codes occurs in coherent detection using suppressed carrier modulation. In QAM signalling, removal of the phase modulation in the phase circuitry in the receiver may cause a phase ambiguity of 90^o. Either we must send special sequences to establish absolute phase, or we need to design trellis codes that are transparent to the phase offset. When we do not use channel coding, *differential encoding*, where the phase differences between successive signals are transmitted, can be used. The operation of differential encoding is explained as follows. Assume we use M-PSK signalling. The phase addition operator $\oplus_p$ and subtraction operator $\ominus_p$ adds and subtracts the two phases of the operands, respectively. Let $x_i,\ i = 0, 1, ...$, be the input sequence, which is then differentially encoded to produce the output sequence $y_i,\ i = 0, 1, ...$, where y_i can be obtained as

$$y_i = x_i \oplus_p y_{i-1} \tag{2.1}$$

Let $\hat{y}_i,\ i = 0, 1, ...$, be the sequence of received signals after some phase offset θ. Then $\hat{y}_i = y_i \oplus_p \theta$. Assume the adjacent transmitted signals y_i and y_{i-1} experience the same phase offset θ, then the recovered input $\hat{x}_i$ is obtained from the phase subtraction of the two adjacent signals. This is explained by the following equations.

$$\begin{aligned} \hat{x}_i &= \hat{y}_i \ominus_p \hat{y}_{i-1} \\ &= (y_i \oplus_p \theta) \ominus_p (y_{i-1} \oplus_p \theta) \end{aligned}$$

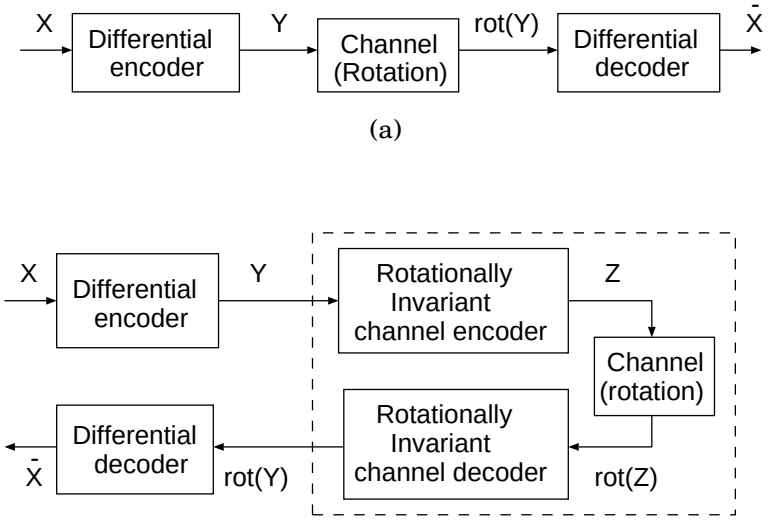


Figure 2.9: Rotationally invariant systems. (a) Ordinary differential encoding/decoding. (b) Rotationally invariant system with channel coders.

$$= \quad ((x_i \oplus_p y_{i-1}) \oplus_p \theta) \ominus_p (y_{i-1} \oplus_p \theta)$$

$$= \quad x_i.$$

When we use channel codes, the problem is much more complicated. A considerable amount of research has been undertaken to design codes which are transparent to phase offsets [9]-[11] [30] [36]. The basic condition for channel codes having rotationally invariant structure can briefly be explained in Figure 2.9. The systems in Figure 2.9 (a) and Figure 2.9 (b) are the same if the dashed block of the Figure 2.9 (b) is the same as phase rotation block of Figure 2.9 (a). This can be restated as follows. If the modulated signal Z is phase rotated, then the decoded signal must be phase rotated version of the input to the encoder Y, i.e. the phase rotation of input to the channel coder is directly related with the phase rotation of the modulated output. This at minimum requires that every 90^o rotation of one code sequence is also a code sequence. Design of effective

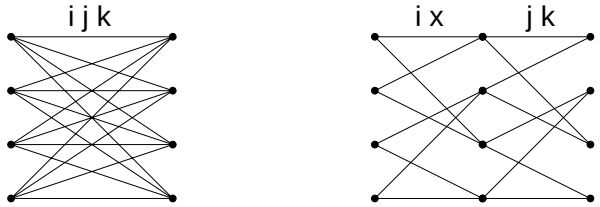
channel codes with this property will be described in chapter 4.

Chapter 3

Variable Rate TCM using Rate-1/2 Trellis

Punctured trellis coded modulation (PTCM) is TCM using a punctured convolutional code. The motivation is the same as in puncturing ordinary convolutional codes, i.e. reduced complexity and the possibility of variable error protection. When we use BPSK or QPSK, we obtain the branch metric per bit. Thus PTCM and punctured convolutional codes have the same decoding procedure. However PTCM has a different branch metric from punctured convolutional codes when we use M-ary QAM or PSK. In this case the branch metric is obtained per symbol and needs to be decomposed when we use a punctured trellis. This is illustrated in Figure 3.1, where we have to decompose the squared Euclidean metric $d^2[ijk]$ into the component metrics $d^2[i]$ and $d^2[jk]$.

In general, use of the punctured structure in the decoder results in a performance loss for trellis codes, due to difficulties in assigning metrics. Thus, care must be taken when puncturing to produce a trellis code which can make good use of soft decisions, when we map more than one encoded bit onto each signal dimension.



(a) Ordinary rate-2/3 trellis. (b) Punctured rate-2/3 trellis.

Figure 3.1: Rate-2/3 trellis and equivalent punctured trellis; i, j, k are binary values and "x" means puncturing position. The label ijk represents one of the eight cosets.

Recently, Chen and Haccoun applied the puncturing technique to TCM for PSK and achieved simplified decoding and code rate flexibility at the expense of a small reduction in the coding gain [2]. However this scheme does not provide satisfactory results for QAM. Another recent application is the *pragmatic punctured* (P^2) trellis code, consisting of two punctured rate-1/2 convolutional codes mapping one bit per dimension. P^2 trellis coded modulation leads to more efficient codes than the non-punctured pragmatic TCM [1]. However, these codes are not rotationally invariant. In the next chapter, we will investigate two methods of designing 90^o rotationally invariant PTCM, which retain the optimal branch metric property.

We provide constructions for punctured rate-2/3 codes based on decomposition of the metric into orthogonal components. These show no loss in performance for trellis coded QAM and PSK. In section 1, we discuss PTCM for QAM. We describe how to construct the punctured trellis and signal constellation, and describe the decoding methods. PTCM for PSK is described in section 2. Simulation results and their interpretation are presented in section 3.

3.1 Variable rate TCM using rate-1/2 trellis

In the following, for illustrative purposes we assume that a coded 32 point QAM signal constellation is used to achieve an effective transmission rate of 4 bits per channel symbol for two-dimensional codes, and 4.5 bits per symbol for four-dimensional codes.

3.1.1 Rate-1/2 PTCM

Rate-1/2 TCM has 4 cosets and each coset has 8 members. Let the four cosets be $a,\ b,\ c$ and d, with bit labels 00, 01, 10 and 11 respectively. The trellis for TCM is obtained from the trellis of the binary convolutional code by replacing the bit labels of the branches by the corresponding coset labels.

Since squared Euclidean distances among cosets should match the Hamming distances of the original binary rate-1/2 code, we let the minimum squared distance between cosets a, d and the minimum squared distance between cosets b, c be $2d_0^2$. The resulting optimal coset partitioning in 32-QAM is illustrated in Figure 3.2. The squared

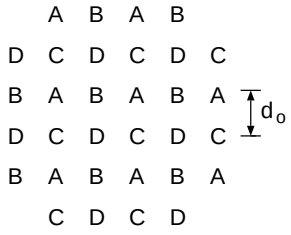


Figure 3.2: 32-QAM constellation for rate-1/2 TCM.

intra coset distance is $4d_0^2$, which limits the gain of the trellis code to 3 dB if used in an equal-error protection scheme. Increasing the number of states beyond four will only serve to further protect one bit per symbol. Thus higher rates are of interest.

3.1.2 Rate-2/3 PTCM

Rate-2/3 TCM has eight cosets and each coset has four members. There are two input bits in one symbol interval and so two branches are assigned to each symbol in the punctured trellis. Since we need 3 output bits, we puncture one of the 4 positions.

In the following, we present good puncturing patterns, a simple method for obtaining two branch metrics from one optimal branch metric, and a coset partitioning for 32-QAM constellations suitable for use with PTCM. The solution results in almost the same performance as the best rate-2/3 TCM reported in the literature [24].

3.1.2.1. Basic constraints of PCC trellis design

In a rate-2/3 trellis, there are 4 in-going branches and 4 out-going branches for each state. A rule of thumb [31] is that the distances among these branches should be as large as possible to produce the maximum free distance. We will later show that it is only for such trellises that Viterbi decoding on the punctured trellis results in optimal soft-decision decoding.

Let cosets "a, c, e, g" constitute group G_0, "b, d, f, h" be the group G_1. The set G_0 is a subset of lattice RZ^2 while G_1 is a subset of lattice $RZ^2 + d_0$. These lattices are depicted in Figure 3.3.

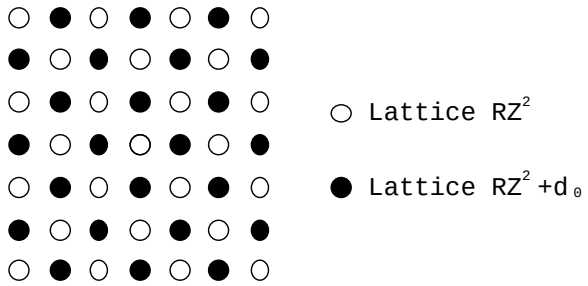


Figure 3.3: Lattice illustration.

Table 3.1: Binary representations of 8 cosets.

cosets	binary labels
a	000
b	001
c	010
d	011
e	100
f	101
g	110
h	111

In-going branches and out-going branches should be drawn from one of these two groups. The binary coset labels are in Table 3.1. Cosets in G_0 and G_1 have values 0 and 1 in the last binary digit, respectively. In a punctured trellis, there are two in-going branches and two out-going branches. The merging of two steps of the punctured trellis should result in a rate-2/3 trellis with maximum separation of in-going and out-going branches at each state. To accomplish this, the following constraints should be observed in puncturing.

In a punctured trellis, every two branches are combined to have one coset output. "*First step*" is defined as a group of all the first branches of the two, and "*second step*" is defined as a group of all the second branches of the two.

Constraint 1 The second digit of two in-going branches in the second step should have the same value to make cosets of the same group go into the same state.

Some trellises which satisfy *constraint 1* are depicted in Figure 3.4. "K_0" branches are defined as branches at the second step which have 0 valued second digit, and "K_1"

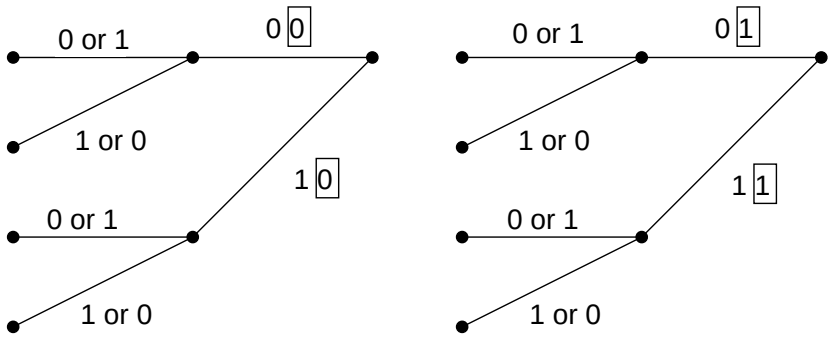
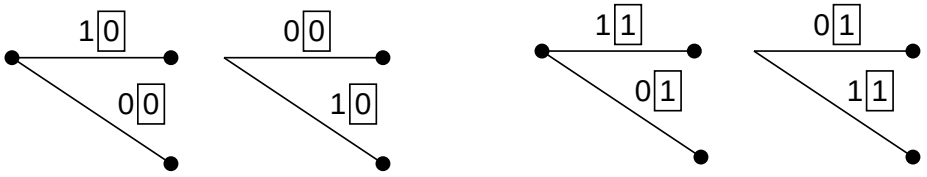


Figure 3.4: Sample sub-trellises which satisfy *constraint 1*.

branches are defined as branches at the second step which have 1 valued second digit.

Constraint 2 The second digit of two out-going branches in the second step should have the same value, and out-going branches at the first step should connect the same kind of out-going branches at the second step to make cosets of the same group stem from the same state.

Figure 3.5 illustrates K_0 and K_1 branches and Figure 3.6 shows some trellises which satisfy *constraint 2*.



(a) K_0 out-going branches. (b) K_1 out-going branches.

Figure 3.5: K_0 and K_1 out-going branches.

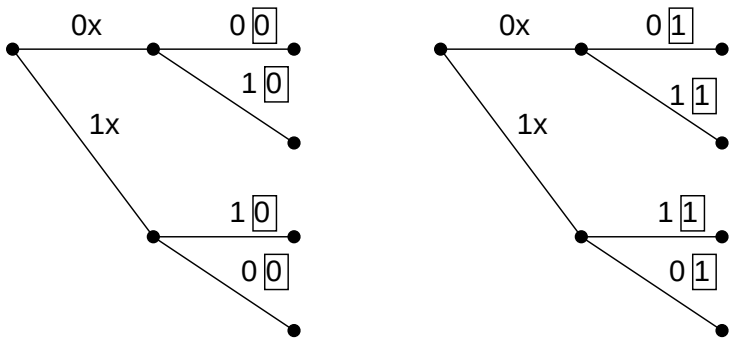


Figure 3.6: Sample sub-trellises which satisfy *constraint 2*.

3.1.2.2. Optimal puncturing pattern

By *constraint 1*, the second digit of in-going branches in the second step should have the same value, which implies that the first digits of in-going branches in the second step cannot have the same value. Thus we cannot puncture output digits at the second step. Since the binary outputs of the rate-1/2 trellis at the first step are the same as that of the second step, the second bits of in-going branches in the first step also have the same values, and so we puncture them.

3.1.2.3. Obtaining branch metrics at the punctured trellis

We now outline the conditions for which that there is no branch metric distortion in rate $\frac{m}{m+1}$ PTCM. A sequence of m outputs in the punctured trellis represents one coset.

Condition 1 (justification of pre-decision) Each step branch metric is independent of the previous step decision results.

In Viterbi decoding, we must decide on the most probable branch at every step. If

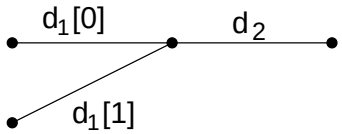
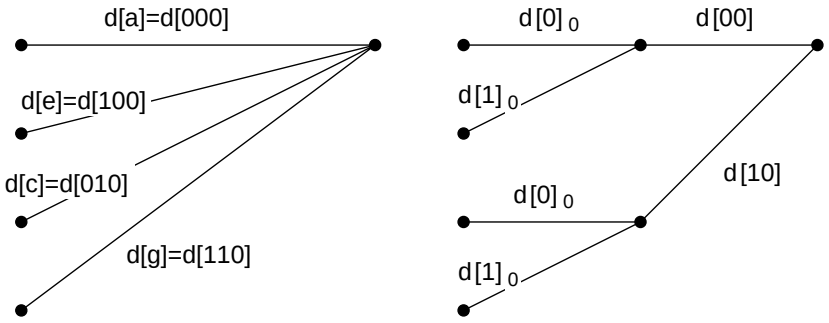


Figure 3.7: Example for illustration of *condition 1*.

condition 1 is not satisfied, the decision at every step is not optimal. For example, suppose we have two branches with $d_1[0]$ and $d_1[1]$ as the first step branch metrics, and with d_2 as the second step branch metric. This is illustrated in Figure 3.7. Assume *condition 1* is not satisfied. If the upper branch is selected, let d_2 have value $d_2[0]$. Otherwise, let d_2 have value $d_2[1]$. When $d_1[0]$ is smaller than $d_1[1]$, we select the upper branch at the first step as the more probable route and the branch metric at the end of the second step is $d_1[0] + d_2[0]$. If the value of $d_2[1]$ is much smaller than $d_2[0]$, then $d_1[1] + d_2[1]$ could be smaller than $d_1[0] + d_2[0]$, and an error results.

Condition 2 The summation of m step squared branch metrics in a punctured trellis is the squared branch metric of the corresponding coset.

In a rate-2/3 trellis, there are four in-going branches. The outputs of these four branches are G_0 cosets a, c, e, g or G_1 cosets b, d, f, h. Without loss of generality, we assume G_0 cosets. One example of rate-2/3 TCM branches and the equivalent punctured branches is illustrated in Figure 3.8. The branch metrics $d[0]_0$, $d[1]_0$, $d[00]$ and $d[10]$ in Figure 3.8 must be expressed in terms of $d[a]$, $d[c]$, $d[e]$ and $d[g]$. The subscript "$_0$" in $d^2[0]_0$ and $d^2[1]_0$ means the branch in the first step connects K_0 branches in the second step. $d^2[0]_1$ and $d^2[1]_1$ are squared branch metrics of the first step when the branch in the first step connects K_1 branches in the second step. The values of these metrics will be obtained later.



(a) Rate-2/3 TCM branches. (b) Equivalent punctured branches.

Figure 3.8: Example of TCM branches and equivalent punctured branches.

The two dimensional received signal vector $\vec{r}$ may be represented as the sum of two component vectors,

$$\vec{r} \;=\; r_1\vec{u_1} + r_2\vec{u_2} \tag{3.1}$$

where $\vec{u_1}$ and $\vec{u_2}$ are mutually orthogonal unit vectors. We define $\vec{d_\xi}$ as

$$\vec{d_\xi} \;\equiv\; \vec{r} - \vec{C_\xi} \tag{3.2}$$

where ξ represents an arbitrary coset and C_ξ is the nearest point of coset ξ to the received signal r. $\vec{C_\xi}$ can be represented as

$$\vec{C_\xi} \;=\; C_{\xi 1}\vec{u_1} + C_{\xi 2}\vec{u_2} \tag{3.3}$$

The squared metric for coset ξ, $d^2[\xi]$, is obtained as follows.

$$\begin{aligned} d^2[\xi] &= |\vec{d_\xi}|^2 \;=\; |\vec{r} - \vec{C_\xi}|^2 \\ &= |\,(r_1 - C_{\xi 1})\vec{u_1} + (r_2 - C_{\xi 2})\vec{u_2}\,|^2 \\ &= (r_1 - C_{\xi 1})^2 + (r_2 - C_{\xi 2})^2. \end{aligned} \tag{3.4}$$

Since $d^2[a]$ is the sum of $d^2[0]_0$ and $d^2[00]$ (by *condition 2*) and $d^2[00]$ must be obtained independently (by *condition 1*), we can obtain the branch metric of the punctured

trellis as follows.

$$\begin{aligned} d^2[0]_0 &\equiv (r_1 - C_{a1})^2 \text{ and } d^2[00] \equiv (r_2 - C_{a2})^2 \\ d^2[0]_0 &\equiv (r_1 - C_{c1})^2 \text{ and } d^2[10] \equiv (r_2 - C_{c2})^2 \\ d^2[1]_0 &\equiv (r_1 - C_{e1})^2 \text{ and } d^2[00] \equiv (r_2 - C_{e2})^2 \\ d^2[1]_0 &\equiv (r_1 - C_{g1})^2 \text{ and } d^2[10] \equiv (r_2 - C_{g2})^2 \end{aligned} \tag{3.5}$$

From the above equations we find that

$$C_{a1} = C_{c1},\ C_{e1} = C_{g1},\ C_{a2} = C_{e2},\ \text{and}\ C_{c2} = C_{g2}$$

Cosets a and c have the same coordinate in $\vec{u_1}$, which means that the line connecting cosets a and c is perpendicular to the unit vector $\vec{u_1}$ and parallel to $\vec{u_2}$. Similar conditions determine how the constellation is partitioned into cosets, so that rate-2/3 PTCM has no branch metric distortion when the outputs of in-going branches and out-going branches are cosets in G_0. Results for G_1 are similar. The coset partitioning of a 32 QAM constellation with no branch metric distortion is shown in Figure 3.9. In this case, the two orthogonal unit vectors $\vec{u_1}$ and $\vec{u_2}$ are

$$\vec{u_1} \equiv -\frac{1}{\sqrt{2}}\vec{a_x} + \frac{1}{\sqrt{2}}\vec{a_y}, \quad \vec{u_2} \equiv \frac{1}{\sqrt{2}}\vec{a_x} + \frac{1}{\sqrt{2}}\vec{a_y} \tag{3.6}$$

where $\vec{a_x}$ and $\vec{a_y}$ are unit vectors along the horizontal and vertical axes, respectively. The branch metric calculation has a simple geometric interpretation. Assume that outputs in some trellis branches are members of G_0 cosets. $d[0]_0$ is the distance between the received signal and the nearest line connecting cosets $a-c$. $d[1]_0$ is the distance between the received signal and the nearest line connecting cosets $e-g$. $d[00]$ is the distance between the received signal and the nearest line connecting cosets $e-a$. $d[10]$ is the distance between the received signal and the nearest line connecting cosets $g-c$.

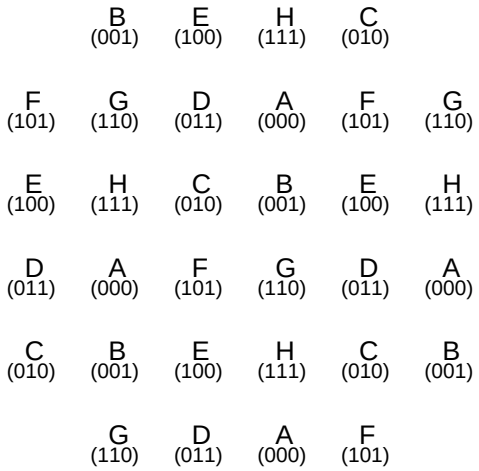


Figure 3.9: 32-QAM signal constellation with optimal coset partitioning.

Practically we can obtain branch metrics by the following procedure.

First, obtain two components r_1 and r_2 of the received signal.

$$r_1 \;=\; \vec{r}\cdot\vec{u_1}\,, \quad r_2 \;=\; \vec{r}\cdot\vec{u_2} \tag{3.7}$$

Second, find branch metrics of all the possible cases.

$$\begin{aligned}
d^2[0]_0 \;&=\; (r_1 - C_{a1})^2, \;\; d^2[1]_0 \;=\; (r_1 - C_{e1})^2 \\
d^2[0]_1 \;&=\; (r_1 - C_{b1})^2, \;\; d^2[1]_1 \;=\; (r_1 - C_{f1})^2 \\
d^2[00] \;&=\; (r_2 - C_{a2})^2, \;\; d^2[10] \;=\; (r_2 - C_{c2})^2 \\
d^2[01] \;&=\; (r_2 - C_{b2})^2, \;\; d^2[11] \;=\; (r_2 - C_{d2})^2
\end{aligned} \tag{3.8}$$

Finally, we use $d^2[0]_0$ and $d^2[1]_0$ as branch metrics for the first step if the branch in the first step connects K_0 branches in the second step. Otherwise, we use $d^2[0]_1$ and $d^2[1]_1$ as branch metrics for the first step.

The branch metric computation has similar complexity to that of original form of TCM. However there is a small performance degradation if we do not take into account

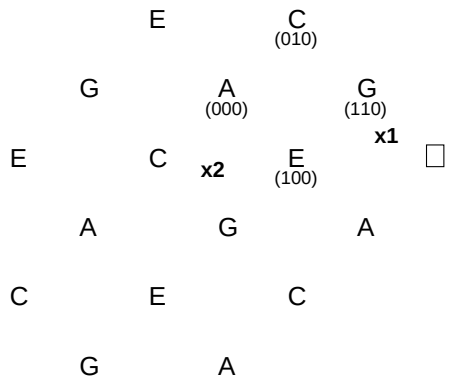


Figure 3.10: Illustration of boundary effect.

the boundary effects in the signal constellation. This is illustrated in Figure 3.10, which depicts the G_0 signal points in 32-QAM. When the received signal is **x2**, branch metrics of cosets $a,\ c,\ e$ and g are correctly obtained by using equation (3.8). However, when the received signal is **x1**, the branch metric of coset c is not the correct value, because the method in equation (3.8) assumes that there is another point of coset c at □, and the branch metric obtained for coset c is smaller than the real one. So we can not use equation (3.8) to obtain branch metrics when the received signal is at the boundary of the constellations. To overcome this effect, we could assign the branch metrics at the second steps according to the decision results in the first steps, which requires more operations and some performance degradation. However, simulations reveal the degradation to be negligible, as illustrated in Figure 3.11. Practically, we can use modulo operations in branch metric calculations to simplify either TCM or PTCM. PTCM with independent step branch metrics has the same performance as TCM, when both employ the modulo reduction technique.

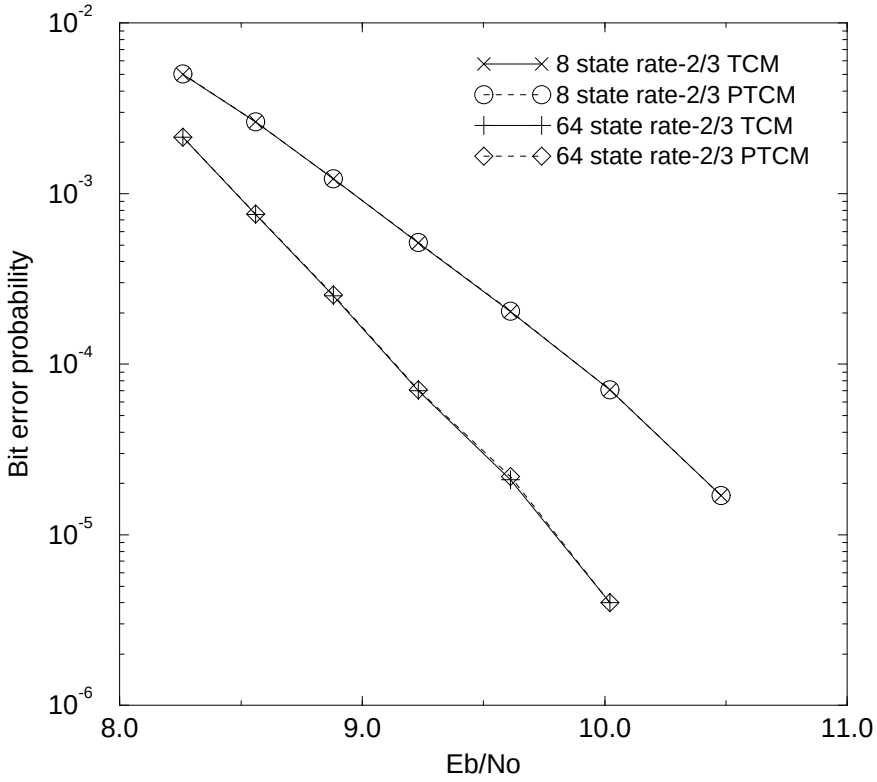


Figure 3.11: Performance comparisons with rate-2/3 TCM and PTCM.

Table 3.2: Performance comparisons of rate-2/3 TCM.

states	Our scheme			Best known code [24]	
	Generators(octal)	d_f^2	N_f	d_f^2	N_f
8	13,2	5	16	5	16
16	31,2	6	56	6	56
32	75,10	6	16	6	8
64	145,4	7	48	7	40

3.1.2.4. Finding generator sequences

We now present our code search results. For a rate-1/2 trellis, the Hamming distance between two branches going into and stemming from the same state should be 2 to guarantee maximum free distance. To satisfy *constraint 1*, we change this rule. Since the second bits of two in-going branches and two out-going branches are the same, we must change the outputs of the original rate-1/2 trellis.

We performed a computer search over all rate-2/3 codes satisfying *condition 1* and *condition 2*. Table 3.2 presents the codes with largest minimum squared distance of the trellis d_f^2 and minimum number of nearest neighbors N_f. As we can be seen, our codes have almost the same performance as the best known codes.

