VARIABLE COMPLEXITY ALGORITHMS AND ADAPTIVE COMPUTATION CONTROL IN VIDEO CODING AND COMMUNICATIONS

by

Wendi Pan

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(ELECTRICAL ENGINEERING)

December 2002

Acknowledgments

My deep gratitude to my advisor, Prof. Antonio Ortega, for his continued support and guidance throughout my Ph.D. research. Thank to Prof. Keith Chugg and Prof. Peter Beerel for their advice and help in the Fano work. Thank to Prof. Jay Kuo and Prof. Ming-Deh Huang for serving in my guidance committee. In addition, I would like to thank STMicroelectronics Inc. for its generous support. The summer spent at the AST Labs with Ibrahim Hajj and Roberto Sannino has been both enjoyable and productive one. I am grateful to Andrea Basso, who gave me the opportunity to work with great minds at AT&T Labs - Research. I would also like to thank my fellow students and colleagues at the Video Communications Lab.

Table of Contents

List of Tables

List of Figures

Abstract

With the current trend towards Internet wireless access, there is an increased interest in portable, hand-held digital communication devices, where power is an important performance metric. If we assume that algorithm complexity is a good approximation of the power consumption, then variable complexity algorithms are beneficial since they enable reduced power modes. This dissertation addresses variable complexity algorithms in three areas:

First, we propose a novel variable complexity algorithm (VCA) for transform coding based on energy thresholds. Since our VCA computes a subset of the transform coefficients, it yields different complexity and rate distortion (RD) performance than standard transform coders. We present analytical results for the optimal bit allocation in this VCA. Experiments show that our VCA can achieve better rate distortion tradeoff at low rates than the popular JPEG techniques.

Second, we propose a new proxy-based framework that is capable of accelerating video decoding operations of portable devices with wireless access ability. We demonstrate that an adaptive proxy can significantly reduce the complexity of the Inverse Discrete Cosine Transform running at the client without violating any applicable constraint on the proxy-client bandwidth.

Third, we solve the problem of computation control for variable complexity Fano decoders. Fano decoders can provide the desirable tradeoff between bit error rate and decoder complexity. However, buffers are required due to large variations in decoding delays. In order to minimize the overall probability of block loss under a finite buffer size constraint, we propose algorithms for controlling the decoding complexity adaptively with changes of buffer occupancy. For memoryless additive white Gaussian noise channels, our control algorithms can reduce the block loss by as much as 40%. For slow flat Rayleigh fading channels corresponding to low / medium degree of mobility, we introduce prediction algorithms based on finite state Markov models that characterize the memory of the channel. Simulations show that block loss can be lowered by an additional 5% as compared to results obtained under the assumption of memoryless channels.

Chapter 1

Introduction

1.1 Motivation

With the merger of the Internet and wireless services, today's world witnesses a host of portable, hand-held digital communication devices that are rapidly evolving in complexity. At the heart of these small devices are microprocessors that run often complex algorithms. In this context, due to the limited battery life, the power consumption has become an important performance metric for microprocessors. In a more general context, power consumption is increasingly more important because

- dissipating heat has a large impact on packaging technology and cost;
- excessive switching currents effect reliability of the system;
- cooling fans create noise and add to the overall cost of the system;
- lower power devices are preferred also for environmental reasons (e.g., Energy Star partnership);
- the national power conservation policies are becoming increasingly common.

Although design decisions made on lower levels such as system, circuit and gate level can affect the overall power consumption, the choice of algorithm at the highest level has the largest impact on the overall system power [5][67]. Thus, our research has been focusing primarily on the algorithmic level. Rather than using the measurement of power consumption directly, we adopt the computational complexity as the metric in our study. Although there does not exist any simple mapping between the complexity of an algorithm and the overall power consumption of a system employing the algorithm, we assume that the complexity can still be used as a good predictor of the power consumption. Complexity can be measured in terms of actual elapsed time, CPU clock cycles or the number of operations consumed by a process.

There are many applications where algorithms that can operate with varying complexity are found to be beneficial. For example, in image/video compression that has to be performed under varying complexity constraints (e.g., with hardware having to operate occasionally in reduced power mode) it is beneficial to design compression algorithms that allow some degree of complexity scalability.

In terms of computational complexity, algorithms can be categorized into two classes, either fixed complexity algorithms (FCA) or variable complexity algorithms (VCA). The complexity of a VCA can vary as a function of the input, e.g., different images result in different amount of computation. The goal of designing a VCA is to achieve lower complexity than an FCA in the average case. While the "worst case" complexity of the VCA may be higher than that of a comparable FCA, a good VCA can be designed to have lower complexity on the average.

1.2 Overview

This dissertation investigates three topics related to variable complexity algorithms. We first describe a novel variable complexity algorithm for transform-based image coding. This algorithm introduces energy thresholding as a way to regulate the complexity cost and it is shown to be an effective transform coding algorithm in image and video compression applications. The second topic focuses on a new variable complexity IDCT framework and associated algorithms in Internet applications where the idea of "more overhead bits for less power consumption" is pursued. This is achieved by making use of a proxy that provides information to the client of a decoder to enable it to run faster. We propose *Variable Complexity* (VC)-IDCT algorithms that are suitable for this proxy/client framework and we demonstrate several ways of reducing the IDCT complexity by using proxy processing. The third topic deals with the "Fano" algorithm, a variable complexity channel decoding algorithm, which is the VCA counterpart to the well known Viterbi decoder. The emphasis is on the design of a good control policy over the complexity of the Fano algorithm in order to achieve a minimal block loss of decoded data blocks.

1.2.1 Variable Complexity Algorithms in Image and Video Compression Applications

In this work, we aim to reduce the computational complexity of transform coding algorithms. The algorithms we study include the discrete Karhunen-Loeve transform (KLT) and the forward discrete cosine transform (DCT) and its inverse transform

(IDCT). DCT/IDCT have been widely used in various international standards such as JPEG [65] for still image coding, MPEG-1[55] and MPEG-2 [31] for video coding and H.261[41]/263 [42] for video conference applications.

Traditional transform image coding algorithms, such as the KLT and the DCT, have fixed computational complexity once the size of the transform matrix is fixed. Variable complexity transform coding algorithms, however, allow the complexity to change as a function of the input. In applications where compression has to be performed under varying complexity constraints (e.g., with hardware having to operate in reduced power mode) it is beneficial to design compression algorithms that allow some degree of complexity scalability.

In this dissertation, we explore variable complexity algorithms for transform coding. In Chapter 2, we proposed a novel variable complexity algorithm (VCA), which uses energy thresholds to determine the number of coefficients to be computed for each input. In our VCA, after each coefficient is computed, the energy of the coefficients already computed is tested against a threshold, to determine at what stage the input signal is sufficiently well represented by the subset of transform coefficients. If the threshold is reached or exceeded, the computation terminates. Since our VCA is based on the computation of a subset of the transform coefficients, it yields both different computation time and rate distortion (RD) performance than standard transform coders. While some existing techniques can provide similar trade-off in complexity, rate and distortion, the novelty of our VCA lies in its much finer granularity in the complexity level selection.

Although the computational cost of energy testing may not be a major concern in hardware implementation, it could become a significant overhead to the overall complexity of a software system. Therefore, we intend to statistically analyze the overall savings in complexity, e.g., upper/lower bounds of the complexity, when the cost of testing is taken into account. Furthermore, the VCA can be viewed as a pattern classifier in that it classifies the source (coefficient sample space) into several disjoint cases. We are interested in characterizing the class probability, and the class variances.

The above statistical study leads us to the relation between the rate and distortion of our VCA, and enables the analysis of an optimal bit allocation for the transform coefficients after the VCA. Generally speaking, the smaller the energy threshold, the faster the computation can finish, and yet, the more distortion the transform will result in. For the general N-point VCA, to the best of our knowledge the rate/distortion relation has not been studied before. Finally, we present experimental results to demonstrate that our VCA approaches can be utilized to improve the rate/distortion tradeoff of the popular JPEG techniques.

1.2.2 Proxy Techniques for IDCT Complexity Reduction

There is a trend of increased heterogeneity in today's Internet, both in terms of access devices (from high end PCs, hand held PDA's to cell phones) and of access bandwidth (from cable modems, DSL to 28.8k dial-up). Application adaptation can be provided by proxies, which are placed between clients and servers to perform computation and services on behalf of the clients [6, 19, 29, 56]. In the boundary between the wired and wireless worlds, proxy-based processing has been recognized as an efficient approach to

provide client-specific adaptation to improve the overall performance. For the portable video decoding devices with wireless access ability, fast, low-power decoding is desirable.

In the widely used image/video coding standards such as JPEG, H.263, MPEG-1, MPEG-2, etc, the inverse discrete cosine transform (IDCT) has long been recognized to consume a significant portion of the total decoding time/power budget. Lowering the complexity of IDCT can help reduce the overall power consumption of the client where image/video decoding is performed. Therefore, we choose IDCT in our study of the proxy-based algorithm acceleration techniques.

Many *variable complexity* IDCT algorithms have been designed that exploit the DCT coefficients quantized to zeros at the encoder, especially when the image quality is low. Significant speed-up is reported to be realizable by exploiting the sparseness of the IDCT blocks[89]. These IDCT algorithms skip computing the zeros in the input blocks, and thus achieve a variable degree of complexity reduction, in accordance with the zero patterns of the IDCT input.

However, inherent to almost any VCA is the overhead of having to spend a certain amount of computation in some form of testing of the input. In the work of Chapter 2, the proposed VCA requires energy test to determine when to terminate the computation. Likewise, many existing VC-IDCT algorithms rely on the classification of zero patterns in the IDCT input block in order to speed up the computation. Nevertheless, if the zero testing is performed at the client, its overhead could become nontrivial. When the zero testing takes an excessive amount of complexity, the benefit achieved by computing fewer inputs will be diminished. In a client-proxy setting, however, the task

of zero testing can be moved from the client to the proxy. Therefore, the central idea is for the proxy to send side information bits to the client to convey the zero pattern information about the IDCT inputs. In turn, the client decodes this side information and can speed up the IDCT operation by getting rid of the zero testing steps.

In order to achieve our goal, several issues need to be addressed. First, it is desirable to choose a VC-IDCT algorithm that is conducive to the "decoupling" of the zero testing and real IDCT operations. Second, once a IDCT algorithm is decided upon, we need to choose the granularity of the zero testing scheme. As expected, there is an interesting trade-off between the amount of overhead bits sent through the channel linking the proxy and the client and the IDCT speedup factor achievable at the proxy. If the testing of zeros is overly coarse-grained, the resulting side information tends to be low, but our ability to speed up the IDCT operations is limited. On the other hand, if the testing is too fine-grained, the number of different zero pattern classes may lead to an excessive increase of the amount of data to be transmitted from proxy to client. Note also that the decoding of the side information at the client becomes an overhead, hence an excessive amount of side information also affects adversely the IDCT speedup achievable at the client. Finally, we need to find a proper way to code the side information for transmission between proxy and client.