3.1.3 Multidimensional PTCM.

The procedures for designing rate-2/3 PTCM are easily generalized to N dimensional signal sets. The received signal $\vec{r}$ is represented by N mutually orthogonal vectors.

$$\vec{r} = \sum_{i=1}^{N} r_i \vec{u}_i \tag{3.9}$$

where $\vec{u_i}$'s are mutually orthogonal unit vectors. Then

$$\vec{C_\xi} \;=\; \sum_{i=1}^{N} C_{\xi i}\vec{u_i} \tag{3.10}$$

The squared metric for coset ξ, $d^2[\xi]$ is

$$d^2[\xi] \;=\; \sum_{i=1}^{N}(r_i - C_{\xi i})^2 \tag{3.11}$$

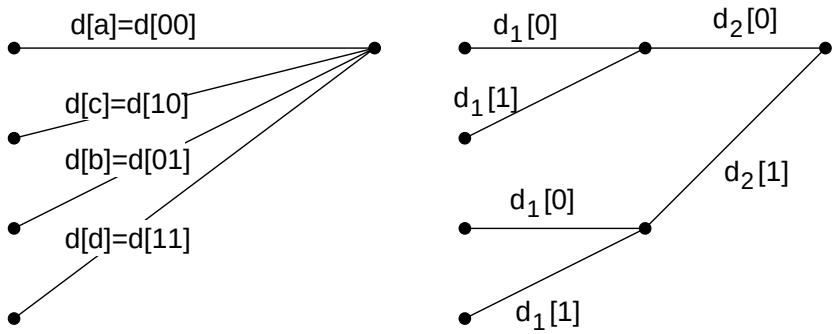
We must decompose $d^2[\xi]$ into m branch metric components. To satisfy *condition 1* and *condition 2*, N should be no less than m. As a result, we need more than two dimensional signalling when m is larger than two (i.e. the code rate is larger than $\frac{2}{3}$).

Wei-type multidimensional TCM achieve high coding gain with rotational invariance [36]. However applying puncturing to Wei-type multidimensional TCM is not possible because the particular coset merging procedure used violates *condition 1*. However, there are many other situations in which we can implement PTCM for QAM signalling with arbitrary m which satisfies the two conditions.

3.1.3.1. Rate-3/4 PTCM

With punctured rate-3/4 codes, it is not possible to avoid branch metric distortion when two dimensional signal sets are used. To avoid branch metric distortion, we propose 4 dimensional signalling. We transmit two consecutive 32 QAM signals, with 4 cosets $a,\ b,\ c,\ d$ in each constellation. Three steps of the punctured trellis are assigned per 2 symbols. Since we puncture 2 of 6 output bits, one step is not punctured and the other two steps are punctured. The non-punctured step has 2 bits which represent 4 cosets and assigned to one signal. The other two punctured steps are combined to represent 4 cosets, and assigned to one signal.

At the non-punctured first step, two bits represent all the 4 cosets and there is no problem in branch metric calculation. On the contrary, at the punctured second and



(a) merged trellis of second and third step (b) second and third step branches

Figure 3.12: Punctured second and third step branches and equivalent merged trellis.

third step we should assign branch metrics at each step, i.e. we obtain $d_1[0],\ d_1[1],\ d_2[0]$ and $d_2[1]$ from $d[a],\ d[b],\ d[c]$ and $d[d]$. This is illustrated in Figure 3.12. By the same procedure of rate-2/3 case, we can do this without any branch metric distortion. From equation (3.4), we can derive that

$$\begin{aligned} d^2[a] &= d_1^2[0] + d_2^2[0] = (r_1 - C_{a1})^2 + (r_2 - C_{a2})^2. \\ d^2[b] &= d_1^2[0] + d_2^2[1] = (r_1 - C_{b1})^2 + (r_2 - C_{b2})^2. \\ d^2[c] &= d_1^2[1] + d_2^2[0] = (r_1 - C_{c1})^2 + (r_2 - C_{c2})^2. \\ d^2[d] &= d_1^2[1] + d_2^2[1] = (r_1 - C_{d1})^2 + (r_2 - C_{d2})^2. \end{aligned} \tag{3.12}$$

From the above relations we find that

$$C_{a1} = C_{b1},\ C_{a2} = C_{c2},\ C_{c1} = C_{d1},\ C_{b2} = C_{d2}.$$

Thus $d_1[0]$ is the distance between the received signal and the nearest line connecting cosets $a - b - a - b$. $d_1[1]$ is the distance between the received signal and the nearest line connecting cosets $c - d - c - d$. $d_2[0]$ is the distance between the received signal and the nearest line connecting cosets $a - c - a - c$. $d_2[1]$ is the distance between the received signal and the nearest line connecting cosets $b - d - b - d$. This structure is

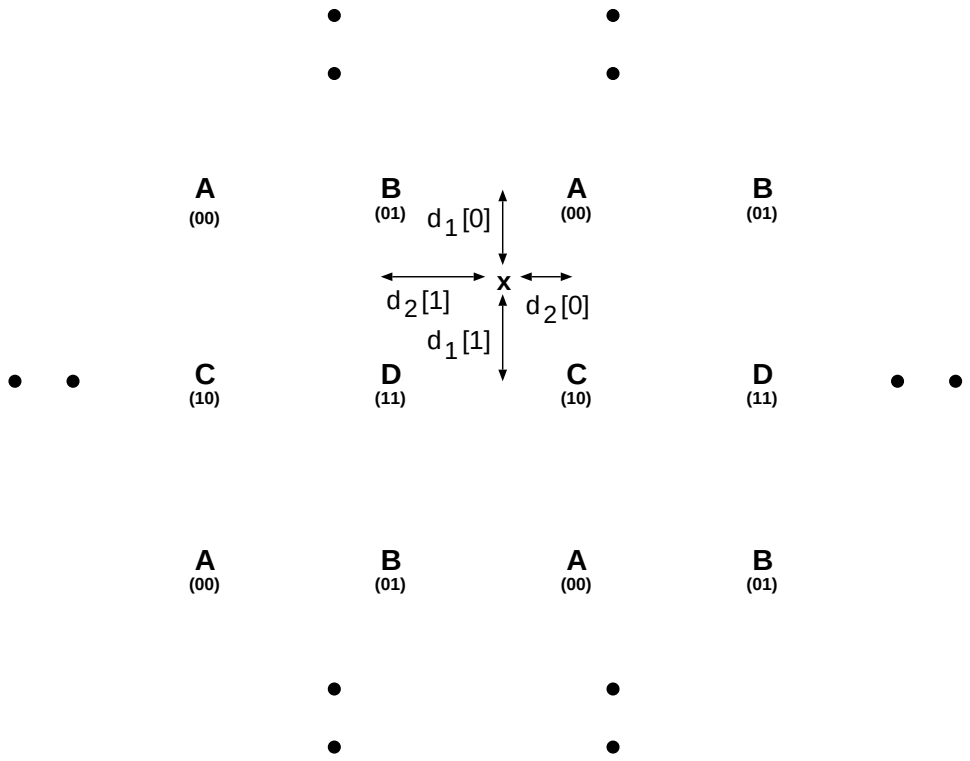


Figure 3.13: Illustration of branch metrics in second and third step branches.

illustrated in Figure 3.13. The signal points with 0 as a first digit are in one line and the signal points with 1 as a first digit are in another line. In the same way, the signal points with 0 as a second digit are in one line and the signal points with 1 as a second digit are in another line. In this case, the unit orthogonal vectors $\vec{u_1}$ and $\vec{u_2}$ are expressed as

$$\vec{u_1} = \vec{a_y}, \quad \vec{u_2} = \vec{a_x}.$$

We can observe in Figure 3.13 that the relations in equation (3.12) are satisfied. This must be true because $\vec{u_1}$ and $\vec{u_2}$ are orthogonal in two dimensional space. The optimal generator and punctured position in the 4D rate-3/4 PTCM scheme may be obtained from the literature on optimal rate-3/4 punctured convolutional codes [15], since for this geometry Euclidean and Hamming distances are simply related. The optimal rate-3/4

Table 3.3: Generators(in octal) for the best rate-3/4 punctured code (0 means punctured position).

number of states	generators	punctured position*	d_f^2	N_f
8	15,17	11-10-01	4	464
16	25,37	11-01-01	4	128
32	43,65	11-01-01	5	512
64	121,173	11-01-01	5	192

punctured convolutional codes are in Table 3.3. This construction is the same as the P^2 TCM proposed in [1]. Since there are 4 cosets in one constellation, the intra coset distance is $2d_0$ which is smaller than $2\sqrt{2}d_0$ in two-dimensional rate-2/3 codes. This limits the coding gain when the number of states is larger than 16. However, rate-3/4 PTCM with four-dimensional signalling has a throughput of 4.5 bits per symbol while rate-1/2 and rate-2/3 with two-dimensional signalling have a throughput of 4 bits per symbol: we may alternatively view this reduced coding redundancy as a 1.5 dB gain [24].

3.1.3.2. High rate PTCM

We may also construct punctured rate-4/5 codes on four-dimensional signal sets by using an 8-way partition on one of the two 2-dimensional signal sets, and applying the rules of the previous sections. The extension to higher dimensional sets is obvious, although the increased values of N_f makes this of doubtful utility.

3.2 Application to PSK

Chen and Haccoun have provided a PTCM construction for PSK. However they employ an approximation which reduces the metric separation, leading to slightly degraded error performance [2]. In this section, we apply our procedure for obtaining optimal PTCM in QAM to PSK.

Assume that we use 8-PSK. Each signal point represents one coset because there are 8 cosets. We classify the 8 signal points into 2 groups (G_0 and G_1) as in Figure 3.14.

Then *constraint 1* and *constraint 2* should be satisfied and so that the puncturing position is the same as that of QAM. The branch metric and optimal coset partitioning can be obtained by the procedure we have used in QAM. As a result we change the order of signal points from $a,\ c,\ e,\ g$ to $a,\ c,\ g,\ e$ and from $b,\ d,\ f,\ h$ to $b,\ d,\ h,\ f$. Newly ordered signal points in each group and the corresponding branch metrics are depicted in Figure 3.14, where the notation for branch metrics is the same as in section 1.2. In this scheme, *condition 1* and *condition 2* are satisfied and so there is no performance degradation due to puncturing. Now consider the optimal generator. A circular shift

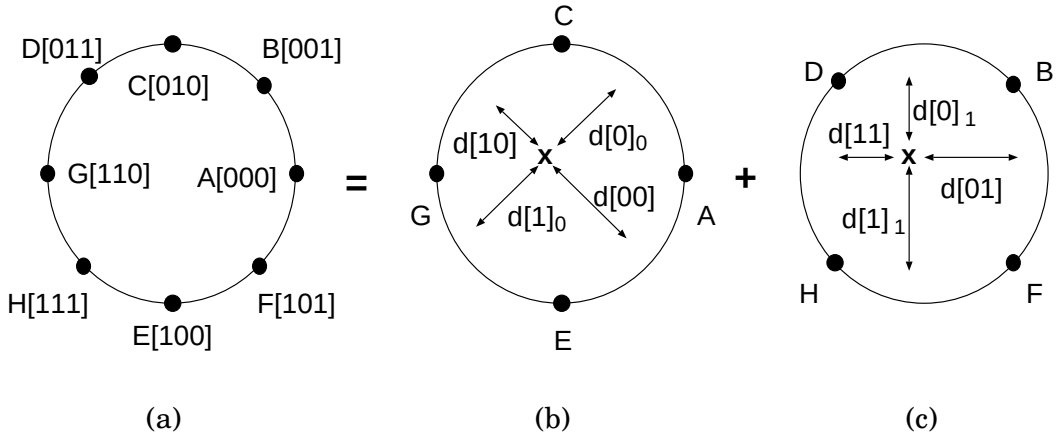


Figure 3.14: Branch metric calculations in each group. (a) 8-PSK signal points. (b) Branch metrics of G_0 signals. (c) Branch metrics of G_1 signals.

Table 3.4: Generators(in octal) for rate-2/3 PTCM with 8-PSK signalling.

states	G_1 shift	generators	d_f^2	N_f	d_f^2 Ungerboeck [31]
8	90^o	15,2	4.59	1	4.59
16	180^o	31,4	5.17	1	5.17
16	0^o	23,4	5.76	6	5.17

of G_1 signals does not change the above conditions, but it does change the distance structures among all the cosets. So we fix the G_0 signals as a reference and make circular shifts ($90^o,\ 180^o,\ 270^o$ and zero shifts toward counterclockwise) of G_1 signals to create a new labeling. In each shift, we tried all the possible generators and found the best one. As a result, we obtained the optimal generator sequences and circular shifts of the G_1 signals. The results are in Table 3.4.

3.3 Complexity of PTCM

It is also instructive to compare the complexity of traceback and ACS operations in punctured codes for the different rates. These are exactly the same for all the members of the family, with the arithmetic precision in the ACS units also having the same requirements. However, the traceback depth is greater in the straightforward implementation for the punctured code than treating it as an ordinary rate-k/n code. This is because we must store transitions for each of the k binary branches in order to re-use the same traceback unit for all members of the family. We need traceback depth of kL when the traceback depth of the ordinary decoder is L. However, we can do much better with slight modifications to the path storage procedure. If the number of states is m, then the traceback depth can be reduced to L in the punctured trellis by adding m memories. This is explained as follows.

Define one unit of the rate-k/n punctured trellis be k steps through the trellis. This is identical to one step of an ordinary rate-k/n trellis. We note that in a rate-k/n punctured trellis, if the number of state is no less than 2^k, we know the input and output sequence of one unit (k step trellis) from the information of the first and the last state in the unit. This follows from the observation that there are 2^k possible routes from the first state to the end of one unit. Therefore, there is a one-to-one mapping between the first and the last states of one unit when the number of states is no less than 2^k. Consequently, we do not need the intermediate state transition in the traceback process if the number of state is no less than 2^k. We need m memories to trace the state transitions in one unit. We put the state transition into the traceback memory every k-th step and do the traceback process and decode k information bits. As a result, the traceback depth of the punctured trellis is the same as that of the corresponding ordinary trellis.

A rate-k/n code has 2^k branches leaving or entering each state, and requires 2^n different branch metrics to be computed. A punctured code of the same rate is treated as k steps of a binary-branching trellis to perform the basic computation. Thus we must perform $m2^k$ ACS operation for the regular form of the trellis, and $2km$ operations in the punctured form. Thus, treating the code in its punctured form results in considerable complexity savings in metric accumulation and branch comparison operations for higher rate codes.

3.4 Simulation and discussion

We have designed rate-2/3 and rate-3/4 PTCM for both QAM and PSK. We found the optimal position for puncturing, generators for the rate-1/2 trellis, branch metrics and coset positioning in QAM and PSK. As a result, we obtained very good rate-2/3 PTCM

Table 3.5: Asymptotic coding gains (in dB) of PTCM (QAM signalling).

number of states	rate-1/2	rate-2/3	rate-3/4
8	3.01	3.98	4.52
16	3.01	4.77	4.52
32	3.01	4.77	4.52
64	3.01	5.44	4.52

codes. Table 3.5 lists the asymptotic coding gains for PTCM signals. Rate-3/4 PTCM has 1.5 dB redundancy gain over the other schemes because there is one redundant bit per 4-D signal. The simulation results for rate-1/2 TCM, rate-2/3 TCM,PTCM and rate-3/4 PTCM with 32-QAM are shown in Figure 3.15, Figure 3.16 and Figure 3.17.

As expected, rate-1/2 TCM has about 3 dB gain over uncoded 16-QAM (6 dB gain due to increased squared minimum distance and 3dB constellation expansion loss), and rate-2/3 TCM has increasing gains as the number of states becomes large and as the required probability of error decreases. Rate-2/3 PTCM has the same performance as that of rate-2/3 TCM using the same code, since there is no distortion in the branch metric calculation in PTCM. Rate-3/4 PTCM shows increased performance as the number of states increases. Even though the asymptotic coding gain of rate-3/4 PTCM are all the same independent of the number of states as we can see in Table 3.5, the decreased path multiplicity at increased number of states gives better actual coding gain. The simulation results for rate-2/3 TCM, PTCM with 8-PSK is shown in Figure 3.18, where TCM and PTCM schemes have the same performance.

In general, whenever the metric can be decomposed into orthogonal components a punctured code can be constructed which apart from negligible boundary effects has no performance loss with respect to the equivalent original TCM scheme.

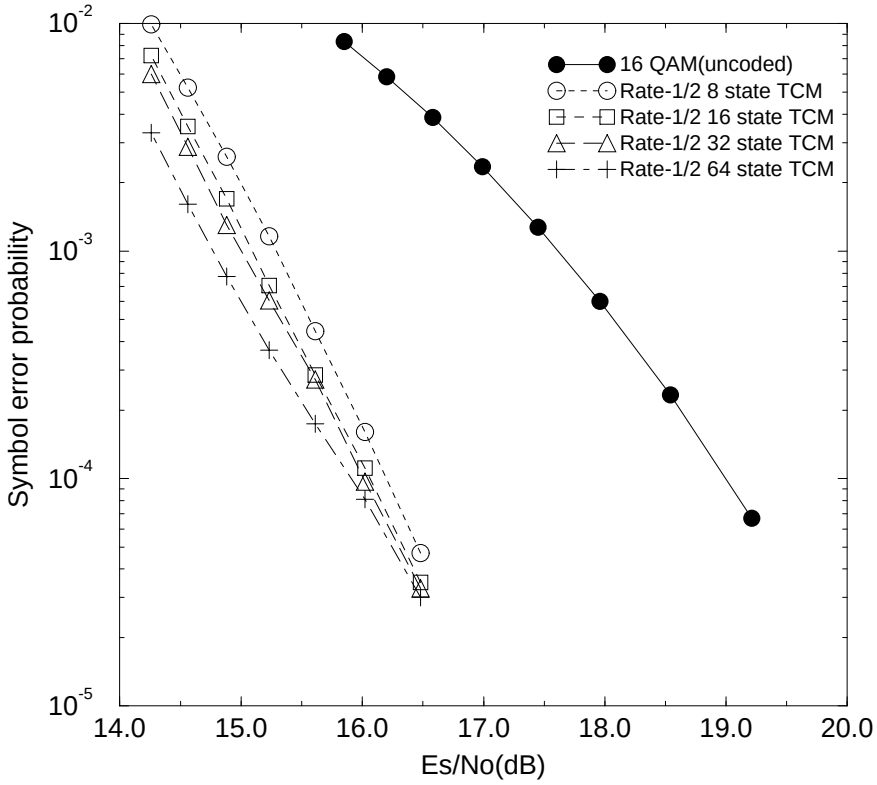


Figure 3.15: Simulation results of uncoded 16-QAM and rate-1/2 TCM (32-QAM signal).

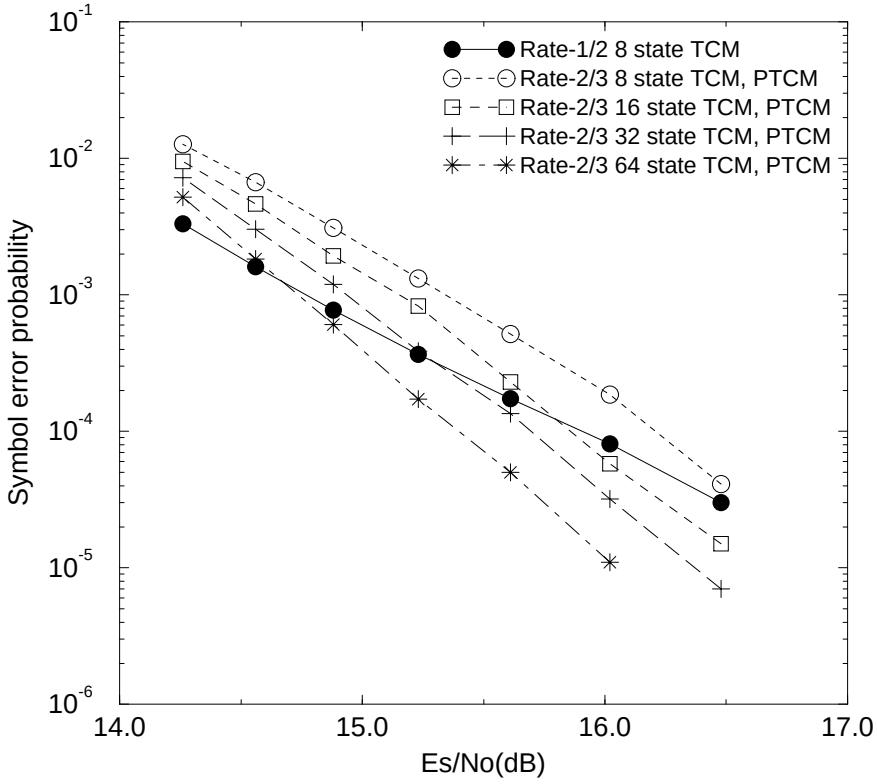


Figure 3.16: Simulation results of rate-1/2 TCM and rate-2/3 TCM, PTCM (32-QAM signal).

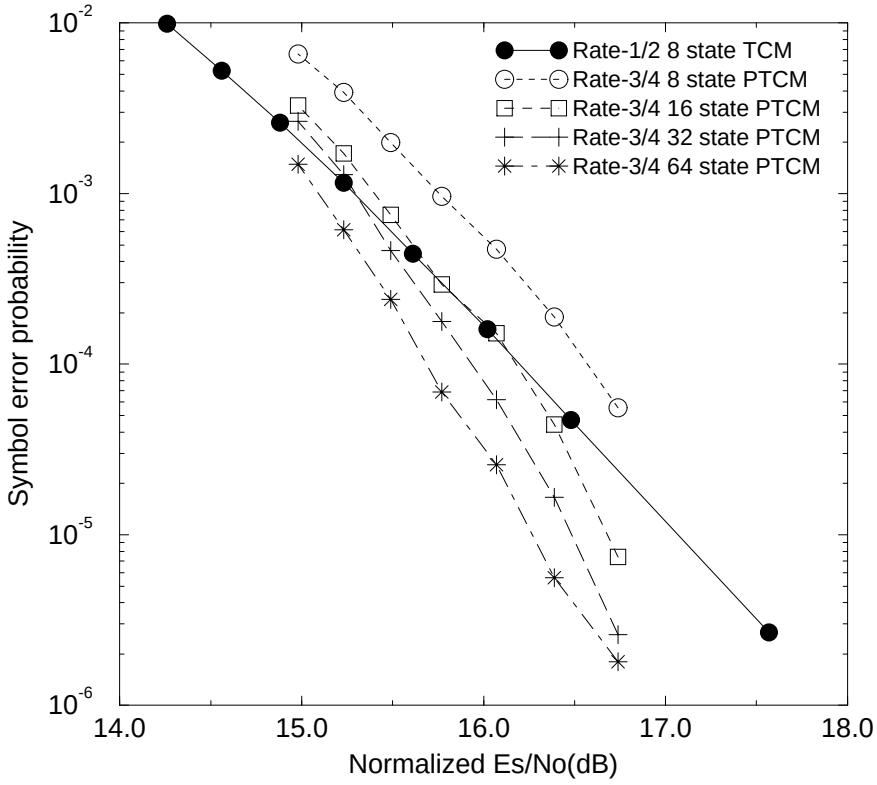


Figure 3.17: Simulation results of rate-1/2 TCM and rate-3/4 PTCM (32-QAM signal): SNR is normalized due to the 0.5 bit redundancy gain of rate-3/4 PTCM.

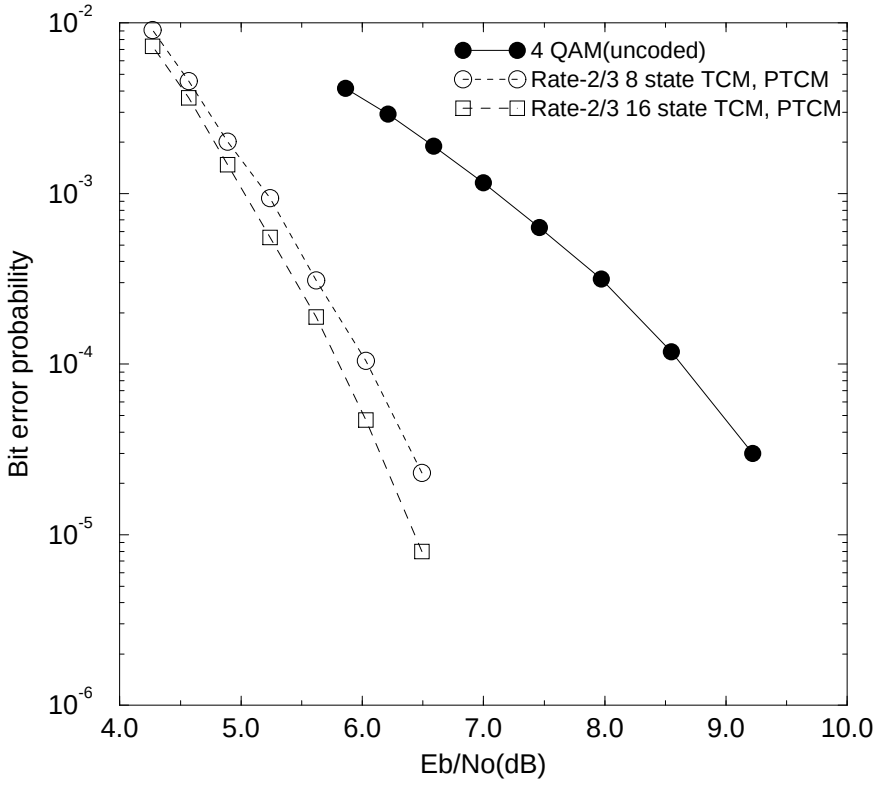


Figure 3.18: Simulation results of uncoded QPSK and rate-2/3 TCM, PTCM (8-PSK signal).

Chapter 4

Rotationally Invariant Punctured TCM

To compensate for a phase ambiguity in the receiver, there are two basic approaches. We can estimate the phase ambiguity by sending a fixed sequence of modulation phases to initialize data communication. On the other hand, we can design trellis codes that are transparent to phase offsets at multiples of the smallest difference between two modulation angles in the signal constellation. The latter approach is the subject of this section.

To provide 180^o invariance is relatively easy with linear codes. However, 90^o invariance requires more complexity. We investigate two methods of designing 90^o rotationally invariant PTCM, which retain optimal branch metric calculations. In the following, we outline the basic problems in constructing rotationally invariant PTCM, and offer two possible approaches. The basic conditions to satisfy to make RI (Rotationally Invariant) TCM are :

Condition 3 (conditions for making RI code) [6] :

1. If n is the number of input bits to be differentially encoded, then the tolerable phase

rotation is any multiple of $2\pi/2^n$ radians.

2. *The rule of equal rotation* : If the differentially encoded n-tuple input to the convolutional encoder is rotated by $2j\pi/2^n$ radians, the corresponding output is a $2j\pi/2^n$ radians rotated version of the original signal. This is illustrated in Figure 4.1 [20]. An input rotation of θ results in an output rotation of θ, where $f(.)$ is a function which maps inputs to corresponding outputs of convolutional codes. r_θ and R_θ are θ degree rotation operators for the input and output signals, respectively. $A^i = (a_1^i, ..., a_n^i)$ is the n bit input for the convolutional encoders and C^i is output signal, where $a_1^i, ..., a_n^i \in \{0, 1\}$ and the superscript i is time index. From Figure 4.1, the following equation must be

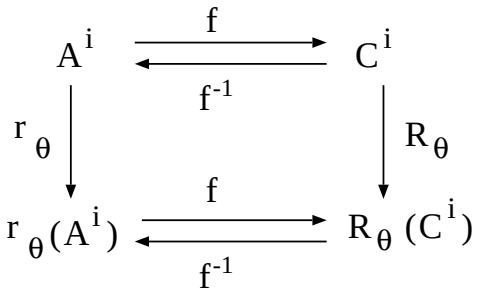


Figure 4.1: Constraint for θ degree rotational invariance.

satisfied for rotational invariance.

$$f \circ r_\theta(.) \; = \; R_\theta \circ f(.) \tag{4.1}$$

where θ can take on only the values $2j\pi/2^n$ radians.

3. All the signal elements in each set obtained by set partitioning should be associated with the convolutional encoder output bits.

4. Any set of signal elements that have the same radius but are $2j\pi/2^n$ radians apart have to be assigned the same combinations of encoder output bits that are not used in the coset labeling.

Table 4.1: Effects of phase rotation on differentially coded vector at the input of the convolutional encoder.

		Phase Rotation(deg.)			
a_2^n	a_1^n	0	90	180	270
0	0	00	01	10	11
0	1	01	10	11	00
1	0	10	11	00	01
1	1	11	00	01	10

Two possible approaches for having rotationally invariant codes are discussed in the following sections. The first scheme is based on the previously described PTCM structure and the second scheme is based on using two convolutional coders.

4.1 RI-PTCM based on one coder

Having a RI-PTCM code satisfying both *condition 3* and the earlier *constraints* without any performance loss or complexity increase for any number of states is quite difficult. First, we illustrate the problems with 8 state rate-2/3 codes. There are two input bits I_1^n and I_2^n going into the convolutional encoder, where the superscript n is the time index. From the first item of *condition 3*, we can see that a 90^o RI code is possible because there are two differentially encoded input bits. One input bit is for each step, i.e. I_1^n is input for the first step branch and I_2^n is input for the second step branch.

Now consider rotation of code states. We have an 8-state code convolutional encoder structure described in Figure 4.2. As we can see from Figure 4.2, input rotation changes the state, which changes the output. Equation (4.1) describes an important condition for the code to be a rotationally invariant. When the input is rotated by θ, the corresponding

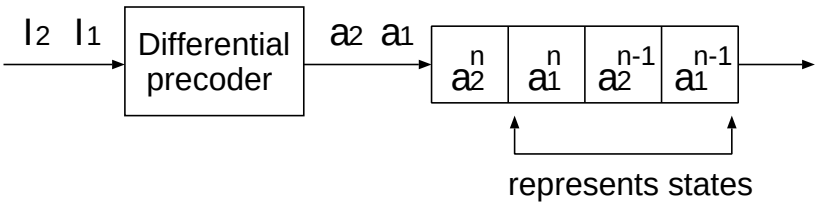


Figure 4.2: Convolutional encoder structure of RI-PTCM.

Table 4.2: Effects of phase rotation on the state vector.

			Phase Rotation(deg.)			
s_3^n	s_2^n	s_1^n	0	90	180	270
0	0	0	000(0)	101(5)	010(2)	111(7)
0	0	1	001(1)	110(6)	011(3)	100(4)
0	1	0	010(2)	111(7)	000(0)	101(5)
0	1	1	011(3)	100(4)	001(1)	110(6)
1	0	0	100(4)	001(1)	110(6)	011(3)
1	0	1	101(5)	010(2)	111(7)	000(0)
1	1	0	110(6)	011(3)	100(4)	001(1)
1	1	1	111(7)	000(0)	101(5)	010(2)

output must be θ rotated version of the original output. Input rotation is directly related with state rotation as we can see as follows.

We can derive the phase rotation of code states from the phase rotation of 2 bit inputs in Table 4.1. The state level is $(a_1^n a_2^{n-1} a_1^{n-1})$ in Figure 4.2. From Table 4.1, a_1 is complemented under $\frac{\pi}{2}$ and $\frac{3\pi}{2}$ rotation. The value of a_2 under rotation depends on the value of a_1. For example, consider a 90^o rotated version of state 000. Since the 90^o rotated input pair($a_2 a_1$) 00 is 01, $a_2^{n-1} a_1^{n-1}$ is 01. a_1^n is 1 because a_1 is flipped under 90^o rotation. Thus state 101 is a 90^o rotated version of state 000. The resulting phase rotation relations on the state vector are in Table 4.2. Now we want to check whether our PTCM structure fits into this RI code framework. The merged rate-2/3 trellis from the two step punctured trellis is in Figure 4.3. The signal constellations and rotation

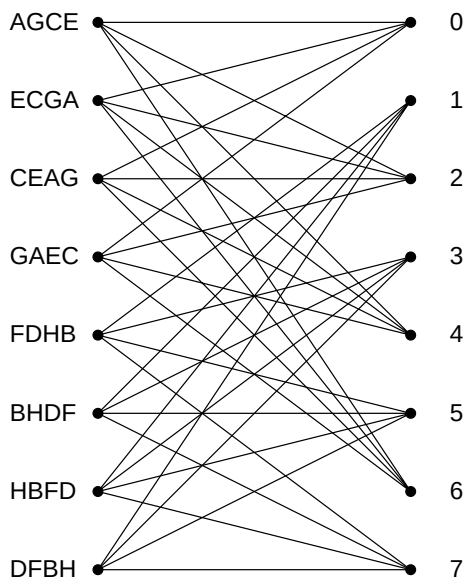
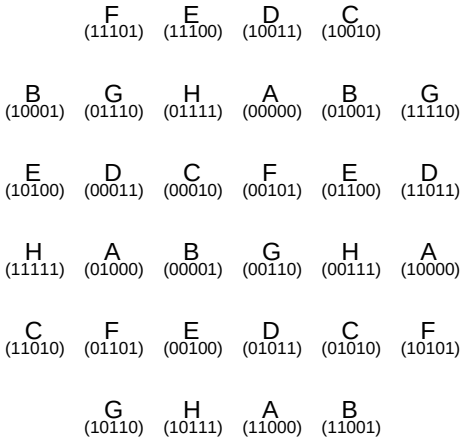


Figure 4.3: Merged rate-2/3 trellis.

relations among cosets are in Figure 4.4, where the first two bits represents uncoded bits and the remaining three bits represents cosets.

For the previously proposed rate-2/3 PTCM to be rotationally invariant, the state rotation (or the input rotation) and corresponding output rotation must be matched. From Figure 4.3, the outputs from state 0 to states 0,2,4 and 6 are A, G, C and E, respectively. From Table 4.2 and Figure 4.4, the 90^o rotated versions of the above state transition is from state 5 to states 5,7,1,3 and the corresponding outputs are D, F, B, H, respectively. Since D, F, B and H are 90^o rotated version of A, G, C and E, this case keeps 90^o rotational invariance. After considering all the possible cases in the same way, we have found that there are two cases of mismatch in the trellis of Figure 4.3. The outputs of the state transition 1 to 0,2,4,6 are E, C, G, A. The 90^o rotated version of the above state transition is from 6 to 5,7,1,3 and the corresponding outputs are H, B, F, D which is not a 90^o rotated version of the output E, C, G, A. Thus the output label of state transition 6



Rotation Relations

0 deg. :	A C E G B D F H
90 deg. :	D B H F G E C A
180 deg. :	E G A C F H B D
270 deg. :	H F D B C A G E

Figure 4.4: 32 QAM constellation in RI PTCM and rotation relations among 8 cosets.

to 5,7,1,3 must be F, D, H, B instead of H, B, F, D to keep the 90^o rotational invariance. For the same reason, the output of the state transition 4 to 5,7,1,3 must be H, B, F, D instead of F, D, H, B.

According to the above results, we change the state 4 and 6 at the left side of the two step trellis. The procedure is illustrated in Figure 4.5, and corresponding rate-1/2 two step *heterogeneous* trellis is in Figure 4.6. We have also designed rate-1/2 RI-TCM and rate-3/4 RI-PTCM. Both codes need 4 dimensional signalling and each 2-D signal has 4 cosets.

4.1.1 Rate-1/2 rotationally invariant TCM

Since there is only one input bits to the convolutional encoder, there is only one differential encoded bit and we do not have a 90^o RI code. To solve this, we use 4-dimensional signaling, where two consecutive input bits are differentially encoded and put into the convolutional encoder. As in the case of rate-2/3 PTCM, two steps are considered as one unit of RI transition, and so the convolutional encoder state structure

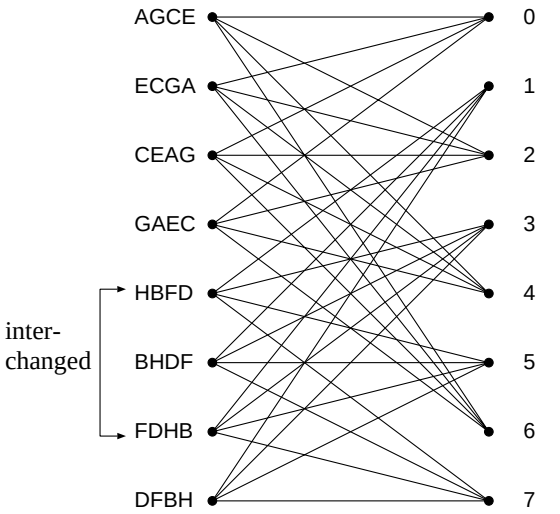


Figure 4.5: New rate-2/3 trellis for RI PTCM.

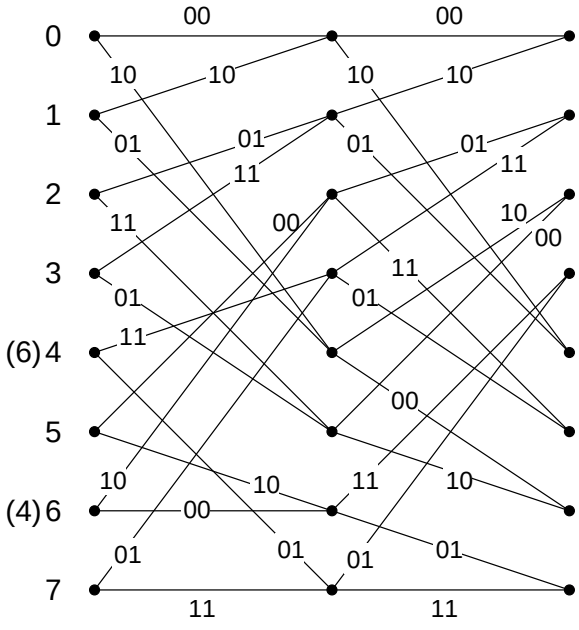


Figure 4.6: Two step heterogeneous rate-1/2 trellis.

Table 4.3: Effects of phase rotation on four cosets.

	Phase Rotation(deg.)			
cosets	0	90	180	270
A	00	01	11	10
B	01	11	10	00
C	10	00	01	11
D	11	10	00	01

related with differential encoder is the same as that of rate-2/3 RI-PTCM. The effect of phase rotation on the state vector is the same as that of rate-2/3 as presented in Table 4.2.

When we consider the two step trellis as one unit, there are 16 possible output labels in the trellis (one 2-D signal has 4 cosets, and the possible number of 4-D output is $4 \times 4 = 16$). The coset partitioning in 32-QAM is illustrated in Figure 3.2. The binary representation and rotational relations of the 4 cosets are in Table 4.3. For the rate-1/2 PTCM having RI structure, the trellis has some constraints on its output labels. This is explained as follows, with reference to the two step trellis. Let the output label assigned from state 0 to 0 be the reference. Consider the constraints from the 90^o RI relations, which are illustrated in Figure 4.7. In Figure 4.7, we label the 16 branches at each step. There are two labels in each branch. The first value is the label of the branch, and the second value in the parenthesis is the label of the branch which must have 90^o rotated output.

For example, the transition from state 0-0-0 has outputs $A - A(0000)$. The 90^o rotated version of the state 0 is 5. The output corresponding to the transition from state 5-2-5 must have output $B - B(0101)$ which is a 90^o rotated version of the output $A - A$. The labels of the branches between state 5 and 2 in the first step and between state 3

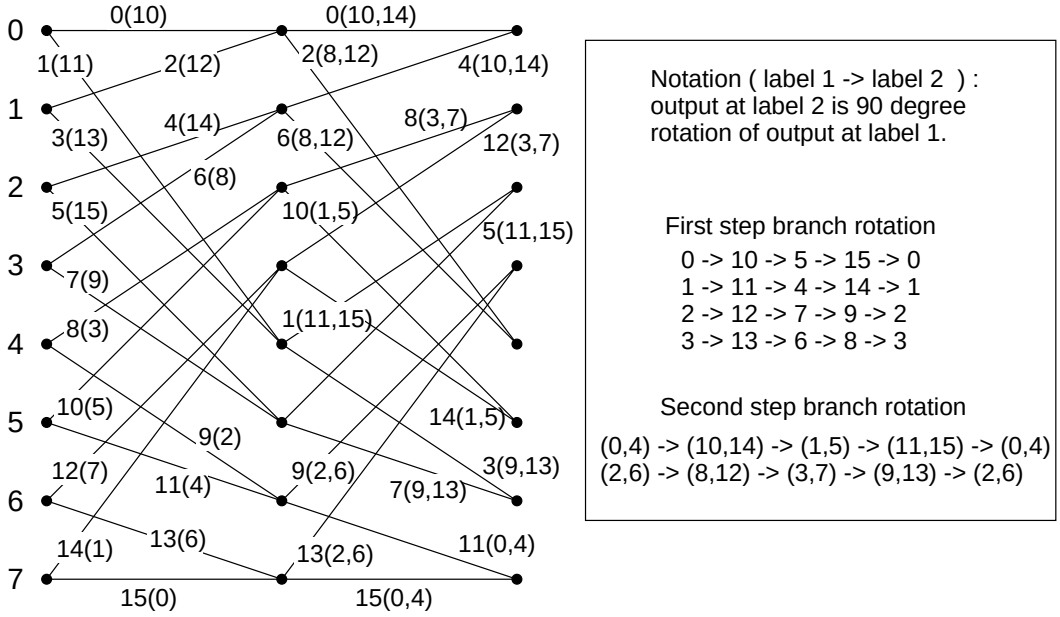


Figure 4.7: Output label constraints from 90^o rotation relations of rate-1/2 TCM.

and 5 in the second step are 10 and 10, respectively. The value in the parenthesis of the branch 0 in the first step and 0 in the second step are 10 and 10, respectively. The meaning of this is that the output of the branch 10 is a 90^o rotated version of the output of the branch 0. Consider the two step transition from state 1 to 0 (1-0-0: corresponding branch label is 2 in the first step and 0 in the second step) – 90^o rotated state is 6 to 5 (6-3-5: corresponding branch label is 12 in the first step and 14 in the second step). From this, we know that, in the second step, the branch 14 is also a 90^o rotated version of the output of the branch 0. That is why we have two labels in the parenthesis in the second step. Furthermore, the second step branch 10 and 14 must have the same output label.

By the same procedure, we can obtain all the output constraints in the trellis for 90^o RI codes. We can use the same methods for the 180^o and 270^o RI constraints. The output label constraints from 180^o rotational relations are illustrated in Figure 4.8.

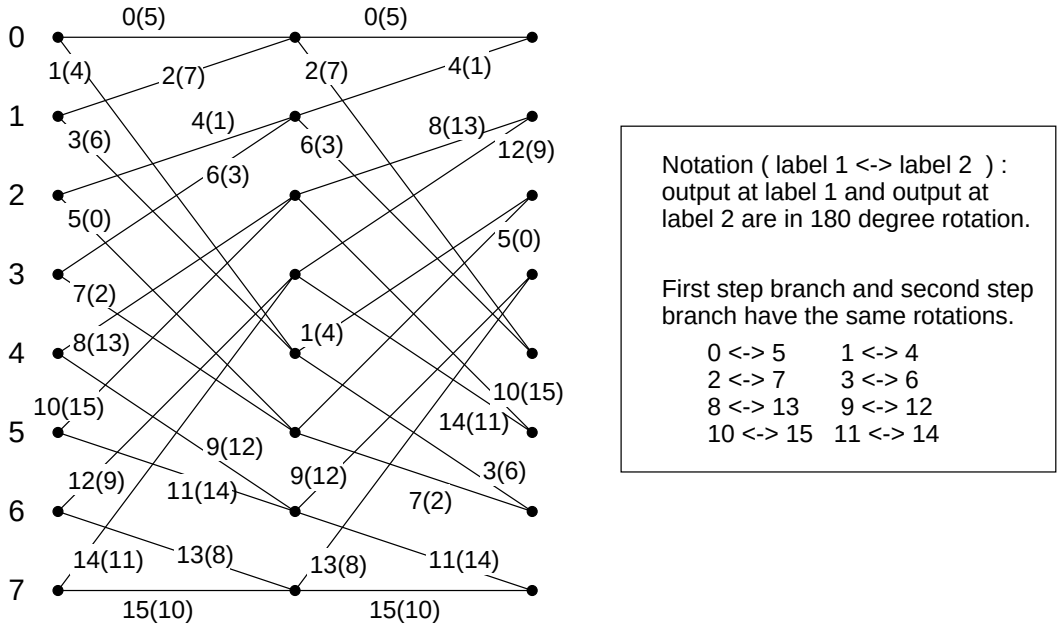


Figure 4.8: Output label constraints from 180^o rotation relations of rate-1/2 TCM.

As we can see in the Figure 4.7 and Figure 4.8, the same outputs are merging into the same state at the second step, which reduces the coding gain. By means of a computer search, the best code satisfying the above constraints shows d_f^2 of 4 and path multiplicity of 16 (The optimum 8 state rate-1/2 TCM has d_f of 6 and path multiplicity of 64).

4.1.2 Rate-3/4 rotational invariant PTCM

As for 90^o rotationally invariant rate-1/2 PTCM, rate-3/4 PTCM uses 4 dimensional signaling. We use the same signal constellation shown in Figure 3.2. Three consecutive input bits are going into the convolutional encoder. Three steps are considered as one unit of rotational invariant transition. Since we need two differentially encoded input bits, the second and third input bits are differentially encoded. The convolutional encoder structure is illustrated in Figure 4.9. The resulting phase rotation relations on

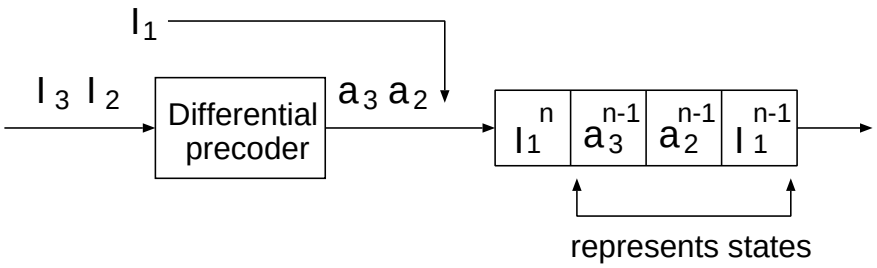


Figure 4.9: Convolutional encoder structure of rate-3/4 PTCM.

Table 4.4: Effects of phase rotation on the state vector (rate-3/4 PTCM).

			Phase Rotation(deg.)			
s_3^n	s_2^n	s_1^n	0	90	180	270
0	0	0	000(0)	010(2)	100(4)	110(6)
0	0	1	001(1)	011(3)	101(5)	111(7)
0	1	0	010(2)	100(4)	110(6)	000(0)
0	1	1	011(3)	101(5)	111(7)	001(1)
1	0	0	100(4)	110(6)	000(0)	010(2)
1	0	1	101(5)	111(7)	001(1)	011(3)
1	1	0	110(6)	000(0)	010(2)	100(4)
1	1	1	111(7)	001(1)	011(3)	101(5)

the state vector are in Table 4.4. To obtain the optimal trellis output label, we could apply the same method we have used for rate-1/2 RI-PTCM. However the rate-3/4 trellis has some differences from the rate-1/2 trellis. We assign only one bit as an output for each of the second and third step branches. To obtain the rotational invariance, we have to consider the second and third steps at the same time. That difference induces the following design constraint on rate-3/4 RI-PTCM.

constraint 3 The output bit label of the third step branch stemming from the same state must have the same value.

The labeling of branches in the three step trellis is shown in Figure 4.10. Consider the state transition 0-0-0-0 and 0-0-0-4 (corresponding second and third step branch labels are 0-0 and 0-4, respectively). The 90^o rotated version of this state transition is

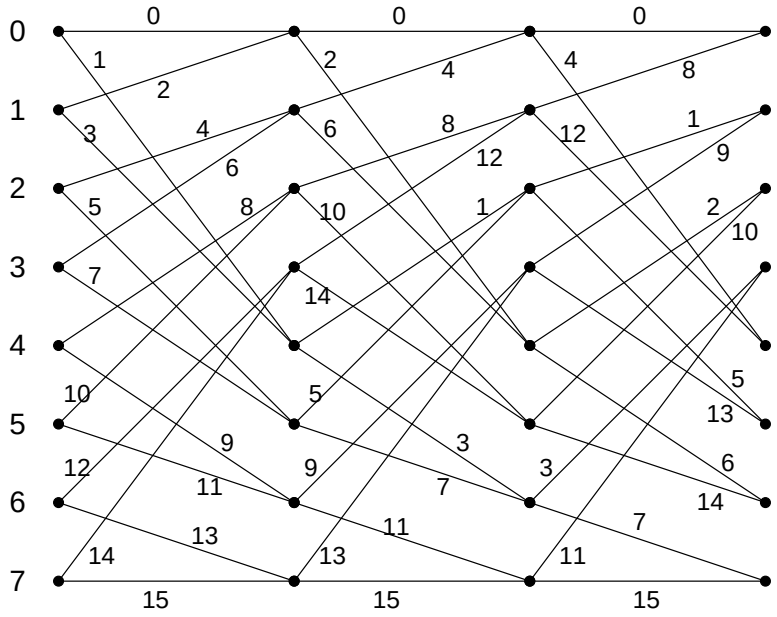


Figure 4.10: Labeling of three step trellis structure of rate-3/4 PTCM.

2-1-4-2 and 2-1-4-6 (with the corresponding second and third step branch labels being 6-2 and 6-6, respectively). Thus, the output of the second step and third step branches labeled by 6-2 and 6-6 must be 90^o rotated versions of the second and third step branches 0-0 and 0-4. Assume now that the output label of the third step branch 0 and 4 are different, for example, the branch 0-0 and 0-4 have outputs 00 and 01, respectively. Since a 90^o rotated version of these are 01 and 11, the branches 6-2 and 6-4 must have outputs 01 and 11. However, the branch 6 at the second step can not be labeled by both 0 and 1.

By the *constraint 3*, we can merge the second and third step trellises and label them as illustrated in Figure 4.11 and Figure 4.12. Then we can use the same method we have

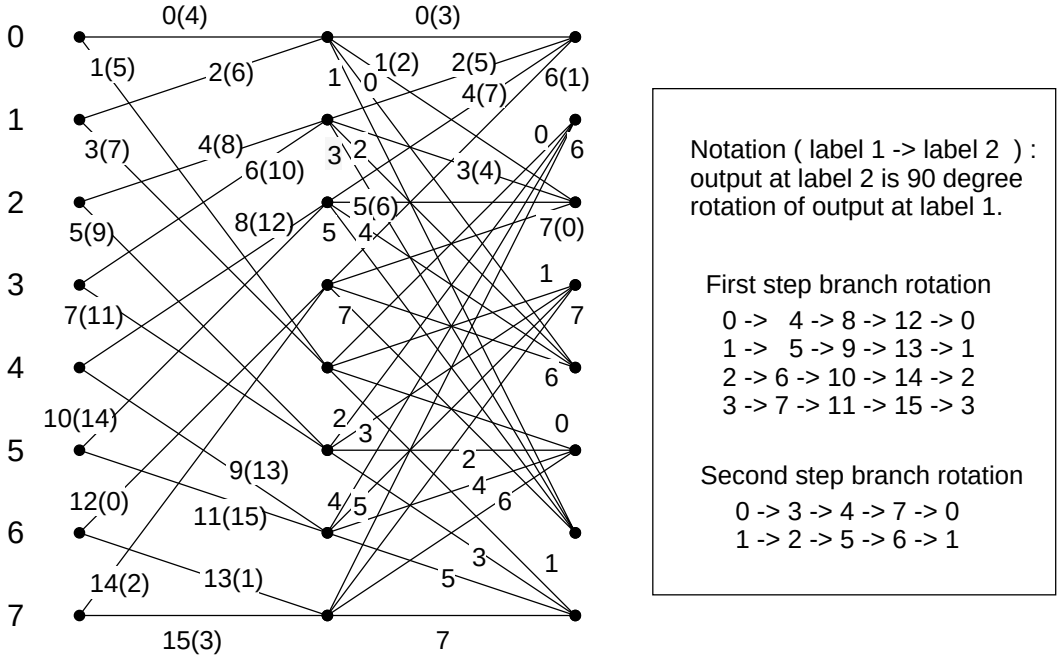


Figure 4.11: Output label constraints from 90^o rotation relations of rate-3/4 PTCM.

used in rate-1/2 RI-PTCM. In Figure 4.11 and Figure 4.12, The value in the parenthesis is the label of the branch which must have 90 and 180^o rotated output of the corresponding output, respectively. The optimum trellis found in a computer search under *constraint 3* is shown in Figure 4.13. The resulting d_f^2 is 3 and the path multiplicity is 48. In the optimal 8 state PTCM, d_f^2 is 4 and path multiplicity is 224. From Figure 4.13, the output label of the state transition 0-0-0-0 and 0-0-0-4 have the same value. This is due to the *constraint 3* and the reason for the performance loss in RI-PTCM. By increasing the number of states (i.e. more than 8 states), we can obtain the same asymptotic coding gain as the optimal rate-3/4 PTCM.

We have found the best RI code under *condition 4*. As a result, swapping the trellis is not necessary and we change the output labels of the PTCM trellis. However, d_f, the minimum distance among different paths in the trellis, is worse than the non-RI code

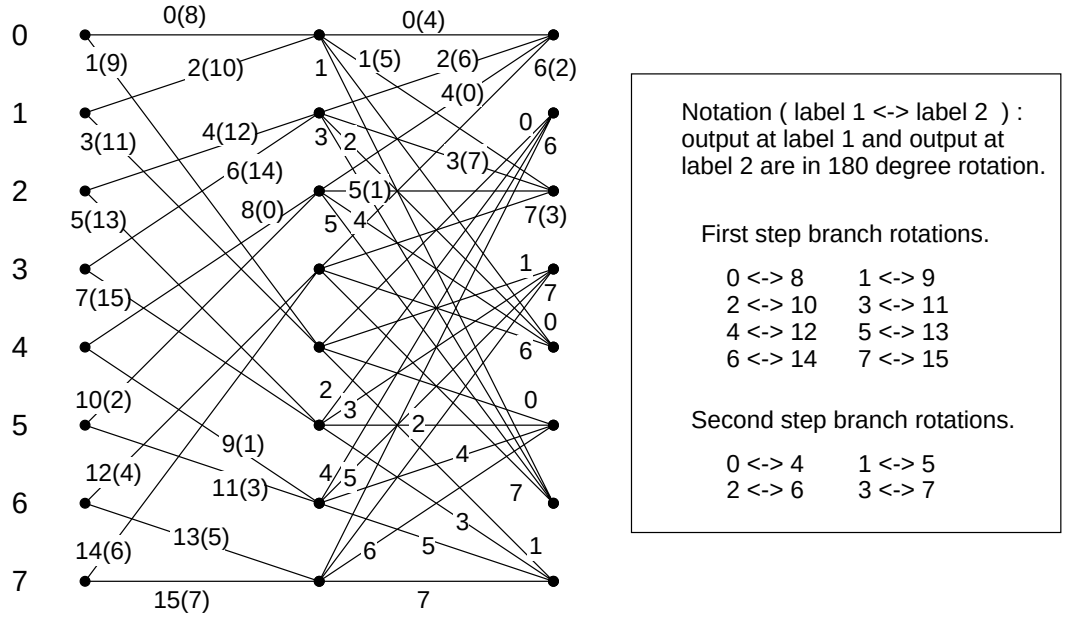


Figure 4.12: Output label constraints from 180^o rotation relations of rate-3/4 PTCM.

with the same number of states. Considering the squared intra-coset distance is 4, we need at least 4 states in the optimal rate-1/2 TCM because d_f^2 is 5 in 4 state trellis. On the contrary, d_f^2 is 4 in 8 state rate-1/2 RI TCM. For rate-3/4 codes, d_f^2 is 4 in 8 state non-RI PTCM and d_t^2 is 4 in 16 state RI-PTCM. Thus we need double the complexity to achieve the same coding gain and 90^o rotational invariance.

4.2 RI-PTCM based on two coders

We have designed PTCM for various code rates. Rate-2/3 PTCM has 8 cosets per two dimensions and has large intra-coset distance. In this case, we classify the 8 cosets into two groups and design a new punctured trellis, i.e. new puncturing positions and generators.

There is another way of designing PTCM which uses existing optimal punctured

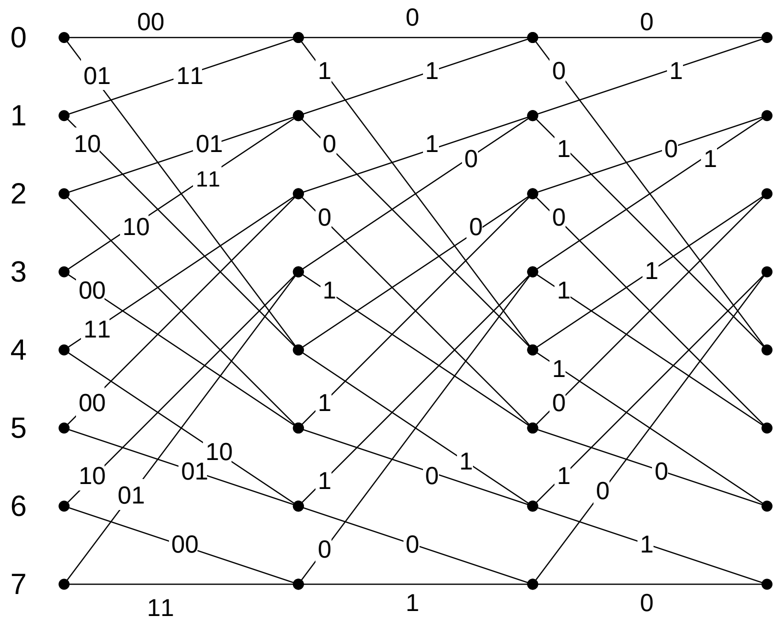


Figure 4.13: Optimal trellis of rate-3/4 RI-PTCM.

convolutional codes and has 4 cosets per two dimensions for all the code rates. We assign two output bits of the punctured convolutional encoder to every two dimensional signal, one bit per dimension. Thus we have 4 cosets per two dimensions. Then the branch metric for each bit can be obtained independently when we use the orthogonal signal constellation in Figure 3.2, which we have used already in rate-3/4 PTCM. In this case, we can use the existing optimal punctured convolutional code because the Hamming distance of the labels is proportional to squared Euclidean distance, and each bit independently contributes one component of the squared distance. As a result, we can design any rate of PTCM without changing the signal constellation structure. When the code rate is m/k, we need k dimensional signalling per m step trellis.