In Chapter 3, we propose a novel proxy-based framework that is capable of accelerating the IDCT operations performed at the client. We introduce a proxy-specific IDCT algorithm that is not only suitable for the proxy framework, but has fine-grained complexity scalability as well. We demonstrate through simulations the effectiveness

of the proxy in providing a useful trade-off between client-proxy bandwidth and IDCT complexity at the client. We also show that through adaptation, an active proxy can further reduce the client complexity without violating the bandwidth constraint. Note that while proxy-based processing is not a new technique, to the best of our knowledge, no similar work in IDCT computation acceleration based on proxy adaptation has been reported.

1.2.3 Computation Control of Fano Decoders

The third topic we study are the adaptive computation control policies for the Fano convolutional decoders.

Convolutional codes have gained widespread applications in today's wireless communication systems such as GSM and CDMA [71][80]. A convolutional code is generated by passing the information sequence to be transmitted through a linear finite-state shift register. In general, the shift register consists of K (k-bit) stages and n linear algebraic function generators. In the decoding of a block code for a memoryless channel, we compute the distances (Hamming distance for hard-decision decoding and Euclidean distance for soft-decision decoding) between the received code word and the 2^k possible transmitted code words. Then we select the code word that is closest in distance to the received code word.

The Viterbi algorithm [84] is the best known decoding algorithm that logically explores in parallel every possible user data sequence. It encodes and compares each one against the received sequence and picks the closest match. For this reason, it is a *maximum likelihood* decoder [66]. The major problem with the Viterbi algorithm

is the computation burden and the storage requirement which makes it impractical for convolutional codes with large constraint length. Another disadvantage is that it is inherently a fixed complexity algorithm, namely, its complexity cannot be made scalable to the channel conditions. In decoding the convolutional codes, metrics are defined for branches and paths in a trellis, as a measure of the closeness of the received sequence to the coded sequence. For example, when the channel condition is reasonably good, the best path has a much lower metric than the others and so in that sense the Viterbi algorithm is an "overkill" because it explores all survivors. Note that, however, the Viterbi algorithm is good for hardware because its computation is very regular. In portable mobile communications, it is often desirable to trade off BER with decoder complexity/power consumption, especially if it is possible to make changes dynamically as the channel conditions change.

Prior to the discovery of the optimum algorithm by Viterbi, a number of other algorithms had been proposed for decoding convolutional codes. The earliest was the sequential decoding algorithm originally proposed by Wozencraft [90], and subsequently modified by Fano [16]. The Fano sequential decoding algorithm searches for the most probable path through the tree or trellis by examining one path at a time. The increment added to the metric along each branch is proportional to the probability of the received signal for the branch. By comparing the metric of a candidate path with a moving threshold, the Fano algorithm detects and discards incorrect paths. Note that sometimes it can discard good paths too. Therefore, Fano sequential decoders are variable complexity convolutional decoders, which have the desirable property of

operating with very low computation at high SNR. However, the variable complexity nature of the Fano algorithm means that buffers are required for the Fano decoder due to large variations in processing delays. Also the Fano algorithm is not guaranteed to be optimal.

In this work, however, our emphasis is on the adaptive control of the speed of a VC algorithm. Our goal is to devise a control mechanism, so that the speed at which the VCA operates can change with the status of the system. For example, in [39], a computation control mechanism was proposed within the MPEG-2 and H.263 encoding framework that allows the encoding algorithm to adapt to the available computational resources, while achieving the best possible reproduction quality. In this dissertation, we investigate the buffer control problem for the variable complexity *Fano* algorithm, where we seek to obtain a good control parameter for the computation speed of the Fano decoder, and design a good control policy of the decoding speed, so that the decoder can run faster if the decoder buffer becomes full, and slower when the buffer comes closer to underflow.

Here we are faced with an interesting tradeoff — if the buffer becomes full, blocks have to be dropped. Alternatively, if the decoder runs faster in order to avoid buffer overflow, the "coarser" decoding will cause more blocks to be decoded in error, so that they are also considered lost. If the decoder is operating slowly, relative to the incoming block rates, the decoded blocks tend to contain few bit errors. However, the buffer will tend to fill up. Thus our goal is to design a buffer control policy that can minimize the average number of lost blocks under the constraint of limited buffer size.

One of the difficulties of this research is that the complexity of the Fano decoder varies randomly. Although there has been extensive research on buffer control techniques in the source coding literature, to the best of our knowledge, no solution has been proposed to this specific computation/buffer control problem under consideration.

In Chapter 4, we formulate the computation control problem and show that the threshold increment (Δ) is a good control parameter for regulating the decoding speed of the Fano algorithm. We provide our solutions to the Δ control problem for two types of channels:

- memoryless AWGN channels. Based on the joint distribution of decoding complexity and *Bit Error Rate* (BER), we obtain a control policy that can always choose the optimal Fano decoder parameter (Δ), at each decoding stage, to minimize block loss. Simulations demonstrate that this control policy is superior to all other conventional control schemes.

- slow flat Rayleigh fading channels. We use finite state Markov models to characterize the memory of the channel. We find that the Fano metrics are good for the estimation of channel states. We develop prediction algorithms based on the Markov models and the joint conditional distribution of decoding complexity and BER. We demonstrate that block loss can be further lowered when we take into account channel memory that corresponds to low / medium degree of mobility as compared to the results obtained under the memoryless channel assumption.

1.3 Contributions of the Research

The major contributions of this dissertation are listed below:

1. Variable Complexity Transform Coding Algorithms

    - We propose a new variable complexity algorithm (VCA), which uses energy thresholds to determine the number of coefficients to be computed for each input.
    - We show that our VCA has a higher degree of scalability, and it is preferable to other alternatives such as a pruned transform, where the same number of coefficients is computed for the whole image.
    - We perform a thorough theoretical analysis of the complexity, rate, distortion of our VCA. For the dimension two case, we find the closed-form relations describing the variance changes in two classes. For a generic N-point VCA transform, we obtain expressions for the average complexity, as well as rate/distortion relations. In addition, rate-distortion-complexity relations are also empirically obtained.
    - We derive the formula for the optimal bit allocations among different coefficients in different classes as a result of our VCA, and present an interpretation of the result in relation to the conventional expression for an N-point transform.

- We demonstrate that our VCA can also increase the compression performance as we take advantage of the energy classification that is needed for VCA operation and design quantizers that match each class.
- We apply the VCA to eight-point KLT and 8×8 DCT in the JPEG framework and experiments show that our VCA approach is superior in rate distortion performance at low rates as compared to the standard transform coding techniques.
- Part of this work has been published in [60].

2. Proxy Based Approaches for IDCT Acceleration

- We propose a novel proxy-based framework that is capable of accelerating the IDCT operations performed at the client.
- We introduce a proxy-specific IDCT algorithm that is not only suitable for the proxy framework, but has fine-grained complexity scalability as well. We demonstrate the effectiveness of the proxy by two examples. Both examples provide the trade-off between the client-proxy bandwidth increment and the IDCT complexity reduction at the client.
- The first example is a simple proxy that is assigned the task of IDCT block classification, which is performed at the client along with the IDCT operation conventionally. Experimental results clearly show the complexity advantage of this method.

- The second example is a more active proxy. Adaptation to the image statistics is introduced. Simulations of the optimization process based on the average complexity criteria show that the client complexity can be further reduced, while maintaining a low side information overhead.
- In the optimization process for the active proxy process, a new greedy method is proposed, which has remarkably lower complexity than the brute-force exhaustive search method, but achieves the same searching performance as shown in the simulations.
- Differential coding is found to be more efficient for the side information transmitted from the proxy to the client in order to describe the class indices of the zero patterns. The entropy results for such overhead information are also presented.
- Part of this work has been published in [62].

3. Computation Control of Variable Complexity Fano Decoders

- We show that Fano decoding algorithm is a variable complexity algorithm and the threshold increment Δ is good control parameter for the useful trade-off between decoding complexity and the Bit Error Rate (BER) of a decoder block. It is shown that faster decoding means higher bit error rate and vice versa.
- Since the Fano decoder has inherent non-deterministic processing delay, which necessitates buffering of data before and after the decoder in practical

real-time decoding systems. We propose the general framework of the Fano decoders with buffer.

- We formulate the buffer control problem as one that seeks to minimize the overall probability of block loss, subject to a finite buffer size constraint. The overall probability of block loss is comprised of two terms. The first term characterizes loss due to excessive bit errors, and the second term describes the decoder buffer overflow.
- We study the joint distribution of decoding complexity and BER and presented the result, based on which we proposed an efficient control policy.
- We develop a simple yet powerful reverse mapping technique that can map the buffer fullness status to a family of Fano decoder speed thresholds, which in turn, are used to define the probability mass on which the optimization operates.
- We present the solutions to the computation control problems over both memoryless and fading channels.
- For memoryless, AWGN channels, we propose a control policy that always selects the optimal Fano decoder parameter (Δ), at each decoding stage, in order to minimize block loss. The optimization process reduces to a simple real-time table lookup algorithm.
- Comparison is made against some popular adaptive rate control techniques, for example, linear, piecewise linear, sigmoidal, logarithmic and exponential.

Simulation results demonstrate the superior performance of our proposed control algorithm.

- For slow flat Rayleigh fading channels, we find that Fano metrics are good for the estimation of channel states, which leads to simple yet effective finite state Markov models to characterize the fading channel.
- We introduce the Kullback - Leibler distance for the partitioning of the Fano metrics into states.
- We develop one-block ahead and two-block ahead prediction algorithms based on the Markov models and the joint conditional distribution of decoding complexity and BER. We show that block loss can be further lowered by up to 5% than the results obtained when assuming the channel is memoryless.
- Part of this work has been published in [61].

1.4 Outline of the Dissertation

The dissertation is organized as follows:

In Chapter 2, we address the proposed variable complexity algorithms in complexity-scalable transform coding. We provide the analysis of several aspects of the performance of VCA approach in Section 2.2. We present the expressions for the complexity, the probabilities and variances of each of the classes generated by the VCA. A formula for the optimal bit allocation for VCA coefficients is also derived and its meaning is discussed. Next, Section 2.3 presents the experimental results of two-point, eight-point

and 8×8 VCA. At the end of the chapter, the rate/distortion performance of our VCA is compared against that of the JPEG standards.