This scheme has worse performance than the previous one for rate-2/3 PTCM because this scheme has smaller intra-coset distance due to the smaller number of cosets.

However, we can construct various rates of PTCM without changing the metric structure. This scheme will be used for the 90^o rotationally invariant code structure described in this section.

The 90^o RI-PTCM designed in the previous section is a non-linear code which uses ordinary punctured coding and then swaps state labels according to the *condition 3*, but we face the situation that the trellis is not exactly the same at each step, and the labels to swap depend on the rate and the number of states. Thus, designing a compatible family of codes appears problematic. Furthermore, the minimum distances of these codes are worse than the minimum distances of the best non-RI code at rate-1/2 and 3/4.

There is another way of implementing 90^o RI-PTCM without swapping the trellis or decreasing the minimum distance. This scheme needs separate PTCM for the in-phase and quadrature-phase components. All we need is for the original rate-1/2 code to be 180^o invariant. We first consider 180^o RI-PTCM with BPSK signalling and extend the concept up to 90^o RI-PTCM with M-QAM signalling.

4.2.1 180^o rotationally invariant TCM

Consider rate-1/2 180^o RI-TCM with BPSK signalling, which is a rate-1/2 180^o RI-convolutional code. Define the time sequence space Φ such that $S \in \Phi$ if and only if

$$S = \{s_0, s_1, s_2, ..., s_n, ...\} \quad , s_n \in \{\phi, 0, 1\}.$$

where the subscript n is the time index. The binary time sequence space Υ is a subspace of Φ which does not have ϕ as its element. "$\overline{S}$" is bit flipped result of S, i.e. it changes the values from 0 to 1, from 1 to 0 and from ϕ to ϕ. $g : \Upsilon \rightarrow \Upsilon^2$ is a rate-1/2 convolutional encoder function, with corresponding decoder function, $g^{-1} : \Upsilon^2 \rightarrow \Upsilon$.

Thus

$$g(A) \;=\; (X1, X2), \;\text{ and }\; g^{-1}(X1, X2) \;=\; A.$$

where $X1, X2$ and A are elements of Υ. g must be a one-to-one function, otherwise it cannot be used as an encoder, and thus g^{-1} exists. Since g is a 180^o RI-code, from equation (4.1)

$$g(r_\pi(A)) \;=\; R_\pi((X1, X2)) \tag{4.2}$$

$r_\pi(A)$ is $\overline{A}$. $R_\pi((X1, X2))$ is $(\overline{X1}, \overline{X2})$ because we use BPSK signalling. Thus the following relation is derived when g is a 180^o RI-code.

$$g(A) = (X1, X2) \iff g(\overline{A}) = (\overline{X1}, \overline{X2}) \tag{4.3}$$

Define the function of the punctured convolutional encoder as $h : \Upsilon \to \Phi^2$ and decoder $h^{-1} : \Phi^2 \to \Upsilon$ such that

$$h(A) \;=\; (X1, X2), \;\text{ and }\; h^{-1}(X1, X2) \;=\; A.$$

$X1$ and $X2$ cannot have ϕ at the same time index. When there is a ϕ, we do not send any signal.

Fact 1 Puncturing does not affect 180^o rotational invariance of a convolutional code.

Proof We have to show that h is 180^o rotationally invariant if g is 180^o rotationally invariant, when h is a punctured version of g.

$$g(A) \;=\; (X, Y), \quad A, X, Y \in \Upsilon$$

Then

$$h(A) \;=\; (U, V), \quad U, V \in \Phi$$

where

$$u_i = \begin{cases} x_i \ , & i \notin P_x \\ \phi \ , & i \in P_x \end{cases}$$

$$v_i = \begin{cases} y_i \ , & i \notin P_y \\ \phi \ , & i \in P_y \end{cases}$$

P_x and P_y are subsets of non-negative integers which indicate the punctured time index of output X and Y, respectively. We have to show that if $g(\overline{A}) = (\overline{X}, \overline{Y})$ then $h(\overline{A}) = (\overline{U}, \overline{V})$.

Let $h(\overline{A}) = (U', V')$, then by definition of h

$$u'_i = \begin{cases} \overline{x_i} \ , & i \notin P_x \\ \phi \ , & i \in P_x \end{cases}$$

$$v'_i = \begin{cases} \overline{y_i} \ , & i \notin P_y \\ \phi \ , & i \in P_y \end{cases}$$

Since $\overline{\phi}$ is ϕ, $U' = \overline{U}$ and $V' = \overline{V}$ $\Box$.

We extend 180^o rotational invariance up to 90^o rotational invariance by using two 180^o RI-codes in the next section.

4.2.2 90^o rotationally invariant PTCM

Calderbank and Mazo proposed a 90^o RI-TCM scheme, where the incoming bit stream is split into two streams and a separate trellis encoder is used for each stream [30]. The output symbols of each encoder separately modulate the in-phase and quadrature-phase carrier components. Then the set of all possible such pairs have 90^o rotational

Table 4.5: Effects of phase rotation on differentially coded vector at the input of the convolutional encoder.

Label	$a_n b_n$	*Phase Rotation(deg.)*			
		0	90	180	270
0	00	00	01	11	10
1	01	01	11	10	00
2	11	11	10	00	01
3	10	10	00	01	11

symmetry. This idea and the PTCM scheme described in the previous section will be used in designing 90^o RI-PTCM.

To achieve 90^o RIC, we needs 2 bits of inputs going into a differential encoder. The input bit rotations and labels are in Table 4.5. Consider 4-QAM signalling with labels as in Table 4.5. The input and output have the same rotation relations (i.e. $R_\theta = r_\theta$). The rotation pattern is expressed by the rotation function $R_\theta : \Phi^2 \rightarrow \Phi^2$ defined as follows.

$$\begin{aligned} R_0(X,Y) &= (X,Y) \\ R_{\frac{\pi}{2}}(X,Y) &= (Y,\overline{X}) \\ R_{\pi}(X,Y) &= (\overline{X},\overline{Y}) \\ R_{\frac{3\pi}{2}}(X,Y) &= (\overline{Y},X) \end{aligned} \tag{4.4}$$

Figure 4.14 illustrates the system. I $\in \Upsilon$ and J $\in \Upsilon$ are information bit sequences. A $\in \Upsilon$ and B $\in \Upsilon$ are the differentially encoded results of I and J.

$$A = f_1(I,J), \quad B = f_2(I,J)$$

where the function $(f_1, f_2) : \Upsilon^2 \rightarrow \Upsilon^2$ is a one to one function of differential encoders and its inverse function is the differential decoder function $(F_1, F_2) : \Upsilon^2 \rightarrow \Upsilon^2$ defined

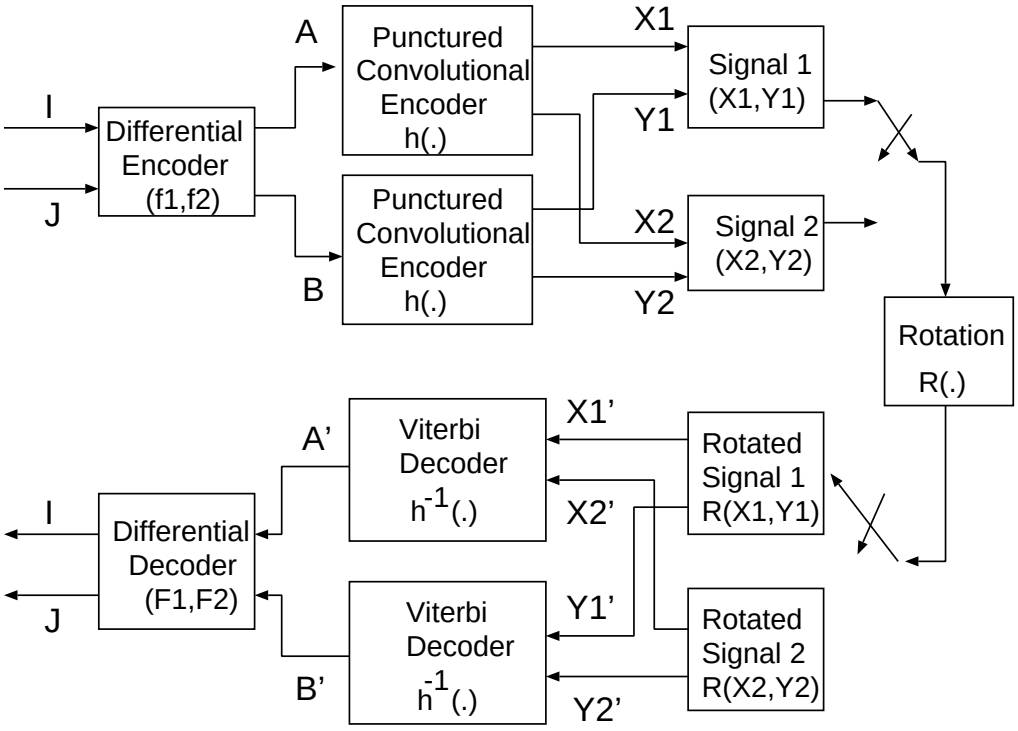


Figure 4.14: 90^o rotationally invariant punctured coding system.

as

$$I = F_1(A,B), \quad J = F_2(A,B)$$

Signal 1 and signal 2 are transmitted alternately, and each has 4 points which can be expressed by two dimensional binary time sequences $(X1, Y1)$ and $(X2, Y2)$, respectively.

Since there are two identical punctured convolutional encoders, we define the punctured convolutional encoder function as $(h,h) : (\Upsilon, \Upsilon) \rightarrow (\Phi^2, \Phi^2)$ such that

$$(h,h)(A,B) = ((X1, Y1), (X2, Y2)) \tag{4.5}$$

where $h(A) = (X1, X2)$ and $h(B) = (Y1, Y2)$. The inverse function $(h,h)^{-1}$ exists and is (h^{-1}, h^{-1}).

Fact 2 the system described in Figure 4.14 is 90^o rotationally invariant.

Proof From equation (4.1), we have to show that

$$\begin{aligned}(h,h)\circ r_\theta(A,B) &= R_\theta\circ(h,h)(A,B)\\ &= R_\theta((X1,Y1),(X2,Y2)) \end{aligned} \tag{4.6}$$

r_θ and R_θ are the same because the input and output have the same rotation relations. Since rotation of two consecutive signals is the same as two consecutive rotated signals, we have

$$R_\theta((X1,Y1),(X2,Y2)) \;=\; (R_\theta(X1,Y1),R_\theta(X2,Y2)) \tag{4.7}$$

Thus we need to show that

$$(h,h)\circ R_\theta(A,B) \;=\; (R_\theta(X1,Y1),R_\theta(X2,Y2)) \tag{4.8}$$

There are two kinds of operations in R_θ, i.e. exchange of two positions and bit flipping. From equation (4.5), exchange of A and B induces exchange of X and Y. By the 180^o rotational invariance of h, a bit flip of A or B induces bit flip of X or Y, respectively. Thus the relation of equation (4.8) is true. □

We have used 4-QAM signalling and showed it is 90^o RIC. We can extend the size of the signal constellation. We illustrate 16-QAM in Figure 4.15, showing that the 4 kinds of cosets keep their rotational relations. The 16-QAM constellation can be decomposed into two 4-PAM constellations as described in Figure 4.15. Every first bit of signal 1 and signal 2 is from one convolutional encoder and every second bit is from the other encoder. Thus the output label in the trellis is expressed by two bits which are from different signals. The branch metric is obtained from the two consecutive signals, one

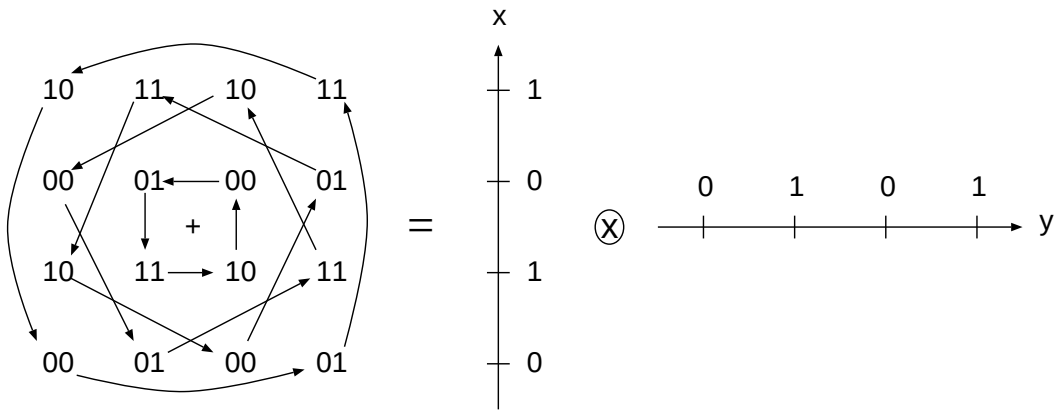


Figure 4.15: 16-QAM signal constellation for rotationally invariant code.

for each dimension. Since there is only one bit per dimension and the distance between 1 and 0 is 1 as we can see in Figure 4.15, the squared Euclidean distance is the scaled Hamming distance. So, we can use the same kind of branch metric calculation in both the punctured and non-punctured trellis.

The intra-coset distance d_c is the minimum distance among the points expressed by the same binary labels at the trellis. Considering two bits labels in each branch are independent and the distance between the same binary labels in each dimension is 2, d_c^2 is 4 in both punctured and non-punctured trellis.

Consider the minimum distance of the code trellis d_f. Since the squared branch metric is the same as Hamming distance, the convolutional code performance is the same as that of a 180^o rotationally invariant binary convolutional code.

In this scheme, all we need is for the original rate-1/2 convolutional code to be 180^o invariant. Then if a two dimensional differential encoder is used, all punctured versions of the code will be invariant. Thus we may continue to use the the same decoder structure for all members of the family.

Good 180^o rotationally invariant rate-1/2 convolutional codes that lead to suitable

Table 4.6: Comparisons of 180^o RI and best known rate-1/2 convolutional codes.

states	best known code [19]			180^o RI-code		
	Generators(octal)	d_f^2	N_f	Generators(octal)	d_f^2	N_f
4	5,7	5	1	4,7	4	2
8	15,17	6	1	13,15	6	2
16	23,35	7	2	23,31	6	1
32	65,57	8	1	51,67	8	2
64	133,171	10	11	133,171	10	11

punctured higher rates and best known rate-1/2 codes are compared in Table 4.6. When we use 4-QAM constellation, the performance bottleneck is not the intra-coset distance. Therefore we can improve code performance by increasing the number of states.

4.3 Comparison of two RI-PTCM schemes

We have considered two kinds of 90^o RI-PTCM. Denote the scheme which uses a non-linear coder by RIC-I and two convolutional coder scheme by RIC-II.

The brief characteristics of scheme RIC-II are as follows. The squared intra-coset distance for all code rates is 4, which confines the ACG (asymptotic coding gain) to be less than or equal to 3 dB for rate-1/2 codes, or 6 dB as the rate goes to 1, considering the effect of reduced code redundancy. We can use the 180^o RI linear convolutional codes in Table 4.6. We may use a family of punctured codes of different rates, all with the same basic decoder. However, two convolutional coders are necessary, doubling decoding delay.

Consider the performance and complexity of both schemes for rates-1/2, 2/3 and 3/4. Both rate-1/2 schemes need 4 dimensional signalling and have squared intra-coset distance of 4. For d_f^2 being no less than 4, we need at least 8 states in scheme RIC-I, but

only 4 states are needed for scheme RIC-II. The coding gain is limited to 3 dB.

Rate-2/3 RI-PTCM shows different characteristics. Scheme RIC-I needs two dimensional signaling and has 8 cosets which results in squared intra-coset distance of 8. Scheme RIC-II uses 6 dimensional signaling and needs 8 states for d_f^2 being no less than 4. For the 8 state codes, d_f^2 of scheme RIC-I is 5 and d_f^2 of scheme RIC-II is 4. However we have to swap the trellises in scheme RIC-I.

In rate-3/4 RI-PTCM, both schemes have the same squared intra-coset distance of 4. Scheme RIC-I needs a 16 state trellis for having d_f^2 no less than 4, while scheme RIC-II needs two 8 state trellises, which are decoded at half the speed.

Consider the throughput of rate-$n/(n+1)$ RI-PTCM in scheme RIC-II where $2(n+1)$ dimensional signalling is used. Assume we use 2^s-QAM. We need $s-2$ uncoded bits per 2-D constellation. The number of information bits per $2(n+1)$-D is $2n+(n+1)(s-2) = (n+1)s-2$ and the number of output bits per $2(n+1)$-D is $2(n+1)+(n+1)(s-2) = (n+1)s$. The code rate per $2(n+1)$-D is $\frac{s(n+1)-2}{s(n+1)}$. The code rate of scheme RIC-I is $\frac{s-1}{s}$ per 2-D in rate-1/2 and 2/3, and $\frac{2s-1}{2s}$ per 4-D in rate-3/4. Both schemes have the same throughput for rate-1/2 and 3/4, and scheme RIC-II shows better throughput for rate-2/3. Thus there is no redundancy penalty in scheme RIC-II even though it is using two convolutional coders.

Even though we use 2 convolutional coders in scheme RIC-II, both schemes have almost the same number of trellis steps per 2-D signal. Thus the number of computations for scheme RIC-II is less because it needs half the number of states of scheme RIC-I.

In summary, scheme RIC-I does not need two convolutional coders but needs more states compared with scheme RIC-II. The rate-2/3 PTCM of scheme RIC-I shows better performance but needs swapping of the trellis.

4.4 Discussion

We provided two types of 90^o RI-PTCM for QAM signalling. 90^o invariance requires non-linear codes when the cosets are two-dimensional. This may be accomplished by using ordinary punctured coding and then swapping state labels, or rearranging the branch labels which decreases the minimum distance. On the other hand, we may use separate PTCM for the in-phase and quadrature components, where the original rate-1/2 code must be 180^o invariant and all punctured versions of the code will be invariant. Thus we may continue to use the the same decoder structure for all members of the family.

@

Chapter 5

Unequal Error Protection Codes

In broadcast systems, the channels for each customer are different. With analog transmission, the result is a variable quality of reception, which depends roughly on the range from the transmitter. In digital systems by contrast, the design usually guarantees a uniformly good quality of service up to a given range, after which point the service rapidly deteriorates. For example, in digital HDTV, forward error correction coding will be used to increase the range of high quality service. However, it is possible to perform coding such that customers may receive a lower resolution signal even when the high definition part of the signal is lost, so that a quality of service comparable for example to NTSC can be obtained out past the range of ordinary analog transmission. Unequal error protection (UEP) codes give more protection to the low-rate important information, and thus are suitable for similar broadcast systems.

Cover [50] has considered the maximum achievable capacity over broadcast channels that may have differing capacities. Two forms of transmission (time sharing and superimposing) were considered, with superimposition shown to be better than time sharing in terms of channel capacity.

Time sharing allocates time slots to different rates of transmission. In time sharing for UEP, two or more coded modulation schemes provide different levels of error protection for different class data in different time slots. Generalized time sharing where a code of non-zero rate specifies the multiplexing rule has been proposed [3]. Additional important bits can be transmitted in this scheme.

In the superimposition technique, information for all the classes is sent at the same time. The low-rate data is recovered when the channel is not good enough to recover all the information. Uncoded non-uniform constellations where there are several clusters (or groups) of signal points and the inter-group distance is larger than the intra-group distance have been considered for a simple superimposing scheme [44]. Wei has proposed a UEP code which has 90^o rotational invariance [5] [36]. In his scheme, the rotationally invariant (RI) code is used only for important data because the phase rotation does not change the non-important data decoding results. However the non-important data coding gain is not satisfactory. Calderbank and Seshadri have proposed various kinds of UEP codes for different ratios of important to less important data by using multilevel codes which can allocate the available redundancy in a rather flexible manner [3] [20] [34]. They have used a non-standard set partitioning as well as non-uniform signal constellations to reduce the path multiplicity. The coding gain reduction of non-important data due to non-standard set partitioning can be compensated by using other coders.

We consider extensions of this work on superimposition of class 1 (important bits) and class 2 (less important bits) data. Ordinary TCM on QAM has an innate capability in this regard. Consider rate-1/2 TCM as an example. The squared minimum distance of the 16 state rate-1/2 convolutional code is 7 and the squared minimum distance of the uncoded bits is 4. We can easily achieve two level UEP by using the convolutional coder for class 1 data. However the standard set partitioning causes a large path multiplicity

which reduces the class 1 coding gains significantly.

TCM based on PAM signalling results in convolutionally coded bits with larger d^2_{min} and much lower path multiplicity. However, this 1-D signalling scheme is not power efficient. We propose to compensate for this inefficiency by locating the signal points for less important data on another 1-D axis. In this case, a standard code search does not give us reliable information on d^2_{min} and corresponding path multiplicities because the code is not geometrically uniform. New code search methods will be introduced for better estimation of the actual coding gain. All the schemes we are considering have the same basic structure as mentioned above.

Our schemes can have 90^o rotational invariance by using a 180^o RI rate-1/2 convolutional code for class 1 data and then resolving the in-phase and quadrature-phase power. Like the Wei UEP code scheme, the coders for class 2 data do not have to be RI coders.

The chapter is organized as follows. In section 1, we discuss coding gain calculations. The UEP code family based on set partitioning on one dimensional lattices will be discussed in sections 2, 3 and 4. Methods to design 90^o rotationally invariant codes are discussed in section 5. Some concluding remarks are presented in section 6.

5.1 Coding gain calculations

Coding gain is a function of various factors including redundancy, constellation expansion, constellation power penalty, minimum distance and path multiplicity. The path multiplicity N_f is the number of nearest neighbors and the normalized path multiplicity N^*_f is N_f per 2 dimensions. $N^*_f = N_f/N$ when we use 2N dimensional signalling. Denote by E_N the penalty due to large path multiplicity. If N^*_f is less than 4 we have gains due to small path multiplicity, because N_f for uncoded QAM is 4. Using the

rule of thumb that doubling N_f causes a 0.2 dB degradation at error probability 10^{-6} [24],

$$E_N = 0.2 \log_2(\frac{N_f^*}{4}). \tag{5.1}$$

Let P be the average power of the proposed scheme and P_I and P_Q be the in-phase and quadrature-phase components of the average power, respectively. Then $P = P_I + P_Q$. Let an uncoded M-QAM of average power P_{ref} be the reference scheme. Then the power penalty E_P can be obtained as

$$E_P = 10 \log \frac{P}{P_{ref}} \tag{5.2}$$

Let R be the bit rate per 2-D signalling, the rate loss ρ is defined as $R - \log_2 M$. Then the penalty due to rate loss E_R is 3.01ρ. The asymptotic coding gain G is obtained in equation (5.3).

$$G = 10 \log d_{min}^2 - E_P - E_R \tag{5.3}$$

Consider the minimum distance property of binary convolutional codes. Let Ψ_B be a set of binary codewords. Since convolutional code are linear, for all $B_i, B_j \in \Psi_B$ there exist $B_k \in \Psi_B$ such that $B_i \oplus B_j = B_k$, where $\oplus$ is bitwise modulo 2 addition. Then,

$$d_H(B_i, B_j) = W_H(B_k) = d_H(B_0, B_k) \tag{5.4}$$

where d_H is the Hamming distance, W_H is the Hamming weight and B_0 is the all-zero codeword. Assume B_k is a reference codeword, d_{Hi} ($i \in Z^+$ and $d_{Hi} < d_{Hj}$, if $i < j$) are the possible Hamming distances between B_k and other codewords, and $N_f(d_{Hi})$ is the number of codewords at distance d_{Hi} from the reference B_k. Ω_{Bk} is defined to be the set of elements $(d_{Hi}, N_f(d_{Hi}))$. From the equation (5.4), we see that $\Omega_{Bk} = \Omega_{B0}$ for all k such that $B_k \in \Psi_B$. Thus the convolutional code has the following property [17].

Property 1 The set of distances of the code sequences generated up to some stage in the tree, from the all zero sequence, is the same as the set of distances of the code sequences with respect to any other code sequences.

From *property 1* we can calculate the minimum distance assuming that the all zero sequence is the input to the encoder. This zero-path reference code search (ZRCS) considerably simplifies the search for codes with large d_{min}.

Calderbank and Sloane [27], Benedetto *et al.,* [23] and Forney [22] investigated the conditions sufficient to ensure that a coset code [24] is distance-invariant, so that d_{min} and path multiplicity of the code do not depend on the transmitted code sequence. Except for the boundary region of the constellation, ordinary TCM based on standard set partitioning of regular lattice satisfies the *property 1*, and we can use ZRCS without much error. If a code do not have that property, the code search is quite complex and we need to find a better way. For that purpose, we also derive the sufficient conditions which allow ZRCS in TCM code search, introducing notation which will be convenient for discussing search procedures when ZRCS is inaccurate.

Let Φ be the set of cosets such that $\Phi = \{\xi_0, \xi_1, ..., \xi_{M-1}\}$, where M is the number of cosets. Let $d^2(\xi_i, \xi_j)$ be the squared Euclidean distance between cosets ξ_i and ξ_j and the *kissing number* $\eta(\xi_i, \xi_j)$ be the number of signal points in coset ξ_j which are at the distance $d^2(\xi_i, \xi_j)$ to coset ξ_i.

Let Ψ_C be a set of codewords in TCM. The codeword $C_i \in \Psi_C$ is a sequence of cosets and is expressed as $C_i = (c_{i1}, c_{i2}, ...)$, where $c_{ij} \in \Phi,\ j \in Z^+$. Ψ_C is a homomorphic set of Ψ_B and there is a one to one mapping $\tau(.)$ between the codewords $C_i \in \Psi_C$ and $B_i \in \Psi_B$, such that $\tau(C_i) = B_i$ and $\tau^{-1}(B_i) = C_i$. Every $m(= \log_2 M)$ bits of B_i are grouped to be a binary label of coset sequences in C_i. The operation $\diamond$ in

the set Ψ_C is defined as follows.

$$C_i \diamond C_j \equiv \tau^{-1}(\tau(C_i) \oplus \tau(C_j))$$

The distance between codewords can be obtained as follows.

$$d^2(C_i, C_j) = \sum_{t=1}^{L} d^2(c_{it}, c_{jt}) \tag{5.5}$$

where L is larger than the largest codeword.

Unlike the binary codewords in convolutional codes, codewords in TCM may have multiple paths when each coset has more than one member. Define $\mu(C_i, C_j)$ to be the number of paths of C_j at the distance $d^2(C_i, C_j)$ to C_i. Then,

$$\mu(C_i, C_j) = \prod_{t=1}^{t=L} \eta(c_{it}, c_{jt}). \tag{5.6}$$

Assume C_k is a reference codeword, and d_i^2 ($i \in Z^+$ and $d_i^2 < d_j^2$, if $i < j$) is the possible squared Euclidean distance between C_k and any other codeword and $N_f(d_i^2)$ is the number of codeword paths at distance d_i^2 from the reference C_k. $N_f(d_i^2)$ can be obtained as follows.

$$N_f(d_i^2) = \sum_{d^2(C_k, C_j) = d_i^2} \mu(C_k, C_j) \tag{5.7}$$

Ω_{Ck} is defined as the set of element $(d_i^2, N_f(d_i^2))$ when $C_k \in \Psi_C$ is a reference codeword. If Ω_{Ck} is Ω_{C0} for all $C_k \in \Psi_C$, we can use ZRCS for the code search. The sufficient conditions for ZRCS in TCM can be derived from the fact that for all $C_i, C_j \in \Psi_C$ there exists $C_k \in \Psi_C$ such that $C_i \diamond C_j = C_0 \diamond C_k$, in other words, for any correct codeword C_i and error codeword C_j pair, there exists a matching pair C_0 and C_k. So, if the metric structures of the pairs (C_i, C_j) and (C_0, C_k) are the same, i.e. $d^2(C_i, C_j) = d^2(C_0, C_k)$ and $\mu(C_i, C_j) = \mu(C_0, C_k)$, we can use ZRCS. The sufficient conditions for ZRCS are described as follows.

For all $\xi_i, \xi_j, \xi_k, \xi_0 \in \Phi$ such that $\xi_i \diamond \xi_j = \xi_k$, where ξ_0 is a coset with all-zero binary notation, if the following conditions are satisfied, we can use ZRCS.

Condition 4 $d^2(\xi_i, \xi_j) = d^2(\xi_0, \xi_k)$.

This is derived from the condition $d^2(C_i, C_j) = d^2(C_0, C_k)$.

$$\begin{aligned} d^2(C_i, C_j) &= d^2(C_0, C_k) \\ \text{from equation (5)} \rightarrow \sum_{t=1}^{L} d^2(c_{it}, c_{jt}) &= \sum_{t=1}^{L} d^2(c_{0t}, c_{kt}) \\ d^2(c_{it}, c_{jt}) &= d^2(c_{0t}, c_{kt}), \quad \text{where } c_{it} \diamond c_{jt} = c_{0t} \diamond c_{kt}. \end{aligned}$$

Condition 5 $\eta(\xi_i, \xi_j) = \eta(\xi_0, \xi_k)$.

This is derived from the condition $\mu(C_i, C_j) = \mu(C_0, C_k)$.

$$\begin{aligned} \mu(C_i, C_j) &= \mu(C_0, C_k) \\ \text{from equation (6)} \rightarrow \prod_{t=1}^{L} \eta(c_{it}, c_{jt}) &= \prod_{t=1}^{L} \eta(c_{0t}, c_{kt}) \\ \eta(c_{it}, c_{jt}) &= \eta(c_{0t}, c_{kt}), \quad \text{where } c_{it} \diamond c_{jt} = c_{0t} \diamond c_{kt}. \end{aligned}$$

If there is more than one member per coset, the values of $d^2(\xi_i, \xi_j)$ and $\eta(\xi_i, \xi_j)$ might depend on a specific choice of members in each coset. For example, the kissing numbers of cosets at the boundary and inside of constellation could be different. In this case, the two conditions are quite difficult to satisfy. However, when we use ordinary TCM with large signal constellations, we can design the code such that most of the signal points satisfy the conditions so that ZRCS can be used without much error. Standard four way set partitioning on uniform 64-QAM constellation is illustrated in Figure 5.1, where

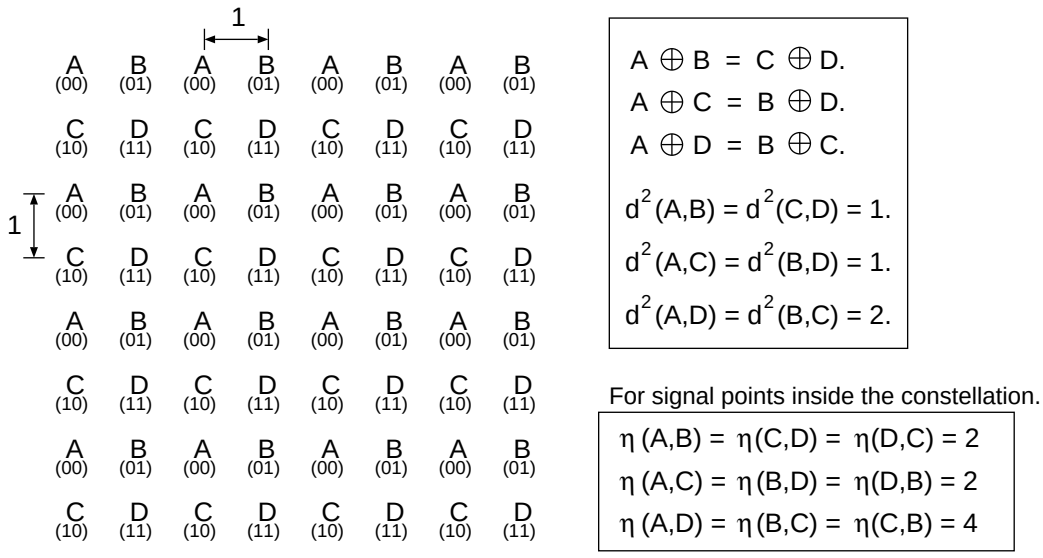


Figure 5.1: Four way partitioning in uniform 64-QAM constellation illustrating the two conditions for ZRCS.

condition 4 is satisfied for all the points but *condition 5* is not satisfied for some points at the boundary of the constellation. However, most of them satisfy the both conditions and we can use ZRCS without much error.

For non-standard partitioning, we may instead use a statistical modification of ZRCS (S-ZRCS), where the all-zero path is still used as a reference but the squared distances and kissing numbers of cosets are varied depending on the transmitted codeword or in a random manner. Suppose that $\xi_i \diamond \xi_j = \xi_k$, $d^2(\xi_i, \xi_j) \neq d^2(\xi_0, \xi_k)$ and $\eta(\xi_i, \xi_j) \neq \eta(\xi_0, \xi_k)$. The metric $M_e(\xi_k)$ and kissing number $K(\xi_k)$ of the branch labeled ξ_k are $d^2(\xi_0, \xi_k)$ and $\eta(\xi_0, \xi_k)$, respectively if ξ_0 is transmitted, or they are $d^2(\xi_i, \xi_j)$ and $\eta(\xi_i, \xi_j)$ if ξ_i is transmitted. In general, $d^2(\xi_i, \xi_j)$ and $\eta(\xi_i, \xi_j)$ may have different values for different signal points in coset ξ_i. Thus, we choose a signal point of cosets in an equiprobable manner and assign the corresponding values to the distance metric

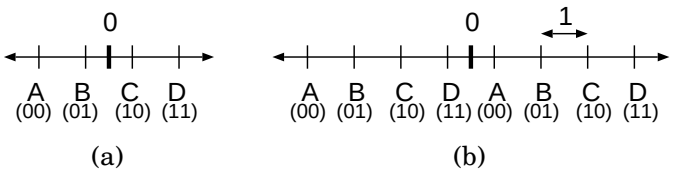


Figure 5.2: Four way partitioning on PAM signal constellations. (a) 4-PAM. (b) 8-PAM.

and kissing number. In the code search, we use many different codewords as a reference and obtain the possible distances and corresponding average path multiplicities. Consider two examples of TCM on PAM signalling. In Figure 5.2 (a), there are 4 cosets A, B, C, D(with binary labels 00, 01, 10, 11, respectively) and each coset has one member, where we do not have to consider the *condition 5*. Since $B \oplus C = D$ but $d^2(B, C) \neq d^2(A, D)$, this does not satisfy *condition 4*. When we use ZRCS, the metric of the branch labeled D, $M_e(D)$, can be $d^2(B, C)$ or $d^2(A, D)$. When the transmitted coset is A or D, $M_e(D)$ is 9. Otherwise, $M_e(D)$ is 1.

In Figure 5.2 (b), each coset has two members. Suppose coset A is transmitted, then $M_e(D)$ and $K(C)$ do not have unique values. If the negative signal is transmitted, $M_e(D)$ is 9 and $K(C)$ is 1, and if the positive signal is transmitted, $M_e(D)$ is 1 and $K(C)$ is 2. By the same way, when coset B is transmitted, then $K(C)(= \eta(B, D))$ is 1 or 2 if the signal is negative or positive, respectively. Since each pair of cosets (A, D) and (B, C) has the same metric structure, we do not have to consider the case when coset C or D is transmitted.

From the previously described code search, we obtain the possible squared Euclidean distances d_i^2 ($i \in Z^+$ and $d_i^2 < d_j^2$ if $i < j$) and corresponding average path multiplicities $N_f^*(d_i^2)$ per 2-D. Then, the component coding gain α_i, i=0,1,2,..., is defined as

$$\alpha_i = 10 \log d_i^2 - 0.2 \log_2 \left(\frac{N_f^*(d_i^2)}{4} \right) \tag{5.8}$$

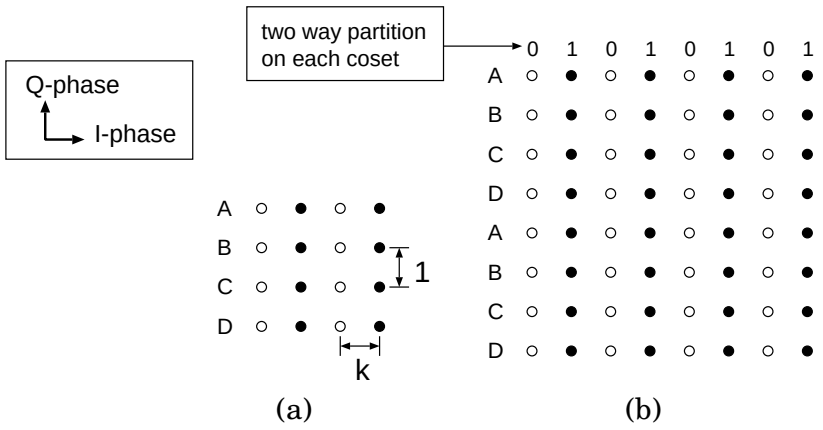


Figure 5.3: Signal constellation for scheme I. (a) 16-QAM. (b) 64-QAM.

A code search finds the generator sequence which maximizes the value $\min \alpha_i$ We define the actual coding gain G_a as

$$G_a = \max \left(\min_{i \in Z^+} \alpha_i \right) - E_P - E_R \tag{5.9}$$

A family of multilevel UEP codes using set partitioning on one dimensional lattice will be considered. The uncoded 16-QAM is considered as a reference scheme in coding gain calculations and simulations.

5.2 Scheme I

We use 16 QAM and 64 QAM in both 2 and 4 dimensional signal constellations. The coset partitioning and signal constellations are shown in Figure 5.3, where we can change the in-phase distance between signal points by changing the value of k.

5.2.1 Two dimensional signaling scheme

Code structures using 16-QAM and 64-QAM are shown in Figure 5.4. There are two schemes (scheme I-A and scheme I-B) for 16-QAM and 64-QAM. The code

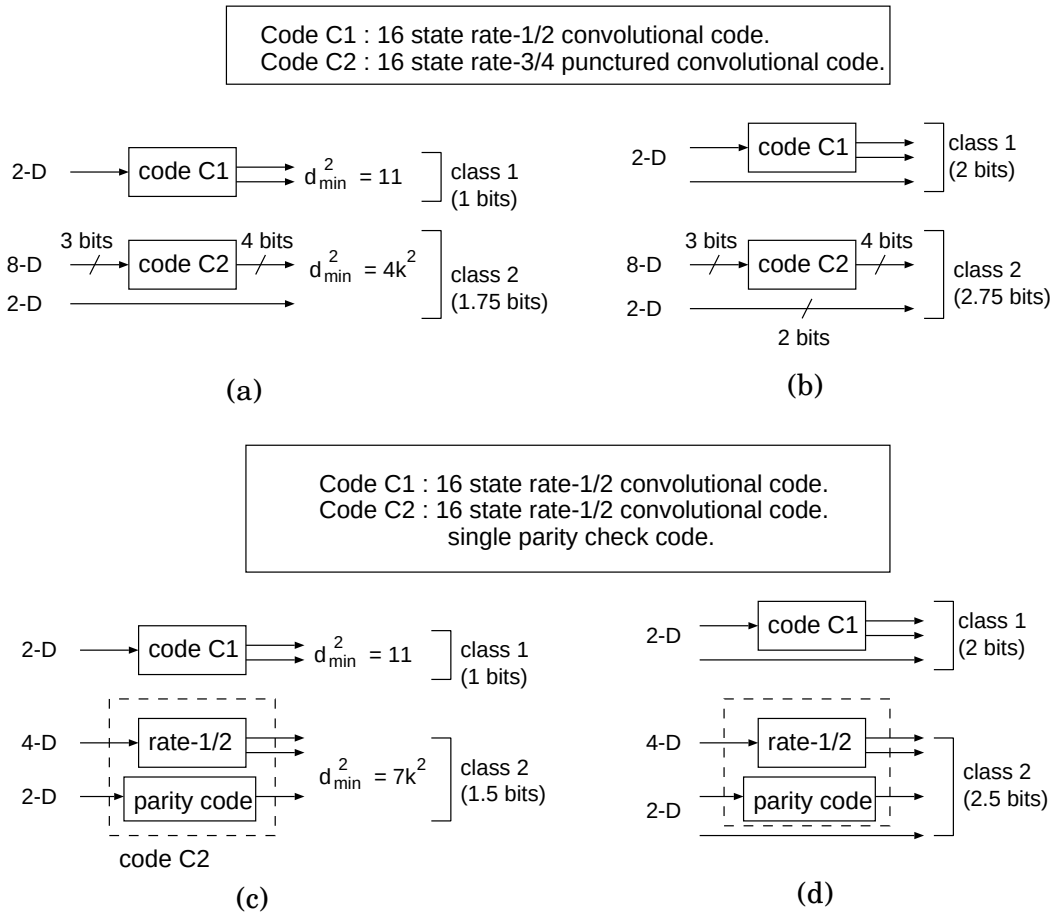


Figure 5.4: Code structures of scheme I using two dimensional signalling. (a) Scheme I-A on 16-QAM. (b) Scheme I-A on 64-QAM. (c) Scheme I-B on 16-QAM. (d) Scheme I-B on 64-QAM.

structure of scheme I-A is illustrated in Figure 5.4 (a) and (b). Scheme I-A uses 16 state rate-1/2 TCM for class 1 data, where the signal looks like 4-PAM or 8-PAM with 4 cosets (A, B, C and D). The code search follows the procedure of the previous section.

In both cases, the squared minimum distance(d_1^2) of code C_1 is 11. The code search results using S-ZRCS are given in Table 5.1. The class 2 is protected by 8 dimensional rate-3/4 punctured TCM, where one output bit of a rate-3/4 convolutional code is assigned for each 2 dimensional signal. d^2_{min} of code C_2 is $4k^2$.

Table 5.1: Code search results using S-ZRCS in scheme I-A and I-B.

Signal size	Average path multiplicity				
	$N_f(11)$	$N_f(12)$	$N_f(13)$	$N_f(14)$	$N_f(15)$
16-QAM	0.13	0.50	1.82	2.03	2.78
64-QAM	0.95	2.54	7.15	9.29	9.43

For example, we calculate the coding gains when k^2 is 1. In scheme I-A on 16-QAM, ρ = 1.25($R = 2.75$), E_P = 0 and E_R = 3.75 dB. Thus,

$$\begin{aligned} \min_{i \in Z^+} \alpha_i &= \alpha_2 = 11.02(\text{dB}). \\ \Gamma_a &= 11.02 - 3.75 = 7.27(\text{dB}). \\ \gamma &= 10\log_{10} 4 - 3.75 = 2.27(\text{dB}). \end{aligned}$$

For 64-QAM, ρ = -0.75($R = 4.75$), E_P = 6.23 dB and E_R = -2.25 dB.

$$\begin{aligned} \min_{i \in Z^+} \alpha_i &= \alpha_1 = 10.83(\text{dB}). \\ \Gamma_a &= 10.83 - 6.23 + 2.25 = 6.85(\text{dB}). \\ \gamma &= 10\log_{10} 4 - 3.75 = 2.04(\text{dB}). \end{aligned}$$

Consider another scheme(scheme I-B) which shows better performance. The code structure is illustrated in Figure 5.4 (c) and (d). The difference here is that for protecting the class 2 data, we use a 16 state rate-1/2 convolutional code which assigns one output bit per 2 dimensional signal and also a single parity check code to improve the intra-coset distance. In this case, we assign 1 bit for class 1 data and 1.5 bits for class 2 data in 16-QAM signalling or 2 bits and 2.5 bits for each classes in 64-QAM signalling. The minimum distance of class 1 data is not changed and the minimum distance of class 2 data $d^2_{min}(C_2)$ is $\min(7k^2, 8k^2)$. The value $7k^2$ comes from the fact that the minimum distance of the 16 state rate-1/2 convolutional code is 7.

Table 5.2: Four dimensional set partitioning.

4-D sub-lattice	*4-D types*	
	parity bit 0	parity bit 1
0 (00)	(AA)	(CC)
1 (01)	(BD)	(DB)
2 (10)	(AC)	(CA)
3 (11)	(BB)	(DD)

For example, we calculate the coding gains when k^2 is 0.7. In 16-QAM signalling, ρ = 1.5($R = 2.5$) E_R = 4.5 dB, and E_P = -0.71 dB. Thus,

$$\begin{aligned} \Gamma_a &= 11.02 + 0.71 - 4.5 = 7.23(\text{dB}). \\ \gamma &= 10\log(7)(0.7) + 0.71 - 4.5 = 3.11(\text{dB}). \end{aligned}$$

For 64-QAM signalling,

$$\begin{aligned} \Gamma_a &= 10.83 - 5.53 + 1.5 = 6.5(\text{dB}). \\ \gamma &= 10\log(7)(0.7) - 5.53 + 1.5 = 2.87(\text{dB}). \end{aligned}$$

Adding a single parity check code on the class 2 data gives us more gain.

In both schemes, we can trade-off the coding gains of the two classes by changing the value of k, i.e. a large value of k gives more gain for class 2 data while increasing the power penalty and thus reducing class 1 coding gain. From now on, we use scheme I-B for class 2 data protection.

5.2.2 Four dimensional signaling

The code structure is shown in Figure 5.6. Four dimensional set partitioning can increase d^2_{min} of class 1 data. The 4-D coset labels are given in Table 5.2, and the

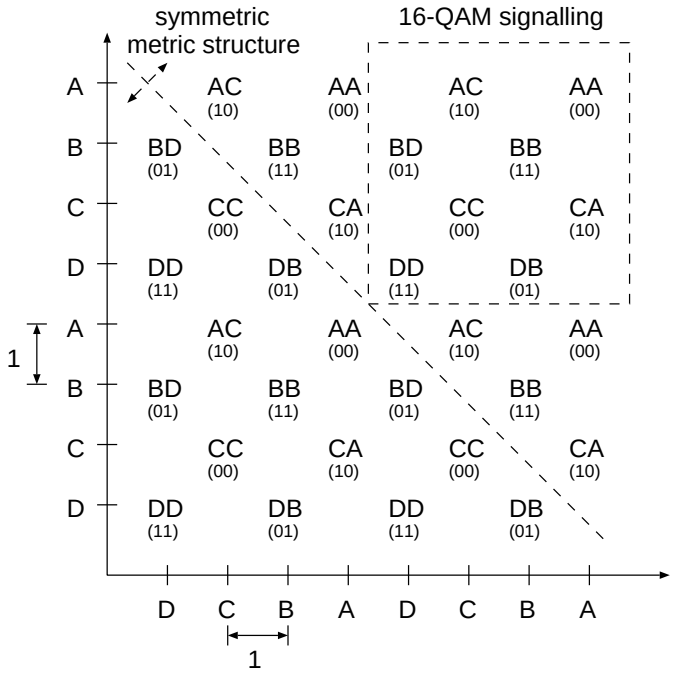


Figure 5.5: Four dimensional metric structure of scheme I-C.

4-D metric structures of 16-QAM and 64-QAM signalling are illustrated in Figure 5.5. When we use 16-QAM, only the signal points in the dashed rectangle can be considered. In 64-QAM signalling, the right upper triangle part and the left lower triangle parts have the same metric structure. So we obtain the metric values for the 16 signal points using the upper triangle. The code search results are in Table 5.3. The code structure is shown in Figure 5.6.

Table 5.3: Code search results using S-ZRCS in scheme I-C.

	Average path multiplicity				
Signal size	$N_f^*(14)$	$N_f^*(16)$	$N_f^*(18)$	$N_f^*(20)$	$N_f^*(22)$
16-QAM	7.92	16.34	21.98	37.12	54.98
64-QAM	26.44	80.0	152.34	288.81	452.53

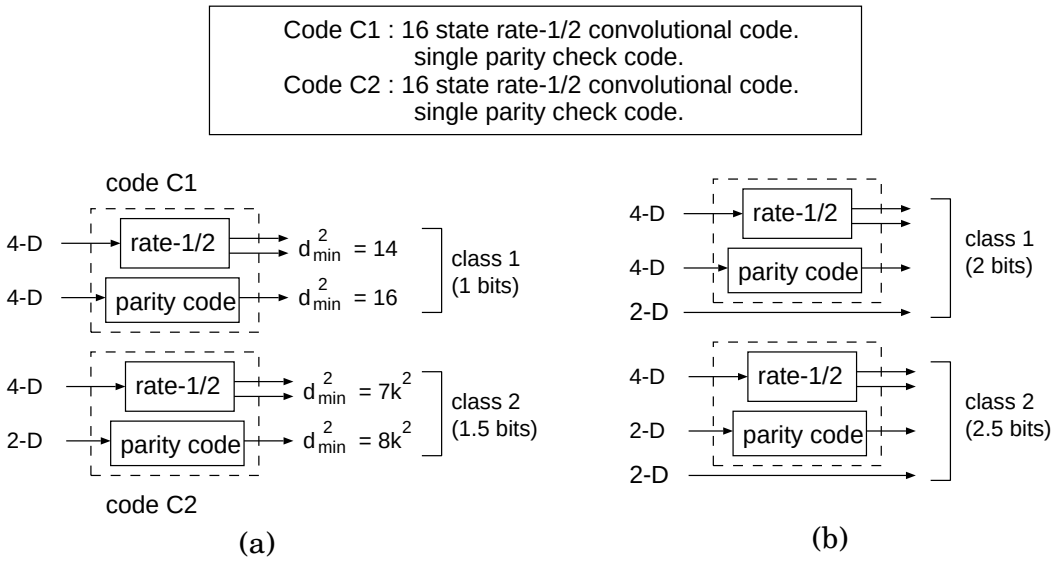


Figure 5.6: Code structures of scheme I using four dimensional signalling. (a) Scheme I-C on 16-QAM. (b) Scheme I-C on 64-QAM.

Consider 16-QAM. d^2_{min} for class 1 data is increased from 11 to 14 but the path multiplicity is also increased by using the 4-D set partitioning. From the code search result,

$$\min_{i \in Z^+} \alpha_i \;=\; \alpha_1 = 11.26(\text{dB}).$$

When k^2 = 0.7, the class 1 data coding gain is

$$\Gamma_a = 11.26 + 0.71 - 4.5 = 7.47(\text{dB}).$$

γ is the same as for 2-D. For 64-QAM,

$$\begin{aligned} \min_{i \in Z^+} \alpha_i &= \alpha_1 = 10.92(\text{dB}), \text{ and} \\ \Gamma_a &= 10.92 - 5.53 + 1.5 = 6.89(\text{dB}). \end{aligned}$$

By using the the 4-D lattice, we can increase the d^2_{min} of the class 1 code. However, the increased path multiplicity reduces some part of gain obtained by increased d^2_{min}.

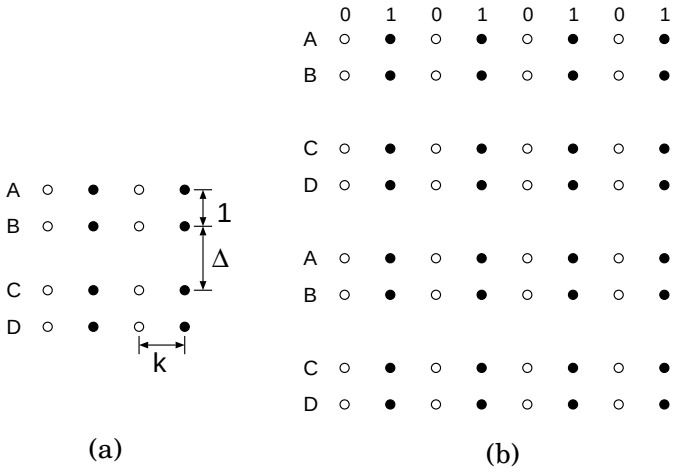


Figure 5.7: Non-uniform signal constellation for scheme II. (a) 16-QAM. (b) 64-QAM.

5.3 Scheme II

Using non-uniform signal constellation may increase d^2_{min} in TCM [43]. The code structure of scheme II is the same as scheme I apart for those non-uniform signal constellations, as illustrated in Figure 5.7. Δ is another variable we use for the trade-off of class 1 and class 2 coding gains. Large Δ gives more gain for class 1 data and less gain for class 2 data. For example, we set $\Delta = 2.0$ and $k^2 = 1.5$. In 16-QAM signalling, d^2_{min} is increased from 11 to 26. $\min \alpha_i = \alpha_1 = 14.43$(dB) ($d^2_1$ is 26 and $N_f(26)$ is 1.5). We pay a power penalty when Δ and k^2 are larger than 1(E_P = 2.43 dB). Then

$$\begin{aligned} \Gamma_a &= 14.43 - 2.43 - 4.5 = 7.5(\text{dB}). \\ \gamma &= 10 \log 7(1.5) - 2.43 - 4.5 = 3.28(\text{dB}). \end{aligned}$$

For 64-QAM, E_P = 8.89 dB and $\min \alpha_i = \alpha_1 = 14.15$(dB) ($N_f(26)$ is 3.94). Then, Γ_a is 6.76 dB and γ is 2.82 dB . We have some gains from using non-uniform signal constellations in 16-QAM signalling by increasing d^2_{min} without increasing N_f.

We have also considered 4-D set-partitioning in this scheme. In this case, d^2_{min} is

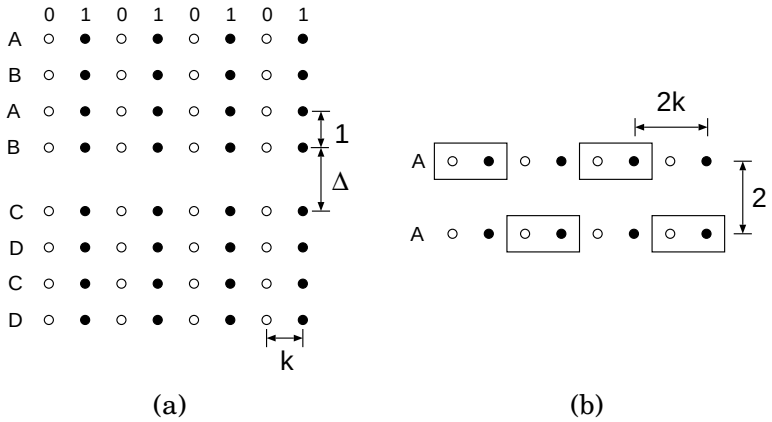


Figure 5.8: Signal constellation for scheme III. (a) 64-QAM constellation. (b) signal points for class 2 data (the points in the rectangle have the same parity bits).

increased from 26 to 29. The calculated coding gains are as follows($k^2 = 1.5$, Δ = 2.0). In 16-QAM, $\min \alpha_i = \alpha_1 = 14.62$(dB) ($N_f^*(29)$ is 4.0) and $\Gamma_a = 7.69$dB. In 64-QAM, $\min \alpha_i = \alpha_1 = 14.39$(dB) ($N_f^*(29)$ is 9.02) and $\Gamma_a = 7.0$ dB. We have some gains over the 2-D scheme. However, for 64-QAM, the constellation expands too much in the Q-phase axis, causing a major power penalty.

5.4 Scheme III

In the previous schemes, the coding gains for 64-QAM are not as good as for 16-QAM. Here we trade a lower rate for important data against coding gains. The signal constellation and coset partitioning is shown in Figure 5.8. The code structure for 2-D signalling is shown in Figure 5.9 (a), where $d^2_{min} = 26$ and $\min \alpha_i = \alpha_1 = 13.97$(dB) ($N_f(26)$ is 7.42). Unlike the previous schemes, the coding for class 2 data is not only a function of k^2 but also confined by the minimum distance of the parity check code, i.e. $d^2_{min} = \min(8, 7k^2)$. Thus the asymptotic coding gain of class 2 data is maximum when

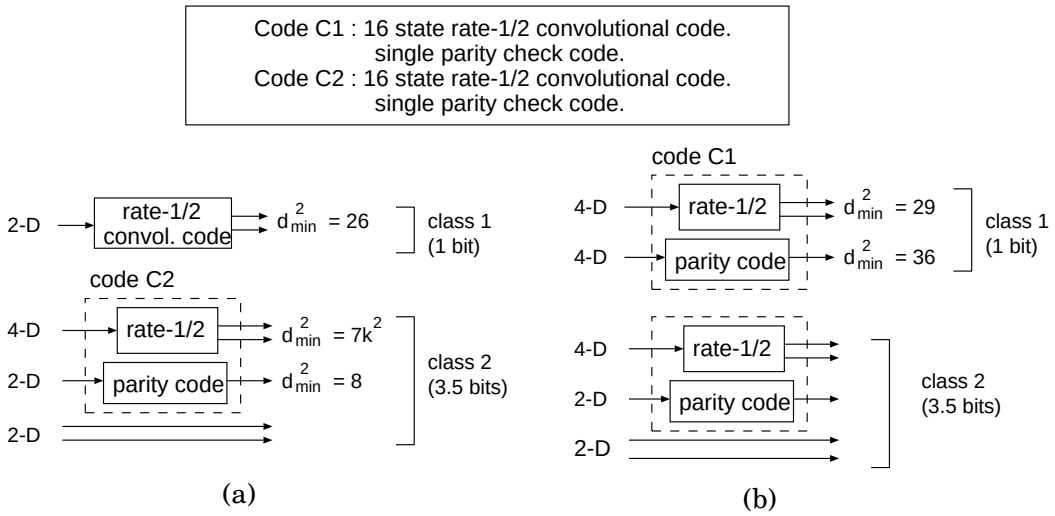


Figure 5.9: Code structures of scheme III. (a) Scheme III-A(2-D signalling). (b) Scheme III-B(4-D signalling).