In Chapter 3, we address the proposed proxy-based framework and its associated techniques for IDCT acceleration. Section 3.2 presents a proxy-client framework that is capable of providing a useful trade-off between the client complexity and the client-proxy bandwidth. A proxy-specific IDCT algorithm is proposed in Section 3.3 and its complexity is discussed. Next, Section 3.4 presents a case where the IDCT operation is accelerated when the block classification task is shifted from the client to the proxy. Section 3.5 further demonstrates the effectiveness of the proxy framework by showing that an active proxy can further lower the client's complexity as adaptation to the image statistics is introduced. The new greedy search approach is illustrated and its complexity and performance are discussed. The chapter concludes in Section 3.6 with a mention of our future work.

The adaptive computation control problem is addressed in Chapter 4. In Section 4.2, we discuss the Δ control in the buffer framework. Next, the relations between the buffer occupancy, complexity and block errors are established in Section 4.3. Section 4.4 presents the solution to the Δ control problem for memoryless, AWGN channels. We then describe a simple table lookup algorithm, followed by its simulation results. Section 4.5 describes the finite-state Markov modeling of the flat, slow fading channel and the calculation of the Fano metric tables for the decoding over fading channels. Section 4.6 extends the table lookup algorithm to cope with channels with memory.

Prediction algorithms are introduced to exploit the memory property of the consecutive blocks. Simulation Results are then presented.

Chapter 2

Complexity-scalable Transform Coding using Variable Complexity Algorithms

2.1 Introduction

The need for compression algorithms that support some sort of complexity scalability is likely to increase as multimedia information is increasingly handled by portable devices where the ability to operate under low power conditions is a prime requirement. In this chapter we study techniques that enable scalable operation of block transforms used in standard algorithms such as JPEG and MPEG.

Traditional transform image coding algorithms, such as the discrete Karhunen-Loeve transform (KLT) and the discrete cosine transform (DCT), have fixed computational complexity once the size of of the transform matrix is decided upon. Variable complexity transform coding algorithms, however, allow the complexity to change as a function of the input. In this case the relevant performance measure is the *average complexity* for a given input source.

In [27], for a Gaussian first-order autoregressive source with correlation coefficient $\rho = -0.9$, it was reported that the KLT is superior to the DCT in terms of computational complexity/distortion relation when block size is smaller than 64. This study indicates that the KLT may be an efficient alternative to other transforms, even if it lacks an input independent fast algorithm such as those available for the DCT.

The KLT possesses the following desirable properties [70]:

- It completely de-correlates the signal in the transform domain.
- It concentrates the most variance (energy) in the fewest number of transform coefficients.

A practical implementation of the KLT involves estimating the auto-covariance matrix of the data sequence, its diagonalization, and the construction of the basis vectors.

The main drawbacks of KLT are: input dependence and lack of fast algorithms. However, energy compaction makes it attractive. Take the zero-mean, Gaussian first-order autoregressive source as an example: we have the auto-covariance matrix $[A]_{ij} = \rho^{|i-j|}$, where $i, j = 0, 1, ..., N-1$. If we take the correlation coefficient $\rho = 0.9$, and $N = 8$ for an eight-point KLT, then the contribution of each KLT coefficient towards the total energy can be found in Figure 2.1.

It can been concluded that by computing only the first two coefficients, over 90% of the energy can be kept on average. If we desire to alleviate the distortion caused by the insufficient energy captured, we may choose to compute more coefficients – computing up to the fifth coefficients can already preserve about 97% of the total energy. Moreover,

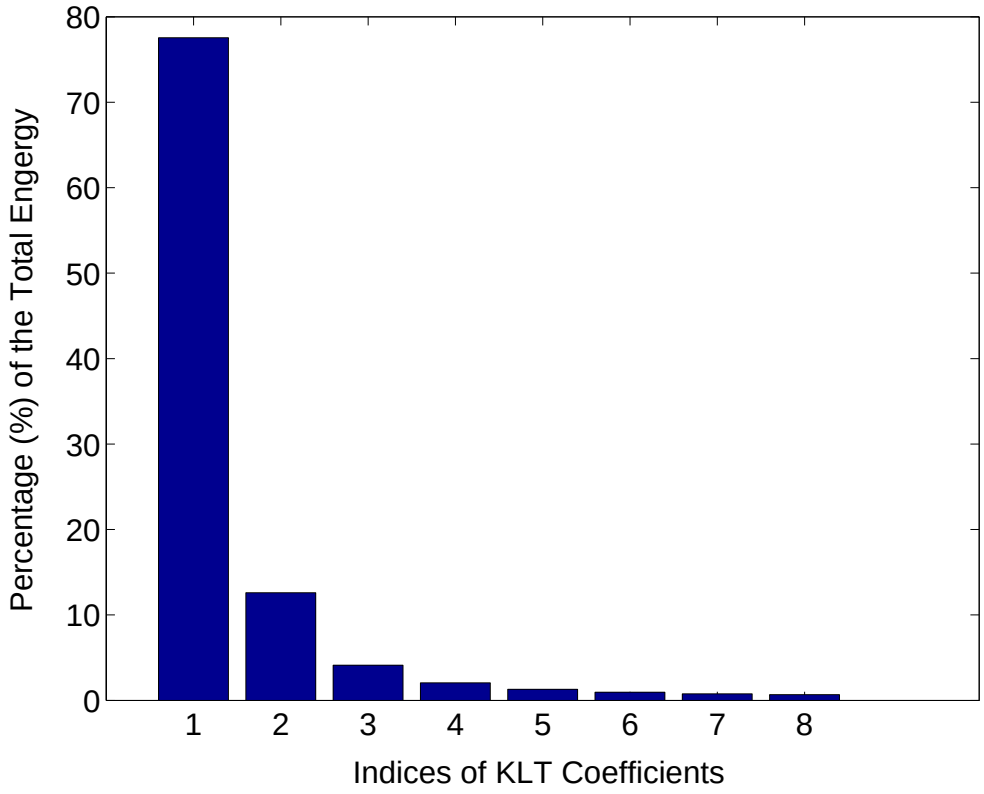


Figure 2.1: The distribution of energy among the 8-point KLT coefficients with first order AR source ($\rho = 0.9$).

the above numbers are average numbers, which means that for some vectors one KLT output may concentrate almost all energy. This suggests that if one could compute only those coefficients needed to capture a certain percentage of the signal energy, the complexity could be significantly reduced.

Motivated by such observations, in this chapter, we propose a VCA which uses energy thresholds to provide a trade-off in complexity, rate and distortion (RD). Since our approach is based on computation of a subset of the transform coefficients, our VCA approach yields both different computation time and RD performance than standard transform coders. In [49], variable complexity is achieved by classifying the DCT inputs

and pruning the DCT computation accordingly. Similarly, in [17], an input dependent inverse discrete wavelet transform algorithm was proposed that takes advantage of the large number of zero coefficients after quantization. In both works, however, no trade-off in complexity, rate and distortion is available. Other complexity-reduction schemes such as [63], use thresholds to select which DCT coefficients to compute, choosing, for example, whether to compute DC only, or compute 2×2, or 4×4, or all 64 coefficients. Our approach provides a much finer degree of granularity than [63] and other related techniques.

We will provide examples based on both the KLT and the DCT. Our objective is two-fold. First, we aim to demonstrate that lower overall complexity, with finer degree of granularity in the complexity level selection is possible with VCA approaches. Second, we demonstrate that the energy thresholds used to achieve variable complexity can also be used to improve the RD performance of the coders.

Our VCA transform coding is illustrated in Figure 2.2, where an eight-point transform is used as an example. Once the k-th coefficient Y_k is computed, a test (denoted by the circles in Figure 2.2) is performed so as to determine whether $\sum_{i=1}^{k} Y_i^2 \geq (T \cdot E)$, where T is the energy threshold, ranging from 0 to 1, and $E = \sum_{i=1}^{8} Y_i^2$, is the source signal energy. If the test passes (i.e. the first k coefficients have captured sufficient energy), then the algorithm stops, leading to a reduced computation for this input; otherwise, it proceeds to calculating the next coefficient. In the succeeding coding operation, all the coefficients that were computed are quantized, whereas those not computed are simply set to zero, with overhead being used by the encoder to indicate

what coefficients are being sent. Note that this is equivalent to the end-of-block (EOB) code that is used in traditional transform coding techniques. Thus in our approach we generate and quantize a variable number of coefficients per input.

This chapter is organized as follows. In Section 2.2 we provide an analysis of various aspects of the performance of VCA approach, including the complexity, the probabilities and variances of each of the classes generated and the rate distortion performance. Section 2.3 presents the experimental results.

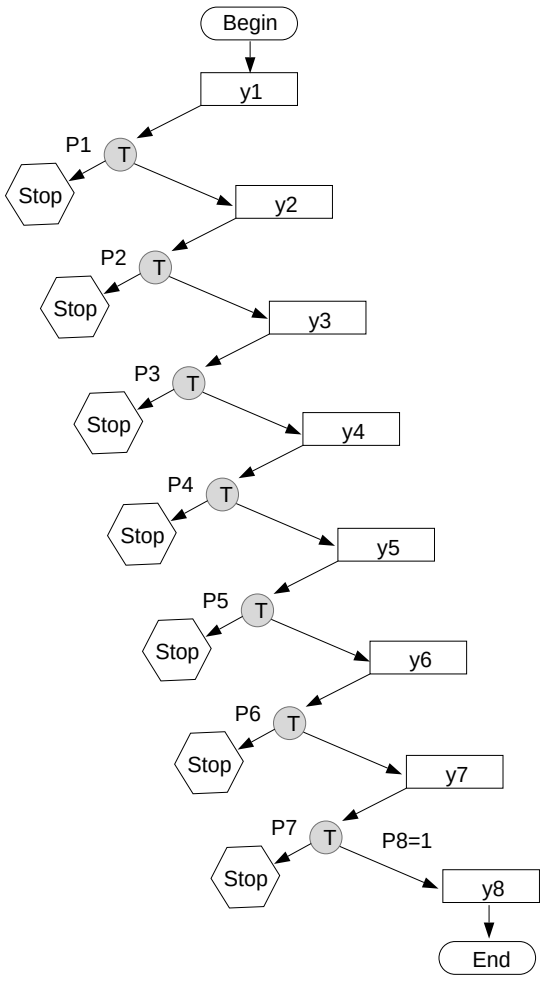


Figure 2.2: Flowchart of the VCA algorithm to compute a size-8 block transform.

2.2 Analysis

In this section we first provide an expression of the average complexity as a function of the probability of each class. We also provide closed-form expressions for these probabilities in a simple case. We then provide the pdf's of data in each class. Finally we combine these results to produce the rate, distortion and complexity characteristic.