$k^2 = 8/7$. In this case, $E_P = 7.32$ dB and coding gains are obtained as follows.

$$\begin{aligned} \Gamma_a &= 13.97 - 7.32 + 1.5 = 8.15(\text{dB}). \\ \gamma &= 10\log 8 - 7.32 + 1.5 = 3.21(\text{dB}). \end{aligned}$$

Thus this code structure gives good coding gains to the two data classes with reasonable complexity (we use 16 state rate-1/2 convolutional code for each class and single parity check code for class 2 data protection). Furthermore, we can trade-off the two coding gains by changing the value of Δ.

We can further increase d^2_{min} for the class 1 code by using a 4-D set partitioning. The code structure for 4-D signalling is illustrated in Figure 5.9 (b). The code search result shows that $d^2_{min} = 29$ and $\min \alpha_i = \alpha_5 = 14.04$(dB) ($N^*_f(34)$ is 330.71) and Γ_a is 8.22dB. Unlike the previous schemes, using 4-D signalling does not give us additional coding gain because of the big path multiplicity at the squared distance 34.

The actual coding gain of class 2 data is less than γ because of path multiplicity.

When we let $k^2 = 8/7$, a large N_f for the rate-1/2 convolutional code for class 2 data contributes to the total N_f, which causes a large coding gain degradation. In a real situation, we may use a little bit larger value of k^2 to reduce the path multiplicity of the class 2 coder. Using the value of k^2 slightly larger than 8/7 can reduce N_f and increase the coding gain of class 2 data with the cost of a power penalty. For example, let k^2 = 1.32. There is an additional power penalty of 0.29 dB and Γ_a is 7.86 dB in the 2-D scheme and 7.93 dB in the 4-D scheme. γ is 2.92 dB. Even though the asymptotic coding gain is reduced by 0.29 dB, the actual coding gain of class 2 data is increased by reducing N_f.

5.5 Rotationally invariant UEP

For the previously described codes to be 180^o rotationally invariant, only the class 1 data protection code needs to be 180^o RIC by assigning bit labels for class 2 data in a 180^o symmetric manner such that a 180^o phase rotation does not change the decoding results of the class 2 data. The bit labels for class 2 data with and without 180^o phase rotation are illustrated in Figure 5.10.

Now, consider having 180^o RI class 1 coders. In 2-D codes, flipping the output bits of the rate-1/2 convolutional code causes 180^o rotation of the cosets by assigning two bit labels 00, 01, 10, 11 to cosets A, B, C, D, respectively. Thus using 180^o RI rate-1/2 convolutional codes for class 1 data protection makes both components of the UEP code 180^o rotationally invariant.

This can be applied to 4-D codes. In Table 5.2, a bit flip of the output label corresponds to 180^o rotation of 4-D types and thus a bit flip of parity bits. For example, consider a 4-D type signal (A, A) which is 180^o rotated: we decode (D, D). Then the

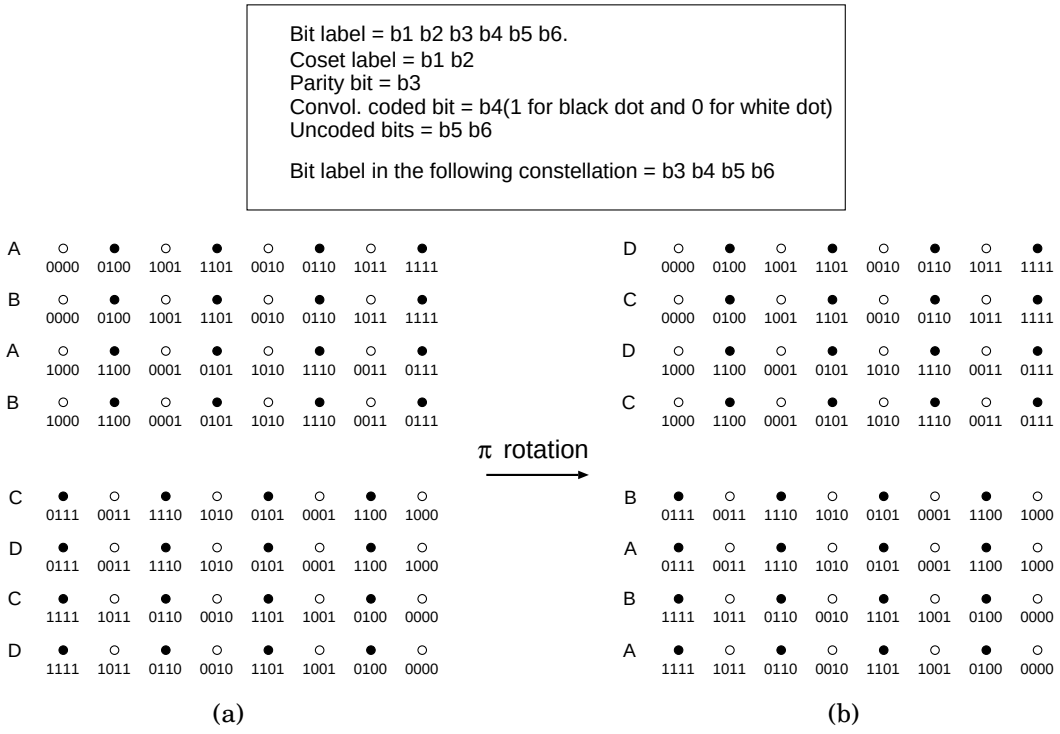


Figure 5.10: Signal constellation for 180^o RI scheme III. (a) Class 2 data bit allocation for 180^o RI code. (b) Received signal after 180^o phase error.

output label is bit flipped from 00 to 11 and the parity bit is flipped from 0 to 1. In this case, we need one more 180^o differential coder for the parity bits. The class 2 code structure is the same as for two dimensions. The structures of the 180^o RI class 1 codes are illustrated in Figure 5.11.

We can make our code 90^o rotationally invariant by using the fact that the in-phase average power P_I and the quadrature-phase average power P_Q are different. At the receiver, we measure the average power of the I and Q-phase directions and decide the polarity of the signal constellation. The remaining 180^o phase ambiguity can be resolved by using the previously described 180^o RI codes. The code search results for the 180^o RI codes are in Table 5.4. In all the schemes but II-B and III-B, the best generators have

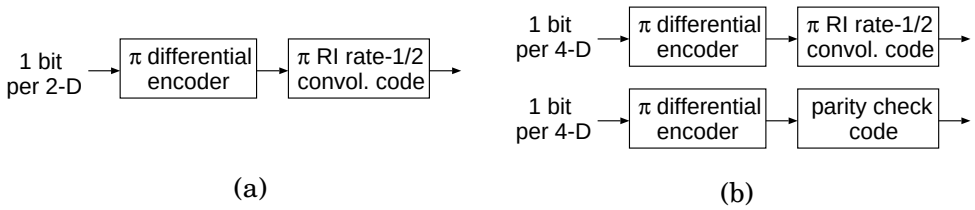


Figure 5.11: Structures of 180^o RI class 1 data protection code in scheme III. (a) 2-D signalling. (b) 4-D signalling.

Table 5.4: Code search results of 16 state convolutional codes for class 1 data protection in scheme I, II and III (180^o RI and non-RI codes).

Scheme	best code		180^o RI-code	
	Generators(octal)	d^2_{min}	Generators(octal)	d^2_{min}
scheme I-A,B(2-D)	23,8	11	23,8	11
scheme I-C(4-D)	23,8	14	23,8	14
scheme II-A(2-D)	23,8	26	23,8	26
scheme II-B(4-D)	34,2	29	25,13	26
scheme III-A(2-D)	23,8	26	23,8	26
scheme III-B(4-D)	34,2	29	25,13	26

a 180^o RI structure. Thus we do not have to pay more to make these codes rotationally invariant. Coding gain calculations are summarized in Table 5.5.

5.6 Simulation results and conclusion

The simulation results for scheme III codes are in Figure 5.12. In this simulation, we fix the value of k^2 to 1.32 to reduce the path multiplicity of the class 2 code. Scheme III-B shows about 7 dB gain for class 1 data and about 2 dB gain for class 2 data at a 10^{-6} bit error rate. This scheme pays about 0.4 dB to have a 90^o RI structure. Scheme III-A having an RI structure shows about 0.3 dB less gain than scheme III-B without rotational invariance. However, scheme III-A could be the best choice if we want to

Table 5.5: Coding gains for the proposed UEP code family.

Scheme	Signal Constellation		Class 1 coding gain	Class 2 coding gain
I-A (2-D) $k^2 = 1.0$	16-QAM 36.36% class 1 data rate = 2.75	RI code	$\Gamma = 6.65$ dB $\Gamma_a = 7.27$ dB	$\gamma = 2.27$ dB
	64-QAM 42.11% class 1 data rate = 4.75	RI code	$\Gamma = 6.43$ dB $\Gamma_a = 6.85$ dB	$\gamma = 2.04$ dB
I-B (2-D) $k^2 = 0.7$	16-QAM 40.0% class 1 data rate = 2.5-1/L	RI code	$\Gamma = 6.62$ dB $\Gamma_a = 7.23$ dB	$\gamma = 3.11$ dB
	64-QAM 44.44% class 1 data rate = 4.5-1/L	RI code	$\Gamma = 6.38$ dB $\Gamma_a = 6.50$ dB	$\gamma = 2.87$ dB
I-C (4-D) $k^2 = 0.7$	16-QAM 40.0% class 1 data rate = 2.5-3/2L	RI code	$\Gamma = 7.67$ dB $\Gamma_a = 7.47$ dB	$\gamma = 3.11$ dB
	64-QAM 44.44% class 1 data rate = 4.5-3/2L	RI code	$\Gamma = 7.43$ dB $\Gamma_a = 6.89$ dB	$\gamma = 2.87$ dB
II-A (2-D) $k^2 = 1.5$ $\Delta = 2.0$	16-QAM 40.0% class 1 data rate = 2.5-1/L	RI code	$\Gamma = 7.22$ dB $\Gamma_a = 7.50$ dB	$\gamma = 3.28$ dB
	64-QAM 44.44% class 1 data rate = 4.5-1/L	RI code	$\Gamma = 6.76$ dB $\Gamma_a = 6.76$ dB	$\gamma = 2.82$ dB
II-B (4-D) $k^2 = 1.5$ $\Delta = 2.0$	16-QAM 40.0% class 1 data rate = 2.5-3/2L	non-RI code	$\Gamma = 7.69$dB $\Gamma_a = 7.69$ dB	$\gamma = 3.28$ dB
	64-QAM 44.44% class 1 data rate = 4.5-3/2L	non-RI code	$\Gamma = 7.23$ dB $\Gamma_a = 7.00$ dB	$\gamma = 2.82$ dB
III-A (2-D) $k^2 = 1.14$ $\Delta = 2.0$	64-QAM 22.22% class 1 data rate = 4.5-1/L	RI code	$\Gamma = 8.33$ dB $\Gamma_a = 8.15$ dB	$\gamma = 3.21$ dB
III-B (4-D) $k^2 = 1.14$ $\Delta = 2.0$	64-QAM 22.22% class 1 data rate = 4.5-3/2L	non-RI code	$\Gamma = 8.80$ dB $\Gamma_a = 8.22$ dB	$\gamma = 3.21$ dB
		RI code	$\Gamma = 8.33$ dB $\Gamma_a = 7.99$ dB	$\gamma = 3.21$ dB

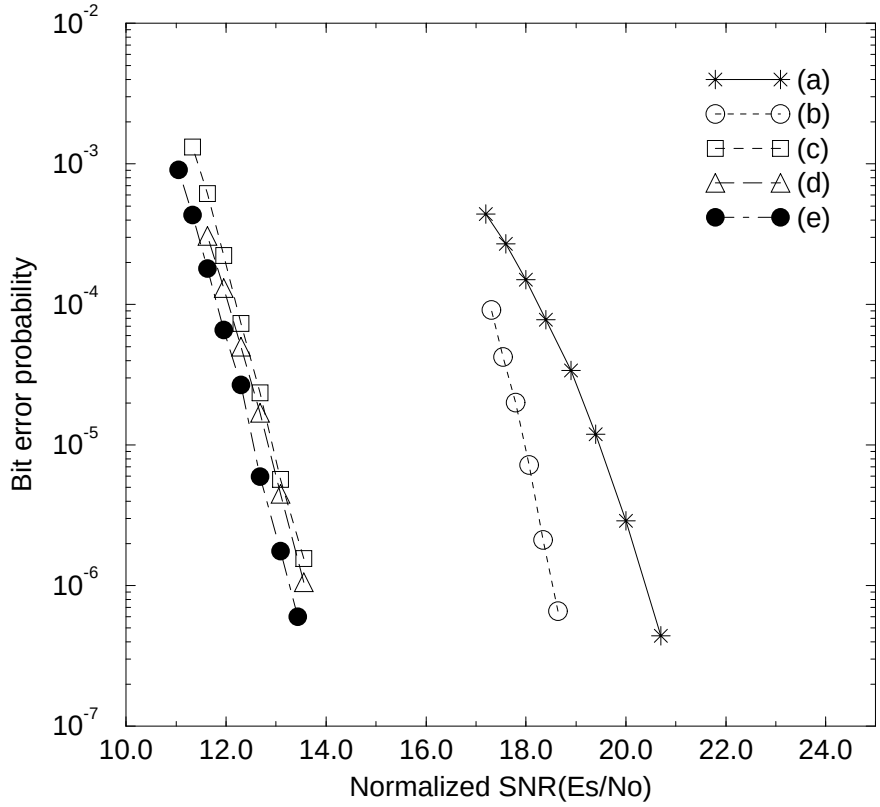


Figure 5.12: Simulation of scheme III in 4-D signalling(k^2 is 1.32) and 180^o RI 2-D and 4-D signalling. (a) Uncoded 16-QAM. (b) Class 2 data protection code. (c) class 1 code (scheme III-B : 90^o RIC on 4-D signalling). (d) class 1 code (scheme III-A : 90^o RIC on 2-D signalling). (e) class 1 code (scheme III-B : non-RIC on 4-D signalling).

have an RI code with less complexity.

The rotationally invariant UEP scheme proposed by Wei [5] provides good protection for the important bits, however the less important bits may even be less reliable than in uncoded transmission. Calderbank and Seshadri [3] suggested two approaches (generalized time sharing and superimposing) for UEP code design and provided greater protection for the less important bits than the Wei codes. Based on the simulation results (25% important bits) in their paper, they have achieved about 6.5 dB gain for the important bits and about 1.5 dB gain for the less important bits at a 10^{-6} bit error rate when they use time sharing. However, when they use a superimposing scheme, the less important bits were not protected.

We have considered a family of multilevel UEP codes based on superimposition for the additive white Gaussian channel. Four way partitioning in a one dimensional lattice combined with a non-uniform signal constellation provides good coding gains for both data classes with reasonable complexity (at most, we use 16 state rate-1/2 convolutional code and single parity check code for each class data protection). Furthermore, we can easily make 90^o rotationally invariant codes by using 180^o rotationally invariant rate-1/2 convolutional code and resolving in-phase and quadrature-phase power.

Chapter 6

HDTV Systems Development Project

The purpose of this project was to implement the channel coding and decoding functions of HDTV (High Definition Television) transceiver standard, proposed by Grand Alliance, in DSP Canvas [16]. DSP Canvas is a group of computer programs that when used together constitute a powerful computer aided design and development tool. It allows development teams to work on different modules or aspects of a system without concern for how other parts of the system have been coded or represented. Another advantage of DSP Canvas is that it permits convenient changes of the arithmetic precision in individual blocks, enabling systems designers to quickly evaluate which blocks are most sensitive to finite precision effects. Our simulation tools are presently in use by engineers at Thompson Consumer Electronics and David Sarnoff Research Center for HDTV receiver development. As a side benefit of this work, we developed a decoding strategy yielding performance better than that so far presented before the standards body. This chapter is organized as follows. Section 1 briefly explains the error correction circuits of HDTV transceiver. Optimum branch metric calculation methods under NTSC interference will be discussed in section 2. Section 3 addresses the questions of

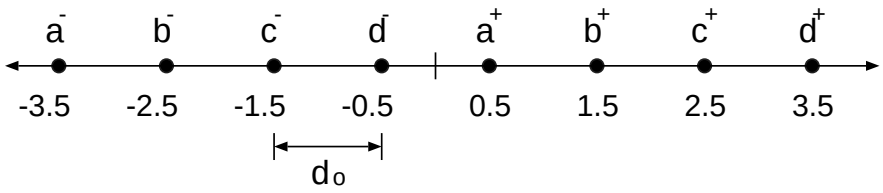


Figure 6.1: 8 PAM constellation.

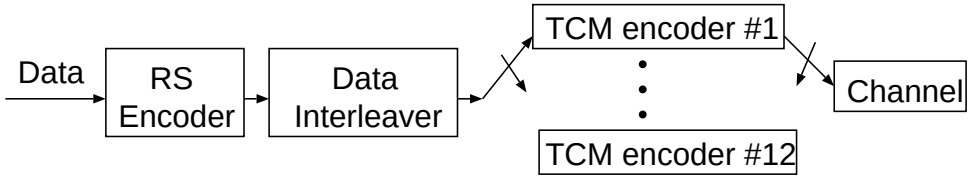


Figure 6.2: HDTV Transmitter.

the minimum level of quantization, traceback depth and bits for dynamic range in the Viterbi decoder.

6.1 HDTV transceiver

Figure 6.2 is a reduced description of the HDTV transmitter. We use 8-PAM signalling as the equivalent baseband form of 8-VSB (Vestigial Side Band). We use a 4 state rate-1/2 convolutional code in TCM. The four way partitioning of 8-PAM is described in Figure 6.1, where the label (a, b, c, d) denotes the corresponding coset and the superscript (+ or -) identifies one of the two members in the same coset.

There are two receiving modes corresponding to the NTSC (regular television) interference situations. The interference level is small near a HDTV transmitter, while we have to consider the NTSC interference at the edge of coverage. Since most of the energy in the NTSC signal is in three frequency tones, a tone rejection filter can potentially lead to significantly better performance. The receiver decides whether there

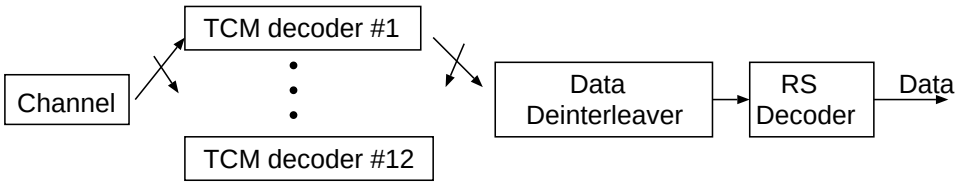


Figure 6.3: Receiver structure of NTSC non-interfered case.

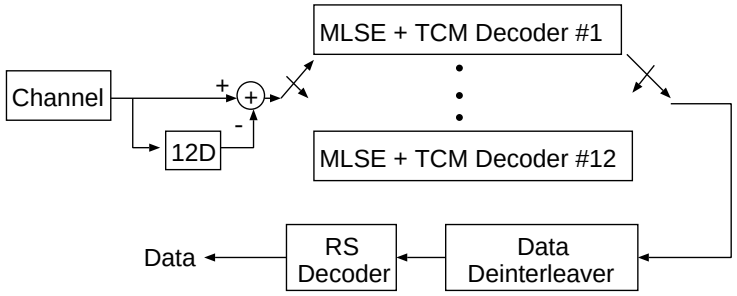


Figure 6.4: Receiver structure of NTSC interfered case.

is significant NTSC interference or not at the first stage. If there is, the receiver is in NTSC-rejection mode and uses a 1-12D FIR filter, otherwise it is in the normal mode. The structure of the normal mode receiver and the NTSC-interfered mode receiver are shown in Figure 6.3 and Figure 6.4, respectively.

6.2 Branch metric calculation in TCM decoder with NTSC interference

We now describe how to perform branch metric calculations and the ACS (Add-Compare-Select) operation in the TCM decoder under the condition of NTSC interference. Because of the combination of interleaving and the FIR filtering process for NTSC rejection, the input to the TCM decoder is a partial response signal, i.e. the channel looks

like a 1-D channel, which can be explained by the following way.

Let $\{x_1, x_2, ..., x_i, ...\}$ be the output sequence of the data interleaver in the transmitter. Since two input bits mapped onto one modulated signal, x_i has two bits of information. If x_i modulo 12 = j, x_i is encoded by j-th TCM encoder and x_{i+12} will be the next input of the encoder because of the one-to-twelve multiplexer. Let $\{y_1, y_2, ..., y_i, ...\}$ and $\{z_1, z_2, ..., z_i, ...\}$ be the received sequence and 1-12D filtered sequence, respectively. Then $y_i = x_i + n_i$ and $z_i = y_i - y_{i-12}$, where n_i is a white Gaussian noise at time index i. If z_i modulo 12 = j, z_i $(= y_i - y_{i-12})$ is decoded by the j-th TCM decoder. Since x_{i-12} and x_i are consecutive inputs in the j-th encoder, in the view point of j-th decoder, the signal z_i can be considered as a signal resulting from a 1-D partial response channel with Gaussian noise.

6.2.1 Combining maximum likelihood sequence estimation and TCM decoding

The optimum decoding strategy in the linear intersymbol interference channel (ISI) is maximum likelihood sequence estimation (MLSE) [48]. In uncoded systems, it is well-known that at least asymptotically MLSE never performs worse than ideal decision feedback equalizer (DFE). MLSE can be implemented with the Viterbi algorithm. Assuming the length of ISI is K and the modulated signal can have M values, then the number of states in the trellis is M^K. In our system, the number of states is eight (M is 8 and N is 1).

We have to use MLSE before the TCM decoder or else suffer a large performance penalty in demodulating the partial response signal. However, in this case, hard decisions at the TCM decoder input are unavoidable, which results in about 3 dB more

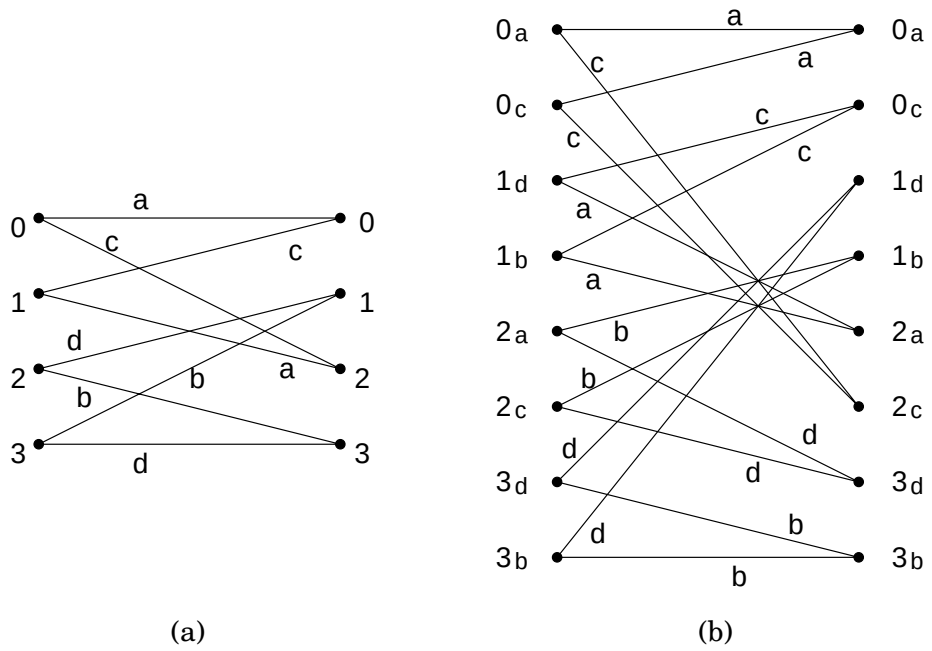


Figure 6.5: Illustration of state splitting to make a partial response(PR) trellis. (a) Four state ordinary trellis. (b) Eight state partial response trellis.

performance loss. As an another alternative, we may consider a combined version of MLSE and TCM decoding, where the number of states required is the multiple of the number of states in MLSE and TCM.

Eyuboglu and Qureshi have proposed reduced-state sequence estimation (RSSE) [40] which can achieve nearly the performance of MLSE at significantly reduced complexity. The result was further extended into the combined version of MLSE and TCM decoding [41]. We have used their schemes and could decrease the number of states from 32 to 8. The procedure is described as follows. We split the four states in the original trellis to produce an eight states trellis as illustrated in Figure 6.5. As we can see in Figure 6.5, only the branches with output a can go to state $0_a,\ 2_a$ and only the branches with output c can go to state $0_c,\ 2_c$, etc... That is, the subscript in state label comes from the coset label of the branches merging in that state.

All eight states have the previous surviving signals in their memories. For example, the state 0_a memory will contain one of the two members of coset a, i.e. 0_a has one of a^- and a^+ in its memory. The previous surviving signals will be used in the branch metric calculation in the next extension. The 8 states also have accumulated metrics in their memories. Thus each state has two values in their memory – accumulated metric and previous surviving signal.

Now we will describe the branch metric calculation and ACS procedure in general. Consider two branches from states s_1 and s_2 to state s_f, and define them as B_1 and B_2, respectively. States s_1 and s_2 have previous surviving signals γ_1 and γ_2, respectively. From the partial response (PR) trellis in Figure 6.5 (b), we know that the same coset ξ labels both branches B_1 and B_2. Then the branch metrics of the two branches are expressed as

$$\text{metric } B_1 \quad = \quad \min [\, \beta_1^+, \beta_1^- \,]. \tag{6.1}$$

$$\text{metric } B_2 \quad = \quad \min [\, \beta_2^+, \beta_2^- \,]. \tag{6.2}$$

where

$$\beta_1^+ \;=\; (r + \gamma_1 - \xi^+)^2, \;\; \beta_1^- \;=\; (r + \gamma_1 - \xi^-)^2$$

$$\beta_2^+ \;=\; (r + \gamma_2 - \xi^+)^2, \;\; \beta_2^- \;=\; (r + \gamma_2 - \xi^-)^2$$

where r is the signal after FIR filtering.

Let ξ_1 be ξ^+ or ξ^- which corresponds to the minimum of β_1^+ and β_1^- and ξ_2 be ξ^+ or ξ^- which corresponds to the minimum of β_2^+ and β_2^-. Then the surviving signal γ_f at state s_f is the ξ_1 or ξ_2 which corresponds to the surviving branch. This can be described by the following example. Consider two branches merging into state 0_a. Define the previous surviving signal at state 0_a to be $\tilde{a}$ which can be a^+ or a^-, and the previous

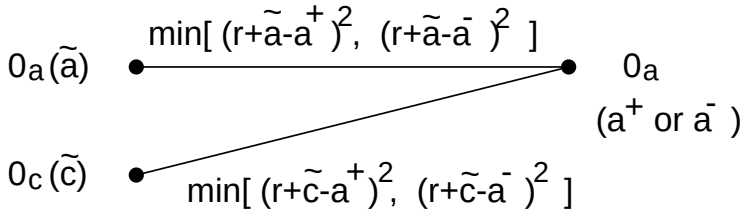


Figure 6.6: Branch metric calculation in PR trellis.

surviving signal at state 0_c to be $\tilde{c}$ which can be c^+ or c^-. Then the branch metric joining states 0_a and 0_a is

$$\min\,[\ (r+\tilde{a}-a^+)^2,\ (r+\tilde{a}-a^-)^2\ ]$$

and the branch metric joining states 0_c and 0_a is

$$\min\,[\ (r+\tilde{c}-a^+)^2,\ (r+\tilde{c}-a^-)^2\ ]$$

The surviving signal at state 0_a is a^+ or a^- which makes the accumulated metric minimum. This example is explained in Figure 6.6. The signal in the parenthesis beside state labels are surviving signal labels.