2.2.1 Complexity

In Figure 2.2, let p_i represent the probability of the first i coefficients containing sufficient energy to pass the test, i.e., $p_i = \text{Prob.}\left[\sum_{k=1}^{i} Y_k^2 \geq (T \cdot E)\right]$. Obviously, the more coefficients are computed, the more likely our VCA will stop after testing. In general, we thus have

$$p_1 \leq p_2 \leq \ldots \leq p_8 = 1. \tag{2.1}$$

It follows that the probability of the computation terminating after Y_i is

$$P_i = p_i - p_{i-1}. \tag{2.2}$$

The average complexity can be expressed as:

$$\begin{aligned} C_{avg} &= p_1[C(Y_1) + H] + (p_2 - p_1)[C(Y_1) + C(Y_2) + 2H] \\ &\quad + (p_3 - p_2)[C(Y_1) + C(Y_2) + C(Y_3) + 3H] + \cdots \\ &\quad + (p_7 - p_6)[C(Y_1) + C(Y_2) + \cdots + C(Y_7) + 7H] \\ &\quad + (1 - p_7)[C(Y_1) + C(Y_2) + \cdots + C(Y_8) + 7H] + F, \end{aligned} \tag{2.3}$$

which can be reduced to

$$C_{avg} = \sum_{i=1}^{8} C(Y_i) + 7H + F - \left\{ \sum_{i=1}^{6} p_i \cdot [C(Y_{i+1}) + H] + p_7 \cdot C(Y_8) \right\}, \tag{2.4}$$

where $C(Y_i)$ denotes the complexity of computing individual coefficient Y_i (for example computing each coefficient requires 7 adds and 8 multiplies in the KLT case), and H is the cost of testing after computing each coefficient Y_i. F represents the one-time cost of computing $T \cdot E = T \cdot \sum_{i=1}^{N} X_i^2$, the input signal energy scaled by the threshold T.

Note that for non-VCA, the complexity is $C_b = \sum_{i=1}^{8} Y_i$. Hence the complexity of VCA differs from that of non-VCA by an amount S as

$$S = \sum_{i=1}^{6} p_i \cdot [C(Y_{i+1}) + H] + p_7 \cdot C(Y_8) - 7H - F. \tag{2.5}$$

By Eq. (2.1), we have $p_1 \leq p_i \quad (\forall i > 1)$, hence

$$\begin{aligned} S \geq S_l &= \sum_{i=1}^{6} p_1 \cdot [C(Y_{i+1}) + H] + p_1 \cdot C(Y_8) - 7H - F \\ &= p_1 \cdot \sum_{i=2}^{8} C(Y_i) - (7 - 6p_1)H - F. \end{aligned} \tag{2.6}$$

S_l can be approximated as

$$S_l \approx p_1 \cdot C_b - (7 - 6p_1)H - F. \tag{2.7}$$

Similarly, we have

$$S \leq S_u = p_7 \cdot C_b - (7 - 6p_7)H - F. \tag{2.8}$$

And the difference can be obtained as

$$S_u - S_l = (p_7 - p_1)(C_b + 6H). \tag{2.9}$$

If $S > 0$, then the VCA introduces a savings in computation cost with respect to its fixed complexity algorithm counterpart. Consider S_l in Eq. (2.7), if H is fixed, a larger p_1 will lead to a greater reduction in complexity. This implies that transforms with high energy compaction capability are desirable since they can increase the probability of test-and-stop after the first coefficient is computed. On the other hand, it can be seen from (2.4) and (2.7) that the cost of testing H is also an important factor in the average complexity of the VCA since it offsets the gain introduced by the VCA. Consider two transforms: First, the KLT has no structure since it depends on the auto covariance matrix of the input signal. Thus our VCA is useful within a KLT framework as it allows to reduce greatly its otherwise complex N^2 operations. The overhead of testing does not offset too much the complexity reduction gained by computing fewer coefficients. For the DCT, however, there are numerous fast algorithms, many of which require only $O(NlogN)$ operations [70]. Although a VCA can still contribute to a reduction of computation, the relative overhead of the testing due to the VCA will be higher than in the KLT case. Therefore, the testing budget tends to be tight for

variable complexity DCT. Some fast testing methods such as using absolute values rather than square(-multiplication) operations for finding the energy can be considered [26, 49, 63]. Moreover, if we use a larger energy threshold, it is more efficient to do the energy test only after several coefficients have been computed so that the overall testing overhead is reduced. Note that although the computational cost of testing could become a significant overhead to the overall complexity of a software system, it may not be a major concern in hardware implementation.

2.2.2 Class Probabilities

The VCA can be viewed as a pattern classifier in the sense that it classifies the source (coefficient sample space) into several disjoint cases. The classification criterion is the energy test controlled by the threshold use. For instance, Figure 2.3 depicts the three partitions of the sample space classified by VCA.

For a closed-form solution, we consider the 2-D scenario: two zero-mean, independent Gaussian random variables Y_1 and Y_2, with variances σ_1^2 and σ_2^2 respectively. Note that Y_1 and Y_2 are the transform coefficients of any general VCA, which can be KLT, DCT or other orthogonal transforms. The probability P of successful test_and_stop by using energy threshold T corresponds to the mass of the shaded region in Figure 2.4.

It can be shown that (see Appendix A.1)

$$P = Prob\left\{\frac{Y_1^2}{Y_1^2+Y_2^2} \geq T\right\} = \frac{2}{\pi}\arctan\left(\frac{\sigma_1}{\sigma_2}\sqrt{\frac{1-T}{T}}\right) = \frac{2}{\pi}\arctan(K\cdot r), \qquad (2.10)$$

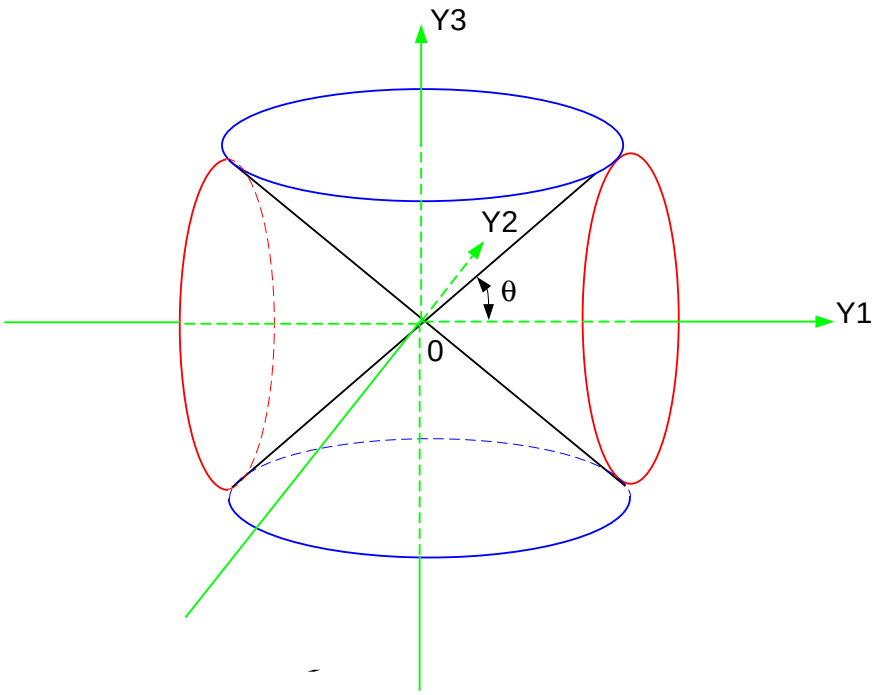


Figure 2.3: Partitioning of the coefficient 3-dimensional sample space by a 3-point VCA. For example, class I: $\left(\frac{Y_1^2}{Y_1^2+Y_2^2+Y_3^2} \geq T\right)$ corresponds to the region enclosed by two horizontal cones. This is the class of inputs such that most of the energy is concentrated in coefficient Y_1.

where $K = \sqrt{\frac{1}{T} - 1} = \tan\Phi$, and $r = \frac{\sigma_1}{\sigma_2}$, is the ratio of standard deviations. In the context of transform coding, r is an indicator of the energy compaction capability of the transform used.

As shown in Figure 2.5, as expected, P decreases monotonically with increasing threshold T. If we fix T and increase r, then samples are more likely to fall into class 1. The reason is that a larger r means that $Y1$ contains more energy than $Y2$ on average and thus the probability of passing the energy test becomes higher. Figure 2.6 shows the complexity savings achieved by the VCA. If the savings drop below zero, it indicates that the VCA does not bring any gain in terms of complexity. This is because for

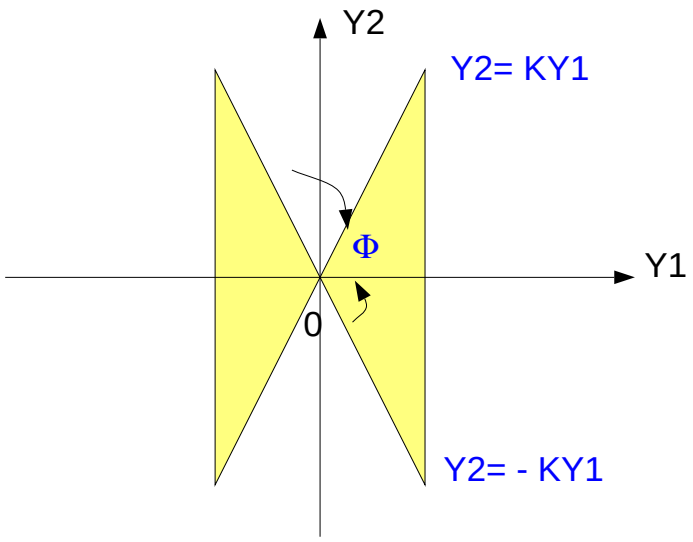


Figure 2.4: Partitioning of the coefficient sample space by the VCA. The shaded region contains all the inputs such that the first coefficient Y_1 contains a percentage of the energy above the threshold.

those thresholds the complexity of testing is not compensated by savings in coefficient calculation, since with high probability both coefficients have to be computed.

As a concrete example of achievable complexity reduction, our simulation shows that 33% and 73% of running time can be saved for eight-point DCT and KLT respectively, when our VCA is used with threshold $T = 0.9$. Note that such transforms are not exact so the choice of a good threshold will depend on the quantization level in use, e.g., for coarse quantization, $T = 0.9$ may be acceptable.

2.2.3 Variances within each Class

The energy threshold T controls the number of coefficients computed and thus the overall complexity. At the same time, the choice of T also affects both rate and distortion, given that the test determines which transform coefficients are quantized. For any

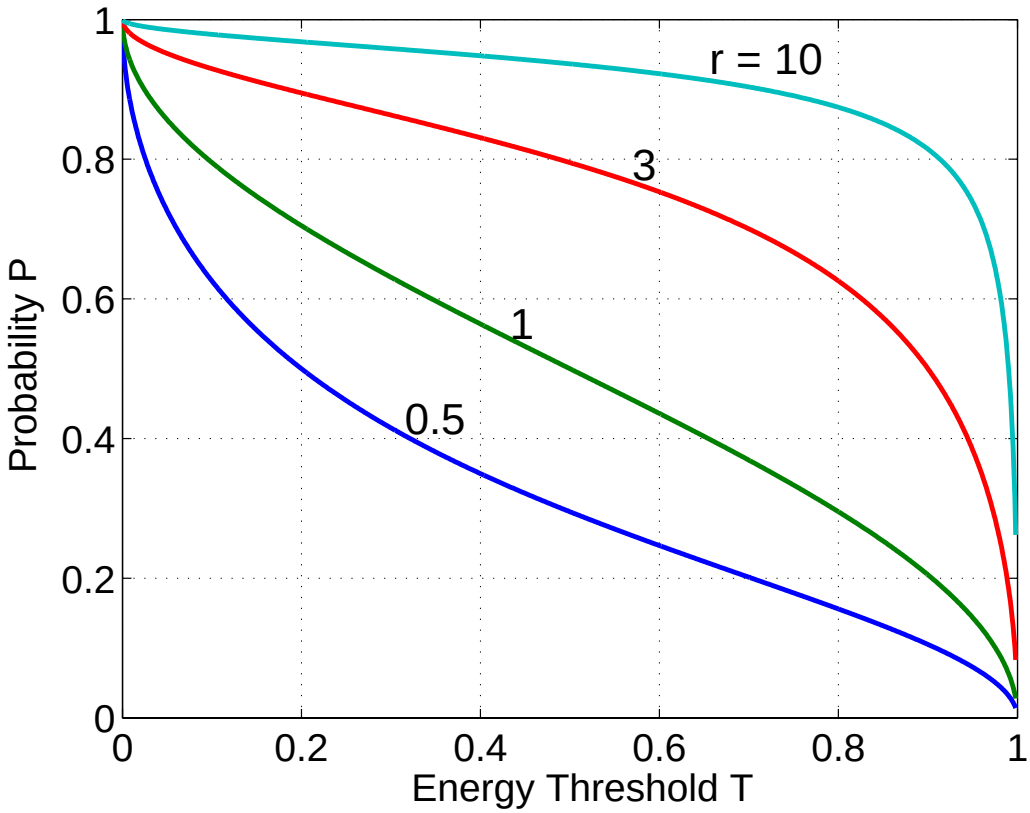


Figure 2.5: P as a function of r and T.

given input vector of dimension N and a particular choice of T, there will be N classes, depending on how many coefficients are computed. For each class k, $(k = 1, 2, ..., N)$, k coefficients are computed and quantized, and the remaining $(N - k)$ are simply set to zero and not quantized. In practice, overhead will be used to inform the decoder that $(N - k)$ coefficients were not encoded, although the overhead may be in the form of End-of-Block (EOB).

We assume that the probability of class k occurring is P_k. We can then represent the variances of the N coefficients *conditioned on* k, that is σ_{ki}^2 for $i = 1, \ldots, N$. See also Table 2.1. It is important to note that the variance will not be the same for each k,

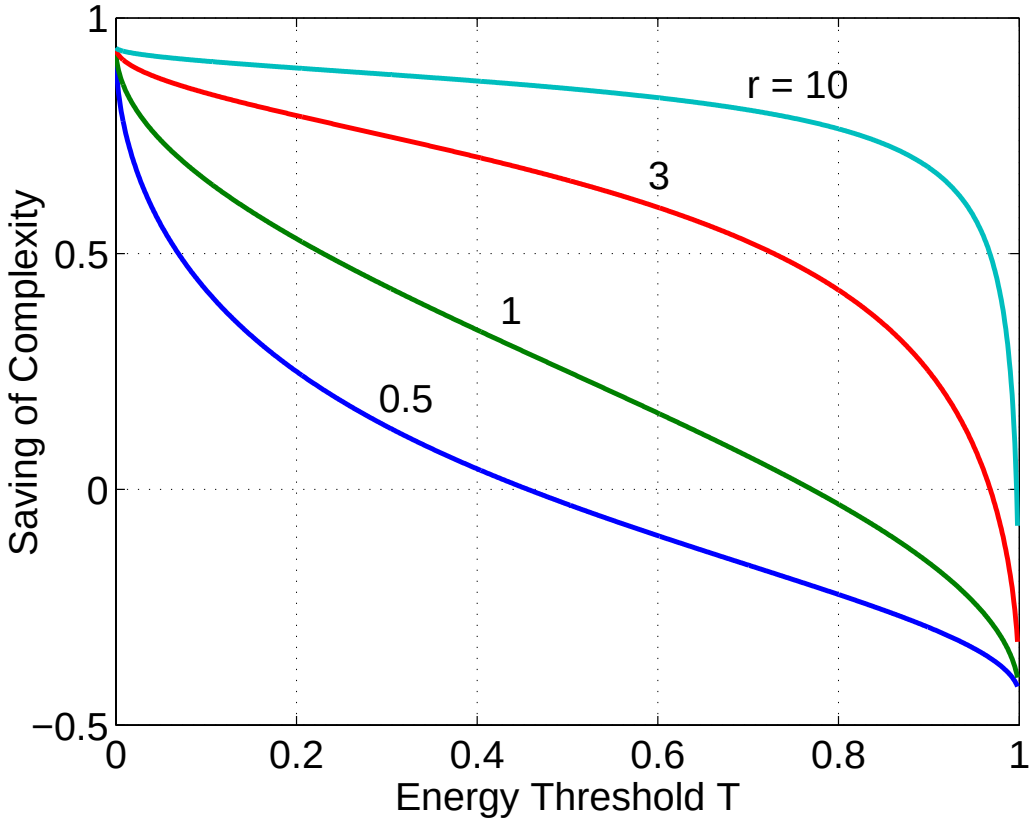


Figure 2.6: Complexity reduction as a function of r and T.

since through testing we are separating several different scenarios. For example, for a given T, if k is small, i.e., a few coefficients contain most of the energy, then σ_{ki}^2 can be expected to be relatively large for small i, and relatively small for large i, as compared to the original variances. This is because the vectors included in this class are such that the energy test is passed and therefore more energy will be concentrated in the lower index coefficients. Thus, in this example, if σ_i^2 is the original variance of the coefficient (when using the standard, non-VCA, transform), then we expect $\sigma_{ki}^2 \geq \sigma_i^2$ for $i < k$ and $\sigma_{ki}^2 \leq \sigma_i^2$ for $i > k$.

Table 2.1: Classes in an N-point VCA.

Class	**Probability**	**Var of Coef's Comp.**	**Var of Coef's Not Comp.**
1	P_1	σ_{11}^2	$\sigma_{12}^2, ..., \sigma_{1N}^2$
2	P_2	$\sigma_{21}^2, \sigma_{22}^2$	$\sigma_{23}^2, ..., \sigma_{2N}^2$
k	P_k	$\sigma_{k1}^2, ..., \sigma_{kk}^2$	$\sigma_{k(k+1)}^2, ..., \sigma_{kN}^2$
N	P_N	$\sigma_{N1}^2, ..., \sigma_{NN}^2$	

We take the two-point VCA as an example. By using the previous Gaussian model, we can find a closed-form expression of the variances of two classes in relation to the original source variance as follows:

Assume that $Y_{ij}, i = 1, 2$, is the jth coefficient belonging to class i. The pdf's of Y_{11} and Y_{12} of class I are:

$$f_{Y_{11}}(y) = \frac{2}{P}\left[\left(\frac{1}{\sqrt{2\pi}\sigma_2}\int_0^{K|y|} e^{-\frac{x^2}{2\sigma_2^2}}dx\right)\frac{1}{\sqrt{2\pi}\sigma_1}e^{-\frac{y^2}{2\sigma_1^2}}\right], \tag{2.11}$$

$$f_{Y_{12}}(y) = \frac{2}{P}\left[\left(\frac{1}{\sqrt{2\pi}\sigma_1}\int_{\frac{|y|}{K}}^{\infty} e^{-\frac{x^2}{2\sigma_1^2}}dx\right)\frac{1}{\sqrt{2\pi}\sigma_2}e^{-\frac{y^2}{2\sigma_2^2}}\right], \tag{2.12}$$

where P is the probability of class I occurring, as given in equation (2.10).

The distributions are plotted in Figure 2.7 and Figure 2.8 for two thresholds. It can be observed that class II is a dual case to class I. Y_{11} has a bi-modal distribution instead of the original normal shape. As T is increased, the two peaks of Y_{11} are increasingly separated while Y_{12} shows an increasingly "peaked" distribution. This suggests that the variance of Y_{11} increases and that of Y_{12} decreases as a result.

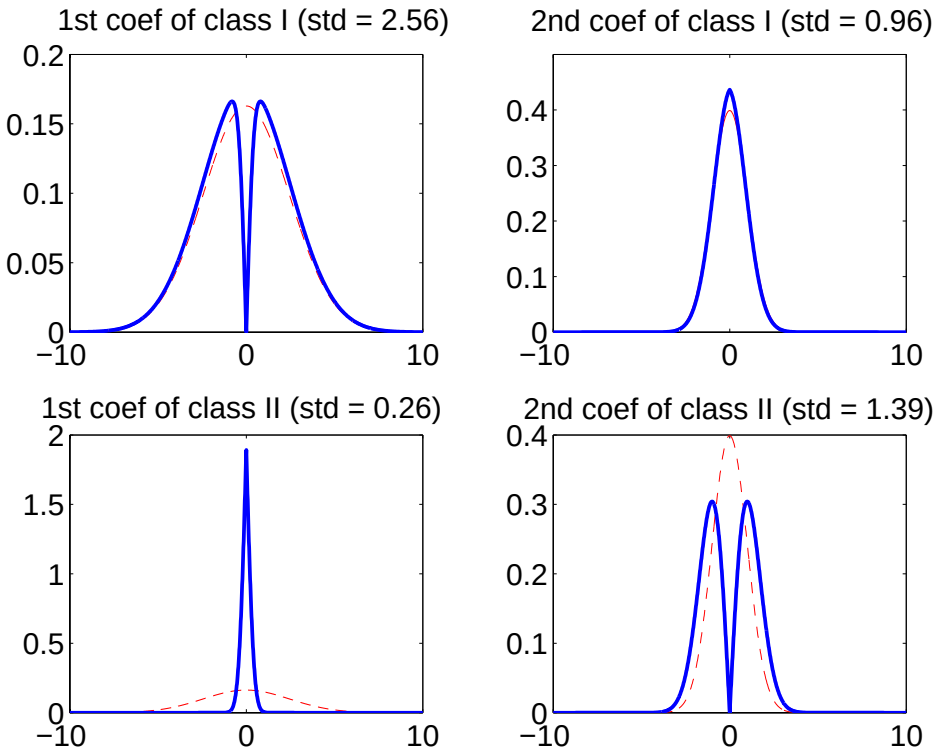


Figure 2.7: Distribution of coefficients in two classes (solid curves) for $T = 0.1$. Dashed curves represent original normal distributions.