More detailed example is illustrated in Figure 6.7. The two values in the parenthesis in Figure 6.7 are in the format of (accumulated metric, previous surviving signal). We can see that the accumulated metric and previous surviving signal of state 0_a are 7.2 and a^+, respectively. The accumulated metric and previous surviving signal of state 0_c are 6.8 and c^-, respectively. Let the previous received signal be 0.7 and present signal be 1.2, then 1-D filtering results in total of 0.6. r is 0.6. Let the metric of the branch $0_a - 0_a$ be $B1$, then

$$\begin{aligned} B1 &= \min\,[(r+\tilde{a}-a^+)^2, (r+\tilde{a}-a^-)^2] \\ &= \min\,[(r+a^+-a^+)^2, (r+a^+-a^-)^2] \\ &= \min\,[0.6^2, (0.6+0.5^+-(-3.5))^2] \ =\ 0.36 \end{aligned}$$

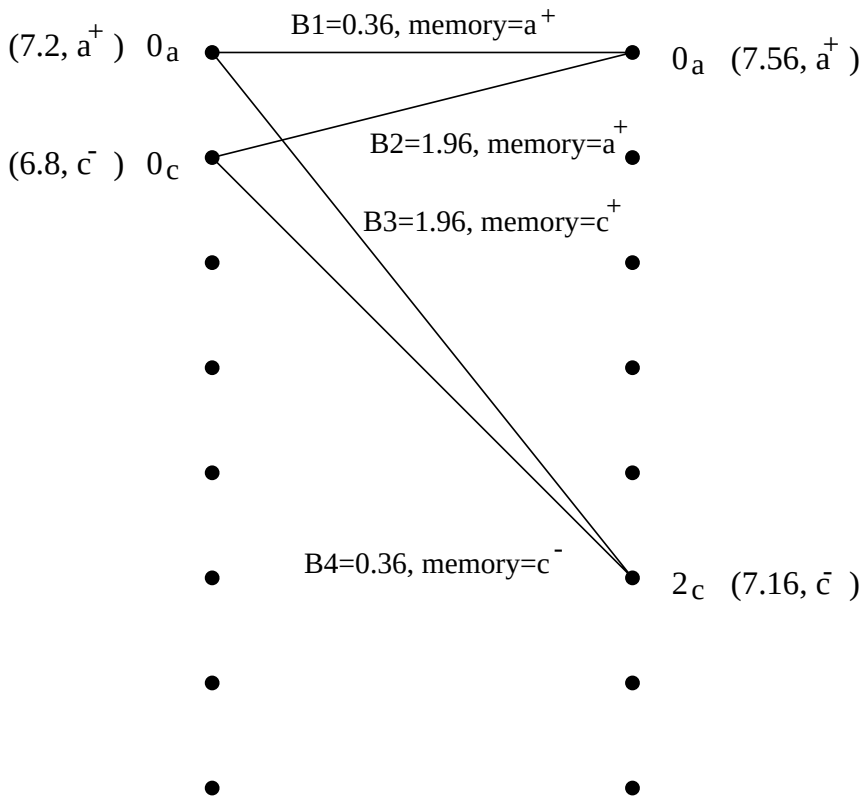


Figure 6.7: Numerical example of branch metric calculation and ACS.

Since $(r+\tilde{a}-a^+)^2$ is smaller than $(r+\tilde{a}-a^-)^2$, we put a^+ in the temporary memory of the branch 0_a-0_a. Let the metric of the branch 0_c-0_a be $B2$, then

$$\begin{aligned} B2 &= \min\left[(r+\tilde{c}-a^+)^2,(r+\tilde{c}-a^-)^2\right] \\ &= \min\left[(r+c^--a^+)^2,(r+c^--a^-)^2\right] \\ &= \min\left[(0.6+(-1.5)-0.5)^2,(0.6+(-1.5)-(-3.5))^2\right] \;=\; 1.96 \end{aligned}$$

Since $(r+\tilde{c}-a^+)^2$ is smaller than $(r+\tilde{c}-a^-)^2$, we put a^+ in the temporary memory of the branch 0_c-0_a. Let the metric of the branch 0_a-2_c be $B3$, then

$$\begin{aligned} B3 &= \min\left[(r+\tilde{a}-c^+)^2,(r+\tilde{a}-c^-)^2\right] \\ &= \min\left[(r+a^+-c^+)^2,(r+a^+-c^-)^2\right] \\ &= \min\left[(0.6+0.5-2.5)^2,(0.6+0.5^+-(-1.5))^2\right] \;=\; 1.96 \end{aligned}$$

Since the signal c^+ corresponds to the minimum of the two values, we put c^+ in the temporary memory of the branch 0_a-2_c. Let the metric of the branch 0_c-2_c be $B4$, then

$$\begin{aligned} B4 &= \min\left[(r+\tilde{c}-c^+)^2,(r+\tilde{c}-c^-)^2\right] \\ &= \min\left[(r+c^--c^+)^2,(r+c^--c^-)^2\right] \\ &= \min\left[(0.6+(-1.5)-0.5)^2,(0.6+(-1.5)-(-1.5))^2\right] \;=\; 0.36 \end{aligned}$$

Since the signal c^- corresponds to the minimum of the two values, we put c^- in the temporary memory of the branch 0_c-2_c.

We have finished the branch metric calculation of the branches in this example, and now consider the ACS procedure starting with state 0_a. We compare 7.2+$B1$ and 6.8+$B2$. If 7.2+$B1$ has a smaller value, the new accumulated metric and the previous surviving signal of state 0_a are 7.2+$B1$ and the signal in the temporary memory of the

branch $0_a - 0_a$, respectively. Otherwise, the new accumulated metric and the previous surviving signal of state 0_a are 6.8+$B2$ and the signal in the temporary memory of the branch $0_c - 0_a$, respectively. Since $B1$ is 0.36 and $B2$ is 1.96, 7.2+$B1$ has a smaller value. As a result, new accumulated metric and the previous surviving signal of state 0_a are 7.56 and a^+, respectively.

Consider the ACS in state 2_c.

$$\min\left[7.2 + B3, 6.8 + B4\right] \;=\; \min\left[7.2 + 1.96, 6.8 + 0.36\right] \;=\; 7.16$$

Since 6.8+$B4$ has a smaller value, the surviving branch is $0_c - 2_c$. As a result, the new accumulated metric and the previous surviving signal of state 2_c are 7.16 and c^-, respectively. The signal c^- was in the memory of the surviving branch $0_c - 2_c$.

6.2.2 Noise prediction at the branch metric calculation

Our simulation reveals that the combined MLSE and TCM decoding of the signals after 1-D FIR filtering shows about 3 dB performance loss compared with the original 4 state TCM without filtering. This is mainly due to the 3 dB noise enhancement resulting from the 1-D FIR filtering. Use of the noise predictive metric mitigates the noise enhancement [39]. The noise prediction coefficients under the minimum mean squared error criterion is derived as follows.

Let $\bar{n} = \{..., n_{i-1}, n_i, n_{i+1}, ...\}$ be a sequence of white Gaussian noise samples with variance σ^2, and let $\bar{u} = \{..., u_{i-1}, u_i, u_{i+1}, ...\}$ be the received noise sequence after 1-D filtering. Then, $u_i = n_i - n_{i-1}$ and

$$E[u_i u_j] = \begin{cases} 2\sigma^2 & \text{if } i = j \\ \sigma^2 & \text{if } |i - j| = 1 \\ 0 & \text{otherwise} \end{cases} \tag{6.3}$$

We can estimate the value u_k by using the P-th order linear prediction, where we predict the present value based on a scaled sum of the previous P samples. To be specific, the predicted value $\tilde{u_k}$ is defined as

$$\tilde{u_k} = \sum_{i=1}^{P} p_i u_{k-i}$$

where the p_i's are prediction coefficients. The prediction error e_P can be obtained as

$$e_P = u_k - \tilde{u_k} = u_k - \sum_{i=1}^{P} p_i u_{k-i} \tag{6.4}$$

Let q_0 be 1 and $q_i = p_i$, i=1,2,...,P. Then, the minimum mean squared error $E[e_P^2]$ is expressed as

$$\begin{aligned} E[e_P^2] &= E\left[\left(u_k - \sum_{i=1}^{P} p_i u_{k-i}\right)\left(u_k - \sum_{j=1}^{P} p_j u_{k-j}\right)\right] \\ &= E\left[\sum_{i=0}^{P}\sum_{j=0}^{P} q_i u_{k-i} q_j u_{k-j}\right] \\ &= \sum_{i=0}^{P} q_i^2 E[u_{k-i}^2] + 2\sum_{i=0}^{P-1} q_i q_{i+1} E[u_{k-i}u_{k-i-1}] \\ &= 2\sigma^2\left(\sum_{i=0}^{P} q_i^2 + \sum_{i=0}^{P-1} q_i q_{i+1}\right) \end{aligned} \tag{6.5}$$

Since $E[e_P^2]$ is a quadrature function of variables q_i's, it has its minimum where

$$\frac{\partial E[e_P^2]}{\partial q_i} = 0, \quad i \in \{0, 1, ..., P\}. \tag{6.6}$$

We obtain the optimal prediction coefficients p_i's from the equations (6.5) (6.6) and the values are

$$p_i = 1 - \frac{i}{P+1}, \quad i \in \{1, 2, ..., P\} \tag{6.7}$$

The resulting mean squared error is

$$E[e_P^2] = 2\sigma^2\left(1 - \frac{P}{2(P+1)}\right) \tag{6.8}$$

thus we can see that $E[e_P^2]$ approximates the uncorrelated Gaussian noise variance, as the prediction order P increases.

For this noise prediction, we need the previous noise values, which can be obtained by subtracting the correctly decoded signal from received signal. However, the decoding delay in TCM makes the correct estimation of the previous noise sequence difficult. We did two simulations using the noise predictive metric. One is the ideal case, i.e. it uses the correct previous noise values. The other uses sub-optimally estimated previous noise values for the particular branch, where each state needs memory of the previous noise estimate. In both cases, we have used first order and second order noise prediction. The simulation results are shown in Figure 6.8. The sub-optimal method of using a second order predictor shows about 1.4 dB gain over the squared distance metric, while the ideal case shows about 2.5 dB gain. Other simulations showed that the sub-optimal method using 10-th order predictor has performance close to that of the ordinary TCM without 1-D filtering (less than 0.5 dB difference).

The enhanced noise power due to the 1-D filter can be suppressed by using noise prediction. Since we know the correlation information of the noise, we can perfectly suppress the enhanced amount of noise. However, an infinite length FIR filter is necessary to do inverse 1-D filtering. Linear prediction using a finite length FIR filter approximates the inverse 1-D filtering. A low order noise prediction scheme for the branch metric calculation has been proposed by Eyuboglu [39]. Considering that the prediction error decreases as the order of filter increases, we have used a higher order predictor and obtained good results. By using 10-th order prediction, we can suppress most of the enhanced noise. As a result, we have an additional 3 dB gain over the performance normally expected from the standard.

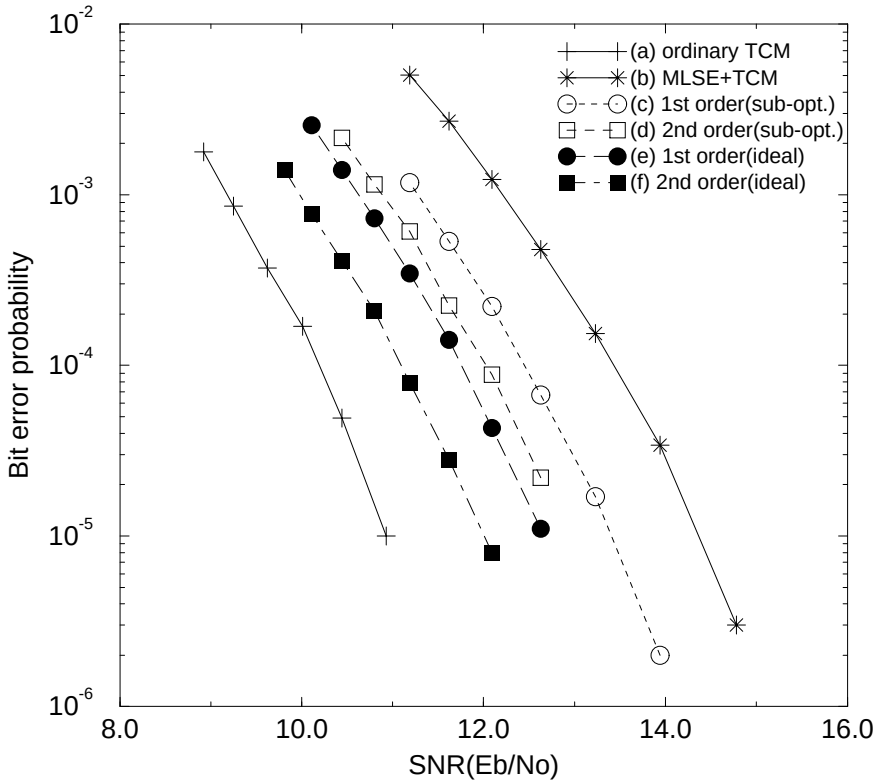


Figure 6.8: Simulation results of TCM in HDTV transceiver with different metrics. (a) 4 state ordinary TCM without NTSC interference. (b) Combined MLSE and TCM decoding with ordinary squared metric. (c,d) Combined MLSE and TCM decoding with first and second order noise predictive metric, respectively (sub-optimal). (e,f) Combined MLSE and TCM decoding with first and second order noise predictive metric, respectively (ideal).

6.3 Effect of fixed point precision on TCM decoder

While theoretical results for coding assume real arithmetic, of course in practice fixed point arithmetic is used. This section addresses the questions of the minimum level of quantization and traceback depth for comparable performance with the ideal decoding. Another question is how many bits of dynamic range are required when we are using two's complement arithmetic to avoid metric rescaling [45].

There are two decoding modes in the HDTV transceiver. One is a decoder with NTSC interference and the other is a decoder without NTSC interference. In both cases, we will find the three factors (minimum level of quantization, traceback depth and number of bits for the minimum required dynamic range) by simulation. We first describe the quantization method we have used, and explain the way to avoid metric rescaling by using two's complement arithmetic. Simulation results will then be presented.

6.3.1 Quantization

We divide the one dimensional 8 PAM signal space into $N(\leq 2^n)$ sections, where n is the number of quantization bits. We use uniform quantization. One condition to be satisfied is that the 8 signal points must be located at the center of the section they belong to. This situation is illustrated in Figure 6.9. From Figure 6.9, there are a total

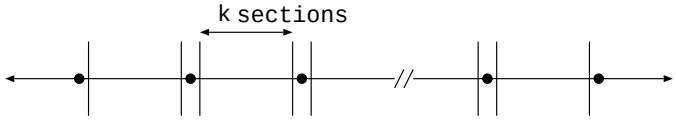


Figure 6.9: Quantization of 8 PAM.

of $7k + 8$ sections. When we use n bits for quantization, the value of k is obtained as

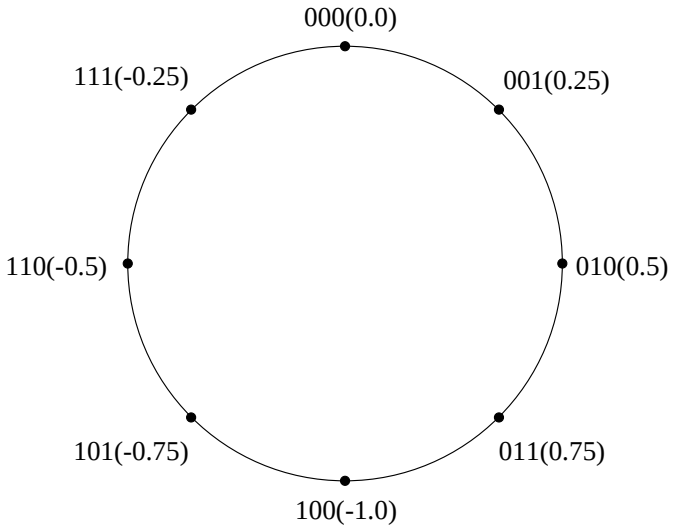


Figure 6.10: Modulo structure of two's complement arithmetic.

follows.

$$7k + 8 \leq 2^n \quad \rightarrow k = \left\lceil \frac{2^n - 8}{7} \right\rceil$$

For example, when we use 6 bits for quantization, $k = \lceil \frac{56}{7} \rceil = 8$ and when we use 7 bits for quantization, $k = \lceil \frac{120}{7} \rceil = 17$.

6.3.2 Metric rescaling

Without metric rescaling, the accumulated metric will overflow. In the rescaling scheme, at each iteration the minimum metric is subtracted from all metrics. The use of two's complement arithmetic avoids rescaling subtraction due to the modulo structure of two's complement arithmetic. This is illustrated in Figure 6.10 when we use three bits. The values in the parentheses are the corresponding decimal values. Even when a positive value overflows into a negative value, nothing special happens in binary notation as the whole binary notation has a modulo structure. The only indication of overflow is the flip of the sign bit. The bit location of the two's complement number is in Figure

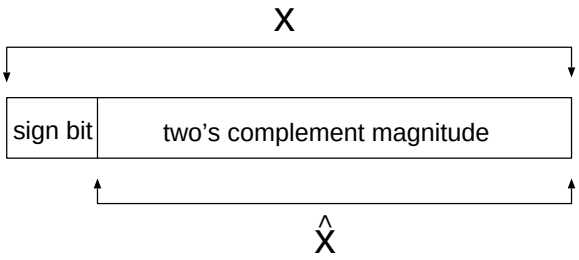


Figure 6.11: Bits location of two's complement number.

6.11. We define the *two's complement magnitude* $\hat{x}$ to be the binary value of x except for the sign bit. Let a and b be two accumulated metrics to be compared, and d_{max} be the maximum possible accumulated metric difference such that,

$$|a - b| \leq d_{max}. \tag{6.9}$$

Then the dynamic range D must be no less than $2d_{max}$ to avoid performance degradation due to overflow.

The compare and selection algorithm is explained as follows. When the sign bits of the two competing branches are the same, the surviving branch is the one with the smaller value. When the sign bits of two competing branches are different, one of them overflowed. We can see from equation (6.9) and Figure 6.12 that the one with smaller two's complement magnitude overflowed, so we choose the one which has larger two's complement magnitude. Two possible cases of different sign bit metric comparison are illustrated in Figure 6.12 ((a_1, b_1) pair and (a_2, b_2) pair). Consider first (a_1, b_1). b_1 overflowed because $|a_1 - b_1| \leq d_{max}$, and we choose a_1. For the same reason we can see that a_2 overflowed in the (a_2, b_2) comparison, so we choose b_2. In both cases, we choose the one with the larger two's complement magnitude. The metric comparison and selection is summarized as follows. When the sign bits are the same, we choose the one with smaller two's complement magnitude and when the sign bits are different, we

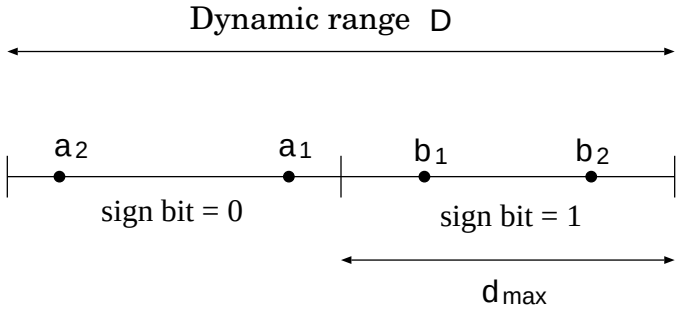


Figure 6.12: Two possible cases of different sign bit branch metric competition.

choose the one with the larger two's complement magnitude. Given below are several numerical examples, where we have 5 bits of dynamic range.

i) a = 01001, b = 01110 : we choose a because a and b have the same sign bits and $\hat{a}(1001) < \hat{b}(1110)$.

ii) a = 10101, b = 10010 : we choose b because a and b have the same sign bits and $\hat{a}(0101) > \hat{b}(0010)$.

iii) a = 10011, b = 01010 : we choose b because a and b have different sign bits and $\hat{a}(0011) < \hat{b}(1010)$.

iv) a = 00111, b = 11100 : we choose b because a and b have different sign bits and $\hat{a}(0111) < \hat{b}(1100)$.

6.3.3 Simulation results

There are two decoder modes: NTSC interfered and non-interfered modes. We simulate the quantization effect, finite traceback depth effect and finite dynamic range effect in both cases. Then we can decide on the minimum required number of quantization bits, the traceback depth and the number of bits for the internal arithmetic without overflow. Two million symbols are transmitted in every simulation.

First, consider the TCM scheme without NTSC interference. The simulation in Figure 6.13 is done under the condition of a very large traceback depth and enough dynamic range, and shows that 6 bits are enough for quantization. Figure 6.14 shows that a traceback depth of 15 (five times the constraint length) is enough. To find the starting state for the traceback, we have to compare the accumulated metrics of all the states. When the number of states is large, we need more comparisons. We can save that comparison cost by starting the traceback process from arbitrary states: however, in this case we need a larger traceback depth. Figure 6.15 shows that the required traceback depth is almost two times the original one. The results in Figure 6.16 indicate that a 13 bit range is enough for the internal arithmetic without overflow.

The simulation results for a TCM decoder with NTSC interference are in Figure 6.17, Figure 6.18 and Figure 6.19. 6 bits of quantization, traceback depth 15 and 14 bits of dynamic range suffices according to the results. This scheme needs a dynamic range of one more bit than the scheme without NTSC interference because of the 1-D filtering process.

Finally, we simulate TCM without NTSC interference using 6 bit quantization, traceback depth 15 and 13 bits range of arithmetic and the TCM with NTSC interference using 6 bit quantization, traceback depth 15 and 14 bits range of arithmetic. We transmitted five million symbols in this simulation. These results are compared with the ideal cases in Figure 6.20. The ideal cases use floating point arithmetic and very large traceback depth (here we have used traceback depth of 60). The performance differences are only about 0.1 dB.

We have considered the effects of fixed point arithmetic in both TCM decoding schemes. We use two's complement arithmetic to avoid metric rescaling. Simulation results show that 6 bits of quantization and a traceback depth of 15 are enough for

both modes of TCM. 13 bits of dynamic range are enough for the code without NTSC interference and 14 bits dynamic range are enough for the code with NTSC interference. The one bit difference is due to the 1-D filtering process. The performance differences between using these parameters versus ideal operation is only about 0.1 dB.

6.4 Summary

We have developed the optimum decoding method under the HDTV standard constraints. The receiver has two modes of operation corresponding to the NTSC interference situations. When the receiver is in NTSC-rejection mode, the decoder has to manage the partial response signal and a 3 dB noise enhancement due to 1-D filtering in front of the receiver. Those problems were solved by using RSSE and higher order noise prediction in the TCM decoder. The noise prediction gives us about 3 dB more gain over the performance normally expected from the standard.

We have found the minimum level of quantization and the smallest traceback depth for comparable performance with ideal decoding. Another question is how many bits of dynamic range are required when we are using two's complement arithmetic to avoid metric rescaling [45]. Those three factors (minimum level of quantization, traceback depth and number of bits for the minimum required dynamic range) were obtained by simulation. Simulation results show that 6 bits of quantization and a traceback depth of 15 are enough for both modes of receiver. 13 bits of dynamic range are enough for the code without NTSC interference and 14 bits dynamic range are enough for the code with NTSC interference. The one bit difference is due to the 1-D filtering process. The performance differences between using these parameters versus ideal operation is only about 0.1 dB.

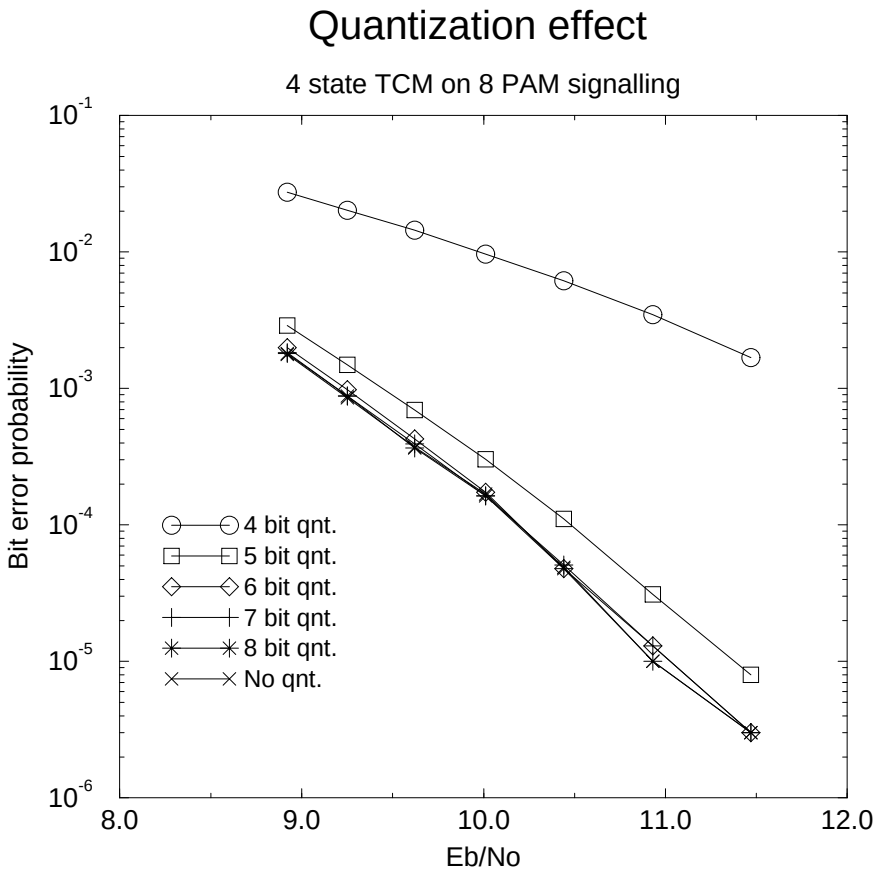


Figure 6.13: Quantization effects of 4 state TCM on 8 PAM signalling.

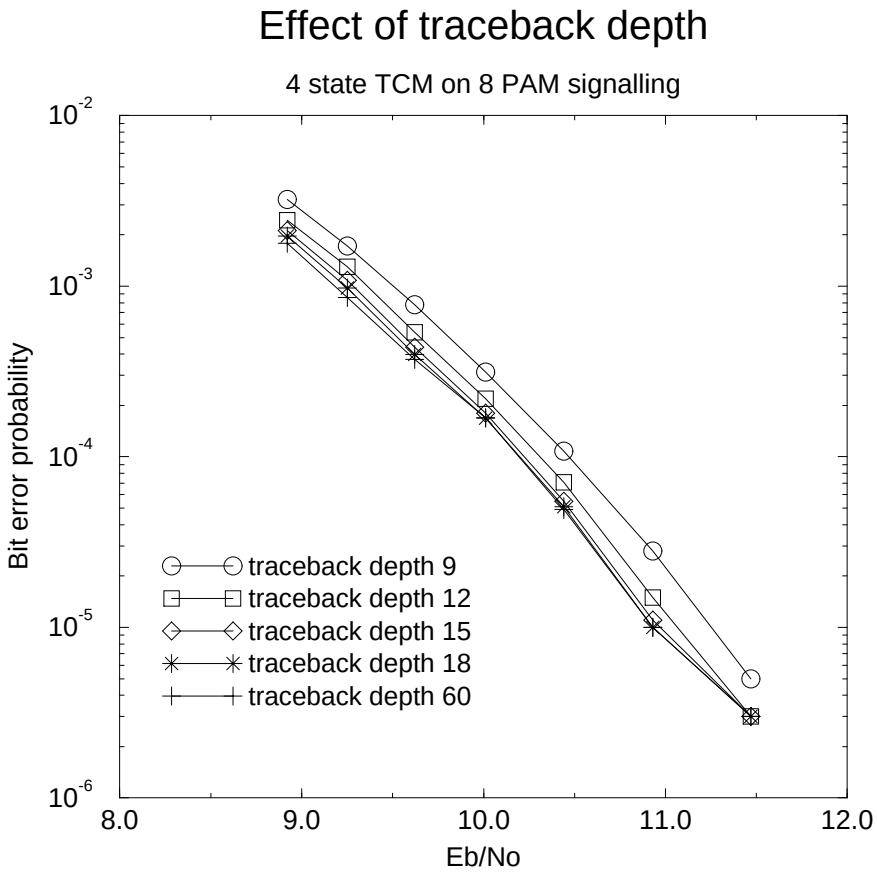


Figure 6.14: Finite traceback depth effects of 4 state TCM on 8 PAM signalling.