It can be shown (in Appendix A) that class variances σ_{ij} are related to the original variances σ_i by two simple adjustment factors, called G and H, as follows:

$$\sigma_{11}^2 = (1+G)\sigma_1^2, \qquad \sigma_{12}^2 = (1-G)\sigma_2^2, \tag{2.13}$$

$$\sigma_{21}^2 = (1-H)\sigma_1^2, \qquad \sigma_{22}^2 = (1+H)\sigma_2^2, \tag{2.14}$$

and

$$G = \frac{2Kr}{P\left(K^2r^2+1\right)\pi} = \left(\frac{1}{P}-1\right)H, \tag{2.15}$$

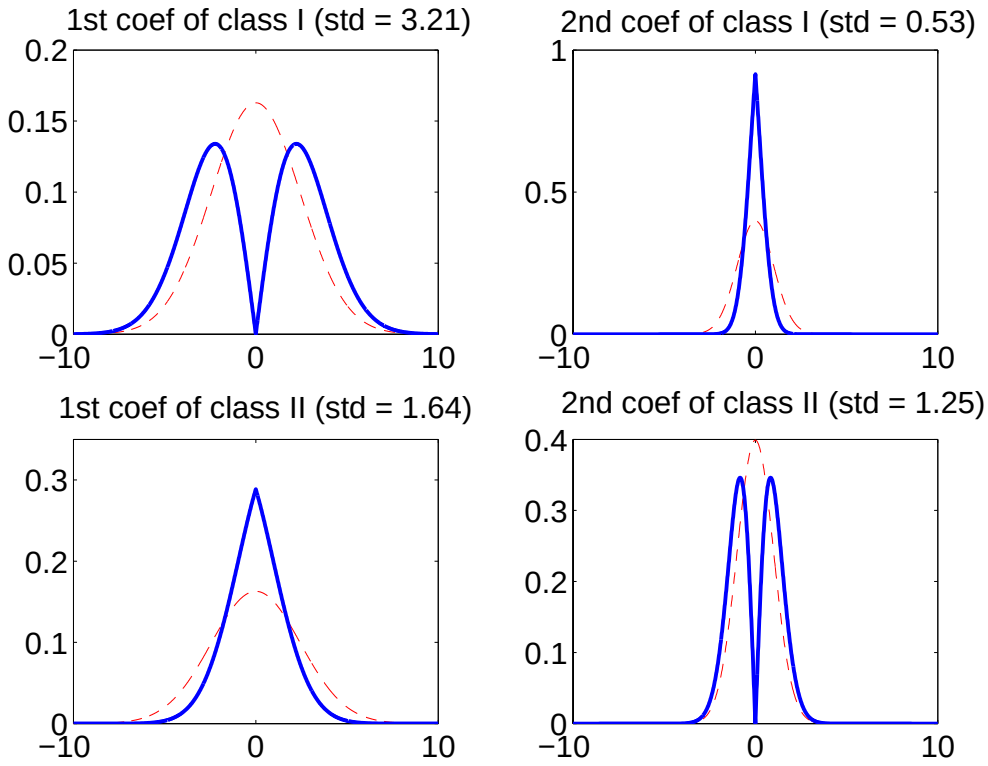


Figure 2.8: Distribution of coefficients in two classes (solid curves) for $T = 0.9$. Dashed curves represent original normal distributions.

where K and P are as defined in Section 2.2.2, and $r = \sigma_1/\sigma_2$.

2.2.4 Rate Distortion Relations

In an N-point VCA, there exist N classes, such that in class k, only the first k coefficients are computed, and the remaining $(N - k)$ coefficients are simply set to zero. We introduce an optimal bit allocation by considering allocation at two levels. At the lower level, we consider coding within each class. There we can use standard bit allocation techniques to determine how to allocate B_k bits among the k coefficients coded

in class k [24]. If coefficients computed are scalar quantized at high resolution, then the optimal distortion for the total number of bits B_k is

$$D_k = kH_k\rho_k^2 2^{-2\frac{B_k}{k}} + \sum_{i=k+1}^{N} \sigma_{ki}^2 \tag{2.16}$$

where, $H_k = \left(\prod_{i=1}^{k} h_{ki}\right)^{\frac{1}{k}}$, $\rho_k^2 = \left(\prod_{i=1}^{k} \sigma_{ki}^2\right)^{\frac{1}{k}}$ are the geometric means of the distortion constants and variances, respectively, which can be obtained from the probability distributions of the coefficients [24].

At a higher level, our goal is to allocate optimally bits to each of the classes, i.e., to find B_k for each class. This allocation has the aim of minimizing the total *average* distortion D under a constraint on total rate B, where:

$$D = \sum_{k=1}^{N} (P_k \cdot D_k), \quad \text{and} \quad B = \sum_{k=1}^{N} (P_k \cdot B_k) \tag{2.17}$$

Note that here the total rate and distortion are averaged with weights corresponding to the probabilities of each class. Conventional Lagrange multiplier techniques can be employed by introducing the cost function J and searching for the rates that minimize it for a given Lagrange multiplier $\lambda \geq 0$,

$$J = B + \lambda \cdot D = \sum_{k=1}^{N} (P_k B_k) + \lambda \cdot \left[\sum_{k=1}^{N} P_k \cdot \left(D_k' + \sum_{i=k+1}^{N} \sigma_{ki}^2\right)\right]. \tag{2.18}$$

where $D_k' = kH_k\rho_k^2 2^{-2\frac{B_k}{k}}$.

We minimize J by taking the N derivatives: $\partial J/\partial B_k = 0$, with $k = 1, ..., N$. It can be shown (see Appendix B) that the optimal bit allocation can be expressed in average number of bits assigned to class k as:

$$B_{k,opt} = \frac{B_k}{k} = \bar{B} + \frac{1}{2}\log_2 \frac{H_k \rho_k^2}{\left[\prod_{i=1}^{N} (H_i \rho_i^2)^{iP_i}\right]^{\frac{1}{P_t}}} \tag{2.19}$$

where $P_t = \sum_{k=1}^{N} kP_k$, is the average number of coefficients computed for all N classes, and $\bar{B} = B/P_t$, is the average number of bits per coefficient. Note that the distortion of (2.16) corresponds to the best bit assignment to the coefficients within a class and has been used in estimating the distortion in (2.19). Thus, this two step procedure (allocation among classes first, then within classes) is also guaranteed to be optimal. Intuitively, (2.19) shows that the allocation to class k exceeds the average allocation $\bar{B}$ if the parameter $H_k \rho_k^2$ is greater than the class *probabilities weighted* geometric average of $H_k \rho_k^2$ for $k = 1, 2, \ldots, N$. Conversely, if $H_k \rho_k^2$ is less than this geometric mean, then the bit allocation is less than the average allocation $\bar{B}$. Note that the main change with respect to standard bit allocation results comes from the fact that the number of coefficients receiving bits varies from class to class. Thus, whereas in the standard geometric mean expression we had a $1/N$ term (assuming N inputs) and equal power for each of the terms, here we use P_t (the average number of inputs) instead of N and iP_i instead of constant weights for each class in the allocation. With this optimal allocation we can obtain the following R/D relations (Appendix B):

$$D = \sigma_t^2 + P_t \cdot \left[2^{-B} \cdot \prod_{k=1}^{N} (H_k \rho_k^2)^{\frac{kP_k}{2}}\right]^{\frac{2}{P_t}}, \tag{2.20}$$

and

$$\sigma_t^2 = \sum_{k=1}^{N} P_k \left(\sum_{i=k+1}^{N} \sigma_{ki}^2 \right). \tag{2.21}$$

In summary, with the usual caveat that these results are only valid at high rates, we can see that the optimal allocation results are the same as for fixed complexity transform coding [24], except that in allocating bits across classes both the source variances *and* the probability of each class are taken into account.

2.3 Experimental Results

2.3.1 Two-point VCA

In our experiments, both the conventional two-point fixed complexity and the variable complexity KLT are applied to adjacent pixels of test image "Lenna". Uniform, midriser quantizers are used. We can vary rate and distortion by changing the quantization cell size. Entropy is measured as an estimate of the overall rate, and an energy threshold is used to control the complexity. Note that the advantage of the VCA approach comes from the fact that in the low rate scenario all the bits can be assigned to the low frequency coefficients whenever they have most of the energy. Because we can test for this condition at the encoder we can in effect use a simple vector quantizer as shown in Figure 2.9, with different quantization characteristics depending on whether the test fails or not. More specifically, as derived in Section 2.2.3, we expect that (i) $\sigma_{11}^2 \geq \sigma_1^2$, $\sigma_{12}^2 \leq \sigma_2^2$ and (ii) $\sigma_{21}^2 \leq \sigma_1^2$, $\sigma_{22}^2 \geq \sigma_2^2$.

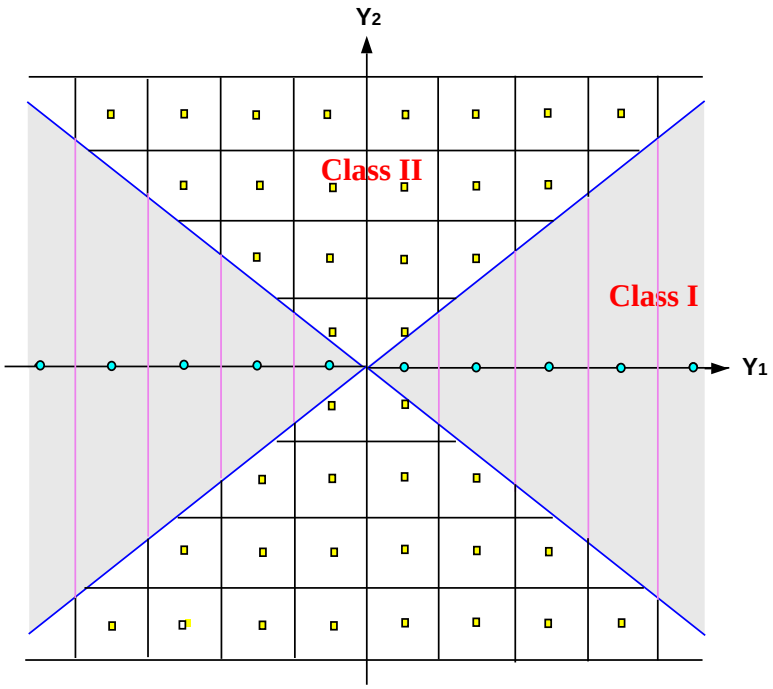


Figure 2.9: Quantization Scheme.

The experimental R/D/C surface is shown in Figure 2.11, where rate denotes the total number of bits assigned to the two KLT coefficients, distortion is the MSE after the VC-KLT. Complexity is represented by the energy threshold T since complexity increases monotonically with T. It can be seen that for a given complexity an increase in rate results in decreased distortion, as was to be expected. Also, the set of coefficients that are computed depends on the complexity. Thus for low threshold, many coefficients are not computed, and are effectively quantized to zero, leading to large distortion.

The curve of optimal distortion for each rate is plotted in Figure 2.10. The rate/distortion curve of the fixed complexity 2-point KLT is shown in Figure 2.11. It is obvious that at low rates (total number of bits lower than 7), VCA does much better in rate/distortion sense. Note that one bit of overhead has been added to the

VCA-KLT approach to notify the decoder of whether one or two coefficients are being transmitted. Clearly, when the rate becomes sufficiently high, the best approach would be to compute both coefficients so that the fixed complexity algorithm should be used.

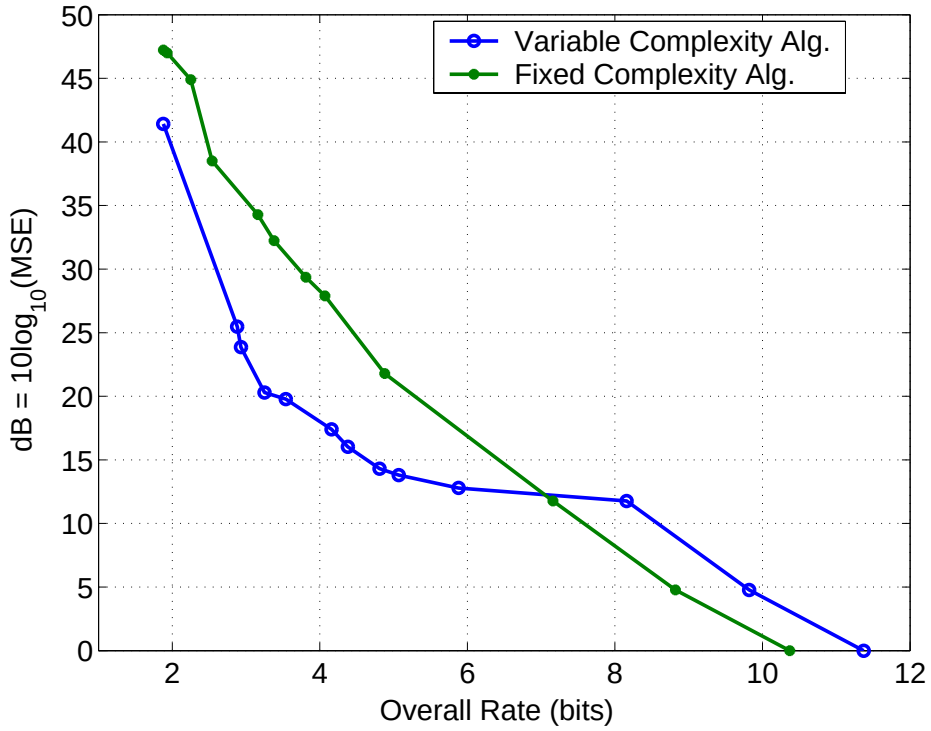


Figure 2.10: Comparison of R/D Relations.

2.3.2 Eight-point VCA

As in the previous experiments, we apply an eight-point variable complexity KLT to adjacent pixels of test image "Lenna".

The rate/distortion curve is shown in Figure 2.12. We test three schemes: (1) fixed complexity KLT (FCA) with one uniform quantizer for all eight coefficients. End_of_block (EOB) is introduced to represent the trailing zeros. EOB is also entropy coded; (2) VC-KLT with separate uniform quantizer for each of the eight classes;

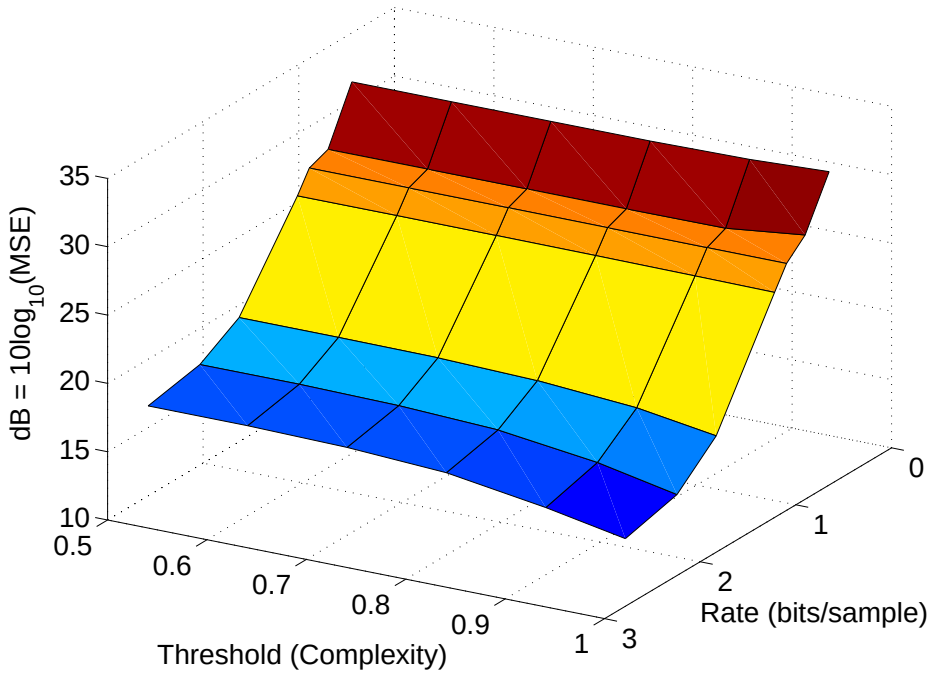


Figure 2.11: R/D/C surface.

(3) Same as (2) except that a separate quantizer is used for each coefficient computed for each class, so that there are a total of 36 such quantizers.

It can be concluded that we can achieve better R/D performance with more quantizers tailored to the characteristics of each coefficient and of each class in the context of the VCA. At rates lower than 1.2bits/sample, the VCA scheme (2) outperforms fixed complexity ones by up to 2dB in MSE. This is consistent with what we have observed in the two-point experiment.

2.3.3 8×8 VCA

We extend our VCA to the 8×8 DCT employed extensively in most existing international standards in image and video compression. VCA divides the 8×8 DCT coefficients into 64 disjoint classes according to the actual number of DCT coefficients

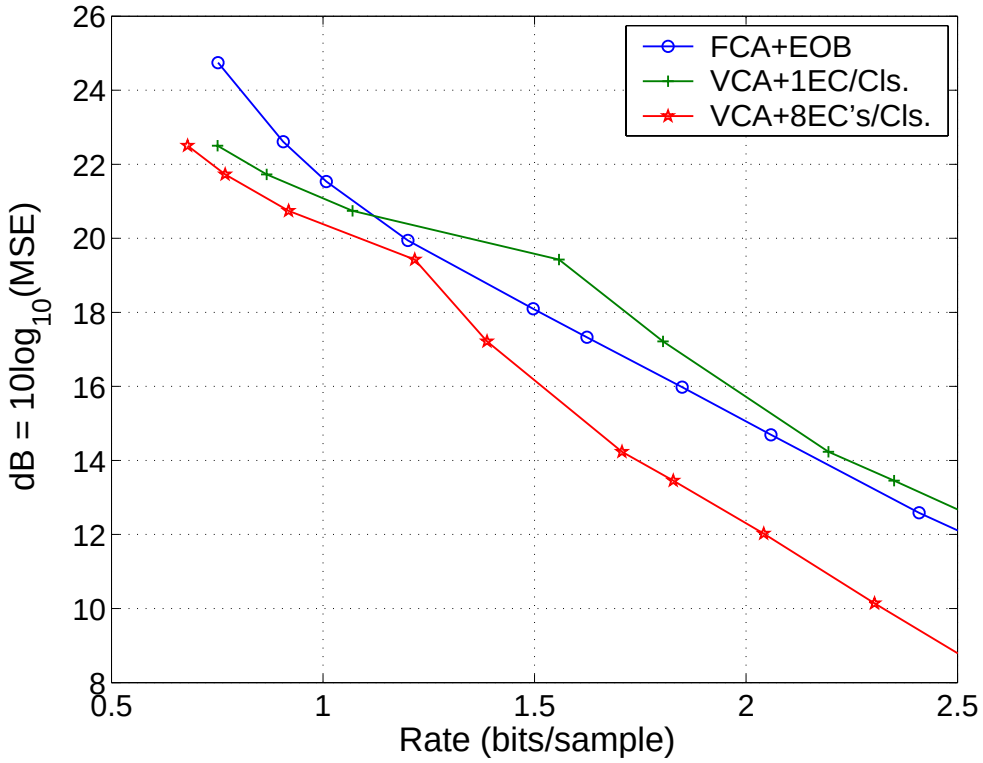


Figure 2.12: R/D curves of FCA and VCA using different quantization schemes on "Lenna".

computed after the energy test. We adopt the zigzag order used in JPEG [65] to index these 64 DCT coefficients in an 8×8 block.

Figure 2.13 illustrates the number of blocks classified into each class by VCA. Obviously, the higher the energy threshold is used in the test, the more blocks are assigned to the classes of higher index. Figure 2.14 illustrates the variances of the DC(1st), second, third and tenth DCT coefficients in each class. A peak in variance always exists which corresponds to the ith coefficient in class i. This feature is caused by the energy test of VCA and can lead to coding gains if properly utilized.