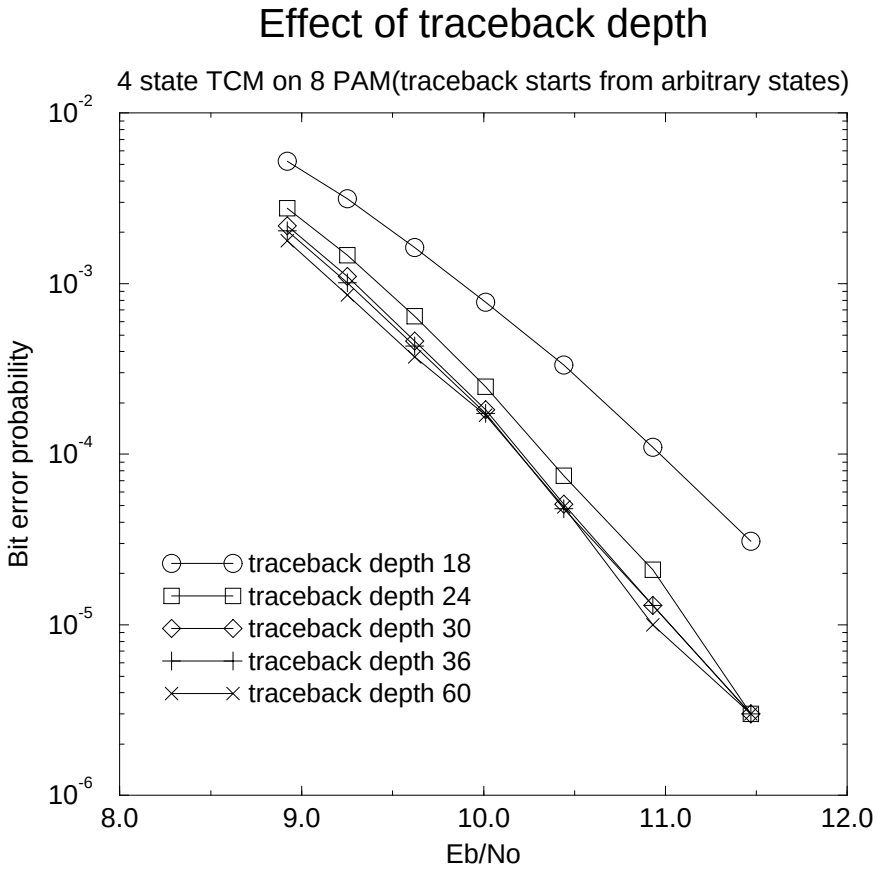


Figure 6.15: Finite traceback depth effects of 4 state TCM on 8 PAM signalling(traceback starts from arbitrary state).

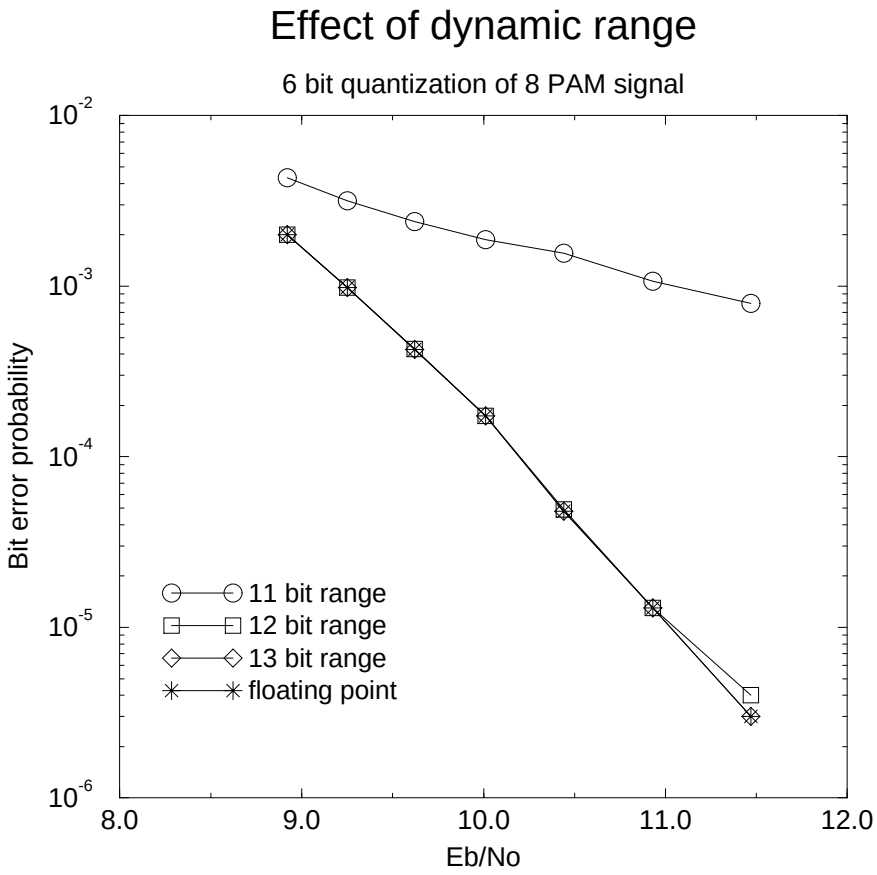


Figure 6.16: Dynamic range effects of 4 state TCM on 8 PAM signalling.

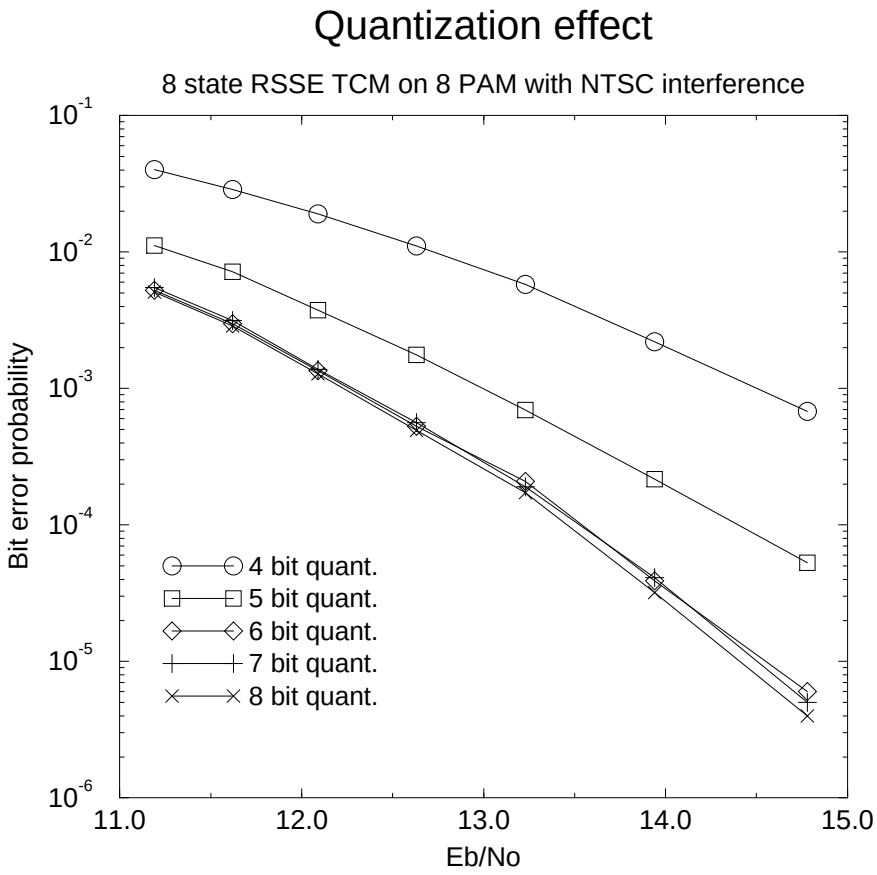


Figure 6.17: Quantization effects of 8 state RSSE TCM with NTSC interference.

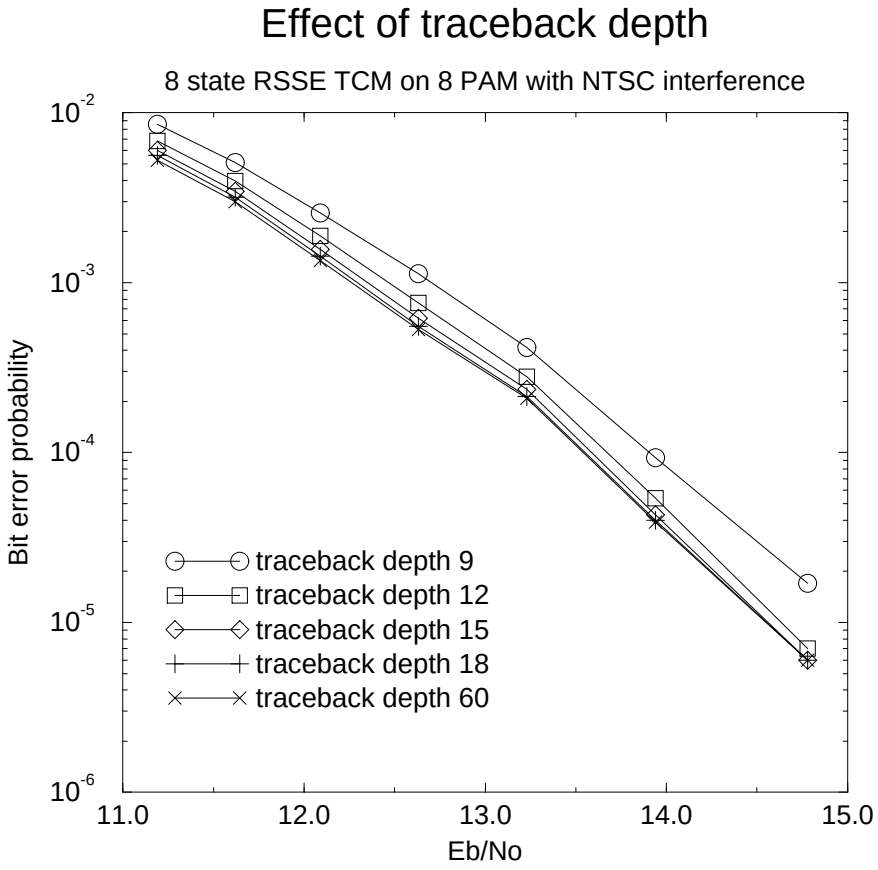


Figure 6.18: Finite traceback depth effects of 8 state RSSE TCM with NTSC interference.

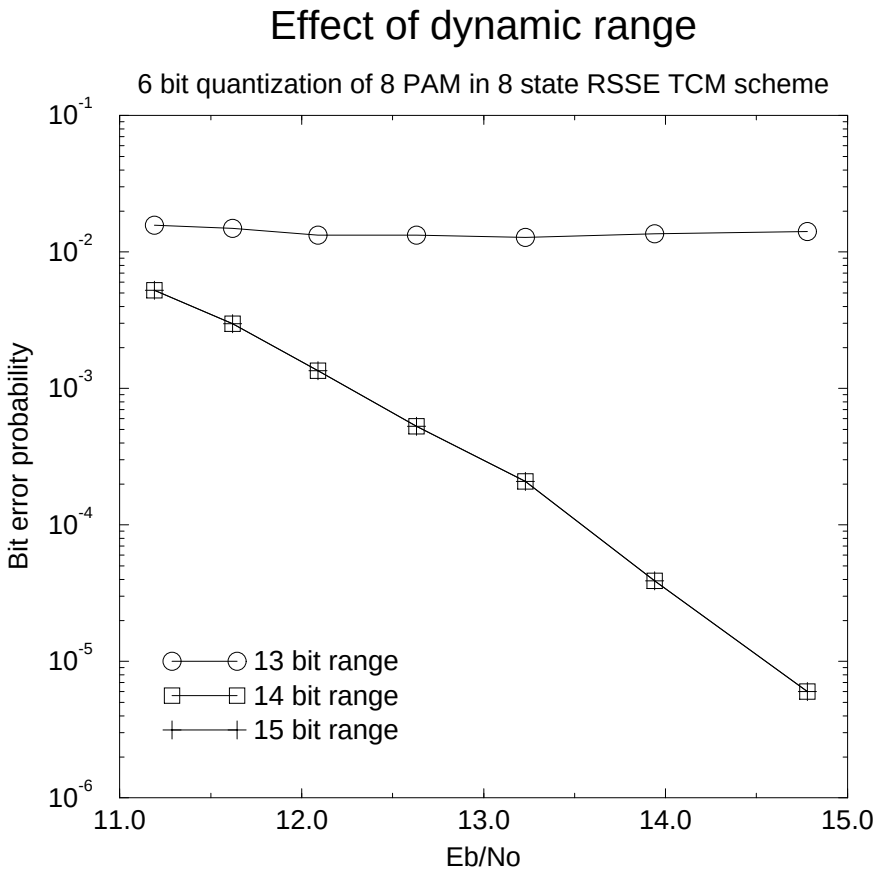


Figure 6.19: Dynamic range effects of 8 state RSSE TCM with NTSC interference.

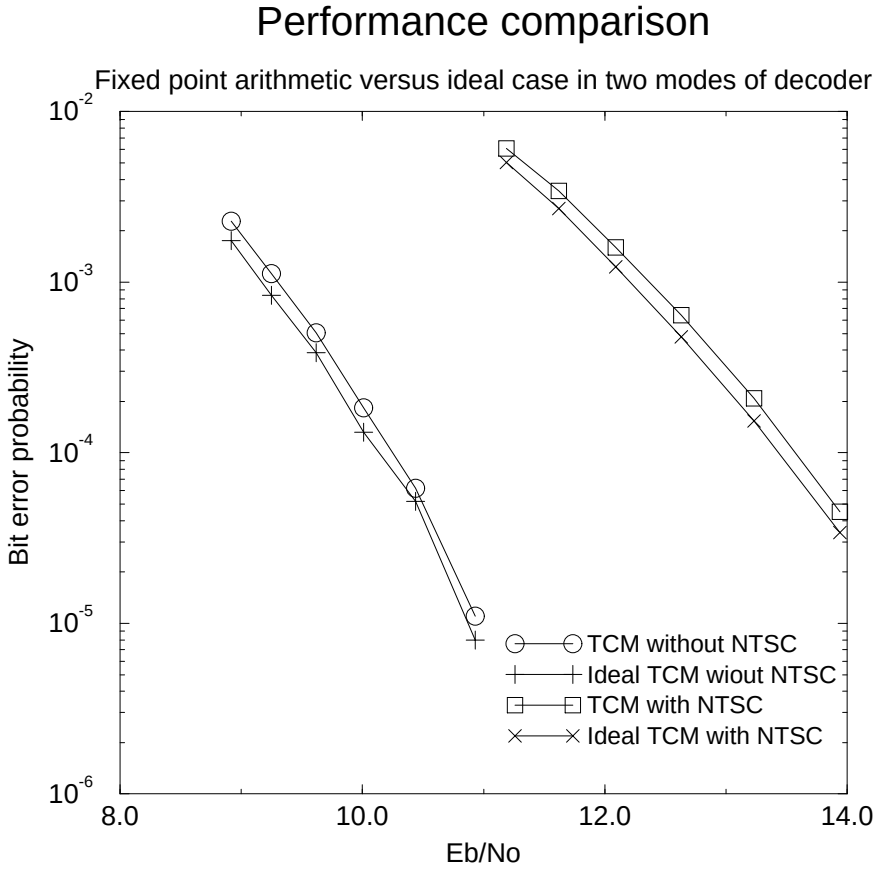


Figure 6.20: Performance comparisons of fixed point TCM implementation and ideal TCM in both NTSC interfered and non-interfered situation.

Chapter 7

Conclusion

We have investigated new channel coding schemes which make convenient use of previous decoder designs as well as methods for variable error protection transmission that extensively re-use a common decoder engine. Punctured convolutional codes (PCC) need one basic decoder for different code rates, reducing the area devoted to the decoder in ASIC implementation. In addition to this, it is less complex than ordinary convolutional codes. Punctured TCM (PTCM) uses PCC as its component and has the usual advantages of PCC. However, in general, use of a punctured structure in the decoder may cause some performance loss due to difficulties in assigning the branch metrics. Furthermore, it is difficult to design rotationally invariant codes.

We have designed PTCM for multilevel QAM and PSK. We found the optimal position for puncturing, generators for the rate-1/2 trellis, branch metrics and coset partitioning in QAM and PSK. In general, whenever the metric can be decomposed into orthogonal components a punctured code can be constructed which apart from negligible boundary effects has no performance loss with respect to the equivalent original TCM scheme.

We provided two types of 90^o RI-PTCM for QAM signalling. 90^o invariance requires non-linear codes when the cosets are two-dimensional. This may be accomplished by using ordinary punctured coding and then swapping state labels, or rearranging the branch labels which decreases the minimum distance. On the other hand, we may use separate PTCM for the in-phase and quadrature components, where the original rate-1/2 code must be 180^o invariant and all punctured versions of the code will be invariant. Thus we may continue to use the the same decoder structure for all members of the family.

Unequal error protection (UEP) codes provide a different error protection on different classes of information. HDTV (High Definition Television) broadcast allows the possibility of offering several grades of service. Customers close to the transmitter could receive the full resolution promised by HDTV while those at a larger range would receive NTSC quality (normal TV quality) images, which can be accomplished by using multi-resolution codes. Multi-resolution code could be a subset of UEP in the sense that NTSC quality information is important and highly protected and the additional information is not very important but can up-grade the quality of image if correctly recovered. In general, a UEP code could be useful in any transmission over unknown channels with variable SNR in the sense that only the receiver knows the channel state and then choose the rate for optimal perceived quality.

Recently a couple of good results on UEP codes were published. Wei has proposed rotationally invariant UEP codes [5] [36]. Unfortunately, in his scheme, the coding gain for the less important is not satisfactory. Calderbank and Seshadri have considered various kinds of UEP codes for different ratios of important to unimportant data [3]. In their schemes, coding gains for the less-important bits are significantly improved, but rotationally invariant code structure Were not mentioned. were not included.

We have proposed a family of multilevel UEP codes based on superimposition for the additive white Gaussian channel. Four way partitioning in a one dimensional lattice combined with a non-uniform signal constellation provides good coding gains to the two data classes with reasonable complexity. Based on the simulation results, our schemes perform better than the previous published papers. Furthermore, we can easily make 90^o rotationally invariant codes by using 180^o rotationally invariant rate-1/2 convolutional code and resolving the in-phase and quadrature-phase power.

Even though the HDTV standard was fixed, specific methods of designing the forward error correction circuits were not described. We have developed the optimum decoding method under the HDTV standard constraints. The receiver has two modes of operation corresponding to the NTSC interference situations. When the receiver is in NTSC-rejection mode, the decoder has to cope with a partial response characteristic and 3 dB noise enhancement due to 1-D filtering in front of the receiver. Those problems were solved by using RSSE and the higher order noise prediction in the TCM decoder. The noise prediction gives us about 3 dB more gain over the performance normally expected from the standard.

As a real implementation issue, we have addressed the questions of the minimum level of quantization and traceback depth for comparable performance with the ideal decoding. Another question is how many bits of dynamic range are required when we are using two's complement arithmetic to avoid metric rescaling. Those three factors were obtained by simulations.

Our main contribution can be summarized as follows. First, we have investigated possible ways of designing various rates of rotationally invariant PTCM and UEP codes by using only rate-1/2 trellis. As a result, the critical components of the decoder will

be re-used for variety of channel coding schemes, which may save the cost, development time and the area devoted to the decoder in ASIC implementation. Second, we have proposed a family of UEP codes which show better performance than the previously proposed schemes with moderate complexity and even with a rotationally invariant structure. These results have variety of applications (for example, speech and video broadcasting and transmission over unknown channels, etc). Third, under the HDTV standard constraints, we have designed the optimum decoding method which shows about 3 dB more gain over the performance normally expected.

We still have some interesting open problems. It is desirable to investigate the relations between punctured TCM and rotational invariance in the trellis structure. This may enable a higher degree of decoder components re-use, and might lead to some computational complexity reduction. Alternatively a surprising number of codes can be constructed in a punctured fashion; a fundamental investigation of such properties would be an interesting task. We have considered two level UEP codes and the performance was compared with the simulation results of the previous papers. If we have UEP codes which have more than two level data protections each with variable portion, it would be much easier to manage with the problem of transmission over unknown channels. For a given portion of each class data, it would be an interesting problem, as a performance reference, to obtain the upper bounds on the performance of different class data protection.

Bibliography

[1] J.K. Wolf and E. Zehavi, "P^2 codes: Pragmatic Trellis Codes Utilizing Punctured Convolutional Codes," *IEEE Communications Magazine*, vol. 33, pp. 94–99, February 1995.

[2] F. Chan and D. Haccoun, "High-rate punctured convolutional codes for trellis-coded modulation," in *Proc. IEEE International Symposium on Information Theory*, p. 414, 1993.

[3] A.R. Calderbank and N. Seshadri, "Multi-level codes for unequal error protection," *IEEE transactions on Information Theory*, vol. IT-39, pp. 1234–1248, July 1993.

[4] K. Ranchandran, A. Ortega, K. M. Uz and M. Vetterli, "Multiresolution broadcast for HDTV using joint source-channel coding," *IEEE journal of selected areas on communicaitons*, vol. 11, pp. 6–23, January 1993.

[5] L.F. Wei, "Coded modulation with unequal error protection," *IEEE Transaction on Communications*, vol. 41, pp. 1439–1449, October 1993.

[6] E. Biglieri, D. Divsalar, P. McLane, and M. Simon, *Introduction to Trellis-Coded Modulation with Applications*. Macmillan Publishing Company, 1991.

[7] C.C. Tung and J.N. Livingston, “Analysis and design of rotationally invariant trellis codes,” in *Proc. IEEE GLOBECOM*, pp. 128–132, 1993.

[8] E. Biglieri and M. Elia, “Rotationally invariant trellis codes,” in *Proc. of the Fifth Tirrenia International Workshop on Digital Communications, Tirrenia, Italy*, pp. 55–65, September 1991.

[9] L. Bweg, R. Buz and P.J. Mclane, “Design procedure for optimum or rotationally invariant trellis codes,” in *Proc. IEEE International Conference on Communications*, pp. 607–613, 1990.

[10] L.F. Wei, “Rotationally invariant convolutional channel coding with expanded signal space - Part I: 180^o,” *IEEE journal of selected areas on communicaitons*, vol. SAC-2, pp. 659–671, September 1984.

[11] L.F. Wei, “Rotationally invariant convolutional channel coding with expanded signal space - Part II: Nonlinear codes,” *IEEE journal of selected areas on communicaitons*, vol. SAC-2, pp. 672–686, September 1984.

[12] J. Hagenauer, N. Seshadri and C.W. Sundberg, “The performance of rate-compatible punctured convolutional codes for digital mobile radio,” *IEEE Transaction on Communications*, vol. 38, pp. 966–980, July 1990.

[13] J. Hagenauer, “rate-compatible punctured convolutional codes (RCPC codes) and their applications,” *IEEE Transaction on Communications*, vol. 36, pp. 389–400, April 1988.

[14] Y. Yasuda, K. Kashiki and Y. Hirata, "High-rate punctured convolutional codes for soft decision Viterbi decoding," *IEEE Transaction on Communications*, vol. 32, pp. 315–319, March 1984.

[15] J.B. Cain, G.C. Clark and J.M. Geist, "Punctured convolutional codes of rate $(n-1)/n$ and simplified maximum likelihood decoding ," *IEEE Transaction on Information Theory*, vol. IT-25, pp. 97–100, January 1979.

[16] P. Tjahjadi *et al.*, "VANDA - A CAD system for communication signal processing circuits design," in *in VLSI Signal Processing IV*, pp. 43–52, New York: IEEE Press, 1990.

[17] J. Proakis, *Digital Communications*. McGraw-Hill Book Company, 1989.

[18] A. Viterbi and J. Omura, *Principles of Digital Communicaitons and Coding*. McGraw-Hill Book Company, 1979.

[19] J.B. Anderson and S. Mohan, *Source and Channel Coding - An Algorithmic Approach*. Kluwer Academic Publishers, 1991.

[20] A.R. Calderbank, "Lecture notes on Trellis Coded Modulation."

[21] Hewlett-Packard/General Instrument, "Interactive Television Appliances Request for Proposal Cable TV Forward Error Correction and 64-QAM Modulator ASIC," July 1994.

[22] G.D. Forney, Jr., "Geometrically uniform codes," *IEEE transactions on Information Theory*, vol. 37, pp. 1241–1260, September 1991.

[23] S. Benedetto, M.A. Marsan, G. Albertengo and E. Giachin, "Combined coding and modulation : theory and applications," *IEEE transactions on Information Theory*, vol. IT-34, pp. 223–236, March 1988.

[24] G.D. Forney Jr., "Coset codes Part I: Introduction and geometrical classification," *IEEE Transaction on Information Theory*, vol. IT-34, pp. 1123–1151, September 1988.

[25] G.D. Forney Jr., "Coset codes Part II: Binary lattices and related codes," *IEEE Transaction on Information Theory*, vol. IT-34, pp. 1152–1187, September 1988.

[26] E. Zehavi and J.K. Wolf, "On the performance evaluation of trellis codes," *IEEE Transaction on Information Theory*, vol. IT-33, pp. 196–202, March 1987.

[27] A.R. Calderbank and N.A. Sloane, "New trellis codes based on lattices and cosets," *IEEE transactions on Information Theory*, vol. IT-33, pp. 177–195, March 1987.

[28] G. Ungerboeck, "Trellis-coded modulation with redundant signal sets, part I : Introduction ," *IEEE Communications Magazine*, vol. 25, pp. 5–11, February 1987.

[29] G. Ungerboeck, "Trellis-coded modulation with redundant signal sets, part II : State of the art," *IEEE Communications Magazine*, vol. 25, pp. 12–21, February 1987.

[30] R. Calderbank and J. E. Mazo, "A new description of trellis codes," *IEEE Transaction on Information Theory*, vol. IT-30, pp. 784–791, November 1984.

[31] G. Ungerboeck, "Channel coding with multilevel/phase signals," *IEEE Transaction on Information Theory*, vol. IT-28, pp. 55–67, January 1982.

[32] A. Viterbi, J.K. Wolf, E. Zehavi, R. Padovani, "A Pragmatic Approach to Trellis Coded Modulation," *IEEE Communications Magazine*, vol. 27, pp. 11–19, July 1989.

[33] A.R. Calderbank, "Multilevel codes and multistage decoding," *IEEE Transaction on Communications*, vol. 37, pp. 222–229, March 1989.

[34] G.J. Pottie and D.P. Taylor, "multilevel codes based on partitioning," *IEEE transactions on Information Theory*, vol. IT-35, pp. 87–98, January 1989.

[35] G.D. Forney Jr. and L.F. Wei, "Multidimensional constellations - Part I: Introduction, figure of merit, and generalized cross constellations," *IEEE Journal on Selected Areas in Communications*, vol. 7, pp. 873–892, August 1989.

[36] L.F. Wei, "Trellis-Coded Modulation with Multidimensional Constellations," *IEEE Transaction on Information Theory*, vol. IT-33, pp. 483–501, July 1987.

[37] A.R. Calderbank and N.J.A. Sloane, "An eight dimensional trellis code," *proc. of the IEEE*, vol. 74, pp. 757–759, May 1986.

[38] A.R. Calderbank and N.A. Sloane, "Four-dimensional modulation with an eight-state trellis code," *Bell Syst. tech. Journal*, vol. 64, pp. 1005–1017, May-June 1985.

[39] M. Vedat Eyuboglu, "Detection of Coded Modulation Signals on Linear, Severely Distorted Channels Using Decision-Feedback Noise Prediction with Interleaving," *IEEE Transaction on Communications*, vol. 34, pp. 401–409, April 1988.

[40] M.V. Eyuboglu and S.U. Qureshi, "Reduces-state sequence estimation with set partitioning and decision feedback," *IEEE Transaction on Communications*, vol. 36, pp. 13–20, January 1988.

[41] M.V. Eyuboglu, S.U. Qureshi and M.P. Chen, “Reduces-state sequence estimation for trellis coded modulation on intersymbol interference channels,” in *Proc. IEEE GLOBECOM*, pp. 878–882, 1988.

[42] E. Biglieri and M. Elia, “Multidimensional modulation and coding for bandlimited digital channels,” *IEEE Transaction on Information Theory*, vol. IT-34, pp. 803–809, July 1988.

[43] D. Divsalar, M.K. Simon and J.H. Yuen, “Trellis coding with asymmetric modulations,” *IEEE Transaction on Communications*, vol. 35, February 1987.

[44] C.-E. W. Sundberg, W.C. Wong and R.S. Steele, “Logarithmic PCM weighted QAM transmission over Gaussian and Rayleigh fading channels,” *IEEE Proc., pt. F*, vol. 134, pp. 557–570, October 1987.

[45] A.P. Hekstra, “An Alternative to Metric Rescaling in Viterbi Decoders,” *IEEE Transaction on Communications*, vol. 37, pp. 1220–1222, November 1989.

[46] R. Blahut, *Theory and Practice of Error Control Codes*. Addison-Wesley Publishing Company, 1984.

[47] G.D. Forney Jr., “The Viterbi Algorithm,” *proc. of the IEEE*, vol. 61, pp. 268–278, March 1973.

[48] G.D. Forney Jr., “Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference,” *IEEE Transaction on Information Theory*, vol. IT-18, pp. 363–378, May 1972.

[49] A.J. Viterbi, “Convolutional codes and their performance in communicaitons systems,” *IEEE Transaction on Communications*, vol. COM-19, pp. 751–772, October 1971.

[50] T. Cover, “Broadcast channels,” *IEEE transactions on Information Theory*, vol. IT-18, pp. 2–14, January 1972.

[51] J. Kim and G.J. Pottie, “On punctured trellis coded modulation,” *IEEE transactions on Information Theory*, vol. 42, pp. 627–636, March 1996.

[52] J. Kim and G.J. Pottie, “On punctured trellis coded modulation,” in *Proc. IEEE International Conference on Communications*, 1995.