We then apply the VCA to the JPEG baseline algorithm. The approach taken in our preliminary experiment is to perform the energy test of the 8×8 DCT coefficients in

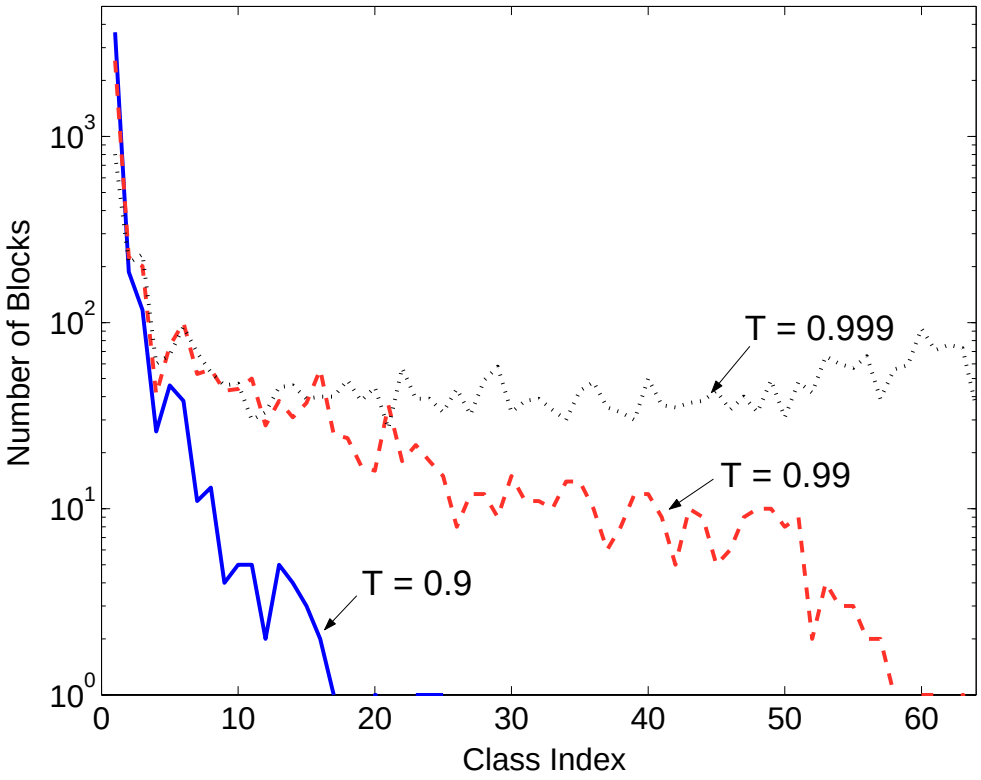


Figure 2.13: Number of blocks falling into each class for 8×8 VC-DCT on image "Peppers".

a block by following the zigzag order. Once the energy threshold is satisfied, we set the remaining coefficients in the block to zero. This procedure can be readily embedded into the JPEG encoder. Note that this is similar to the approach of thresholding [14, 69], with the difference being that here coefficients are not even computed if the energy of the vector is deemed to be sufficient already, whereas in [69] this decision was made after the coefficients were computed.

The rate of JPEG can be varied by changing the scaling factor of the quantization table. We then obtain the rate (bits/sample) and distortion (PSNR) curves with/without the proposed energy test, see Figure 2.15 for the comparison of JPEG versus VCA. It can be seen that VCA provides a better R/D performance for a certain

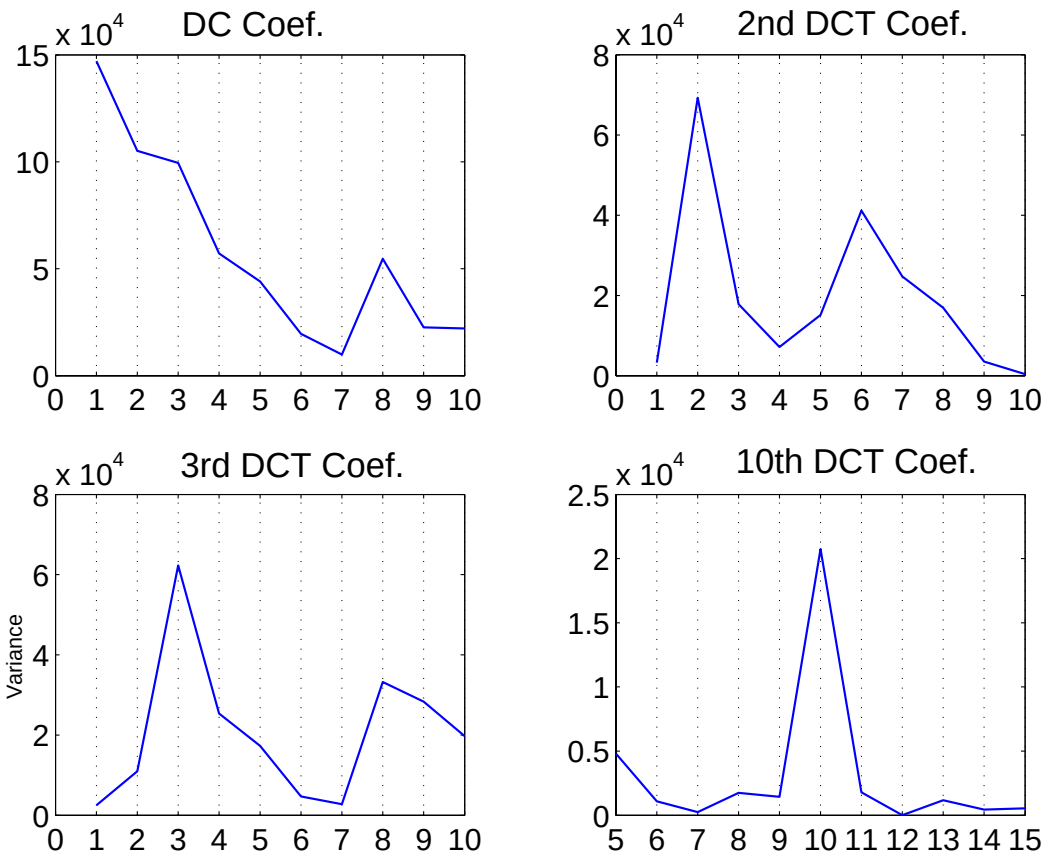


Figure 2.14: Variances of DCT coef's for classes versus the class index.

range of thresholds than that obtained by merely decreasing the scale factor in JPEG. This effect is even more pronounced at rates below 0.2 bits/sample, which agrees well with our findings in two-point and eight-point cases.

Note that the threshold we have been using so far is relative in the sense that it represents the *percentage* of the total vector energy we want to retain. From a rate/distortion tradeoff perspective, however, an *absolute* energy threshold may be more desirable in some situations. We introduce such an absolute threshold scheme in the JPEG experiment:

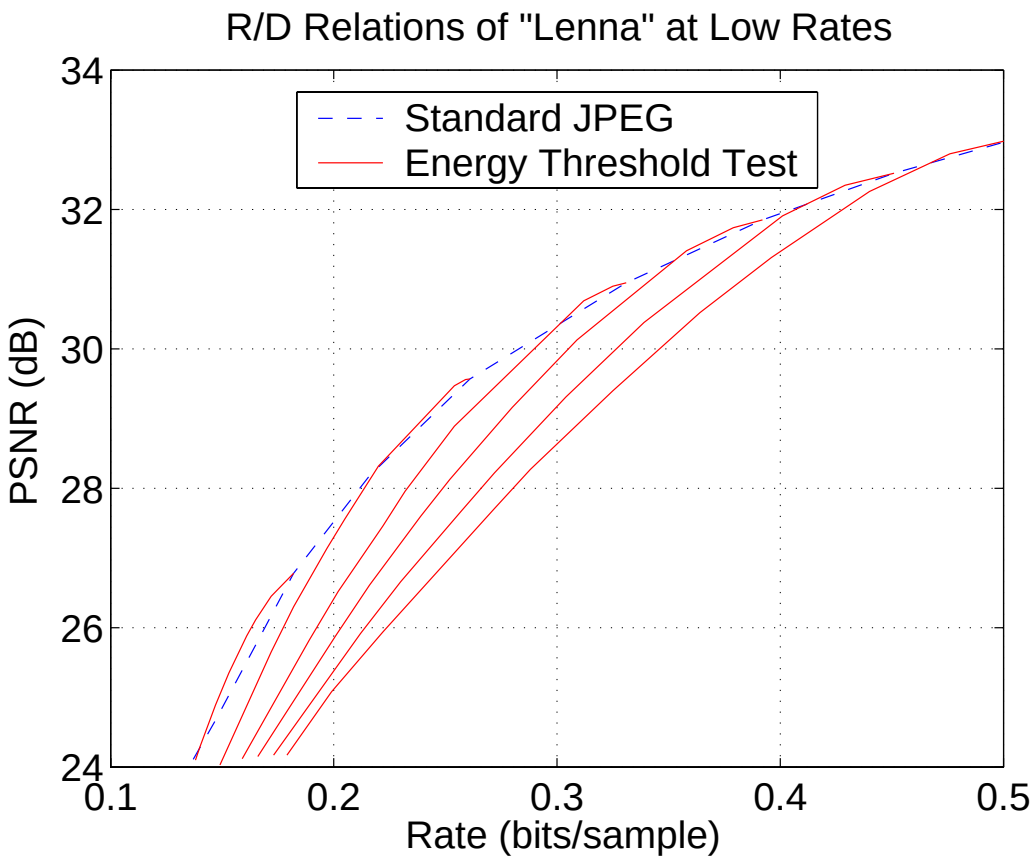


Figure 2.15: Comparison of R/D relations. The dashed curve is for the JPEG baseline, and solid lines stand for the curves obtained by using different energy thresholds when we fix the scale factor of the quantization table.

$$\left(E_{total} - \sum_{i=1}^{k} Y_i^2 \right) \geq T_{abs}. \tag{2.22}$$

In so doing, we keep computing DCT coefficients until the energy of the coefficients not yet computed drops below an absolute threshold. For these residual coefficients not computed, we simply set them to zero, exactly as we have done in the relative threshold approach. The absolute threshold, as a control parameter, represents the distortion (MSE) of the remaining $(N - k)$ coefficients, and thus enables us to have

direct control over the distortion. By contrast, in the relative threshold approach, we can not distinguish DCT coefficient blocks of different total energy. Therefore the distortion caused by the coefficients which are not computed varies from block to block and is not directly related to the threshold itself as in the absolute threshold case. In general, we thus expect the absolute threshold approach to perform much better than the relative one on images of high inter-block energy variations.

The R/D curve when applying the absolute energy threshold in JPEG for the "Lenna" image is shown in Figure 2.16. The curves obtained when using relative thresholds are also shown in the same plot for comparison. It can be seen that absolute threshold approach can always outperform the relative one (up to about 1dB). Consequently, the absolute threshold approach does better than standard JPEG within a wider range of rates than the relative threshold method.

2.4 Conclusions

This chapter presents a variable complexity algorithm (VCA) for transform coding. VCA uses energy thresholds to provide flexible tradeoffs among computational complexity, rate and distortion. We view VCA as a pattern classifier and carry out extensive statistical analysis on the average complexity, as well as rate/distortion relations for the generic N-point VCA. By introducing a standard model in a two point case, we derive closed-form relations which describe the variance variation after classification. In addition, rate-distortion-complexity relations are also empirically obtained. In the eight-point VCA study, we show that by quantizing each coefficient within each class

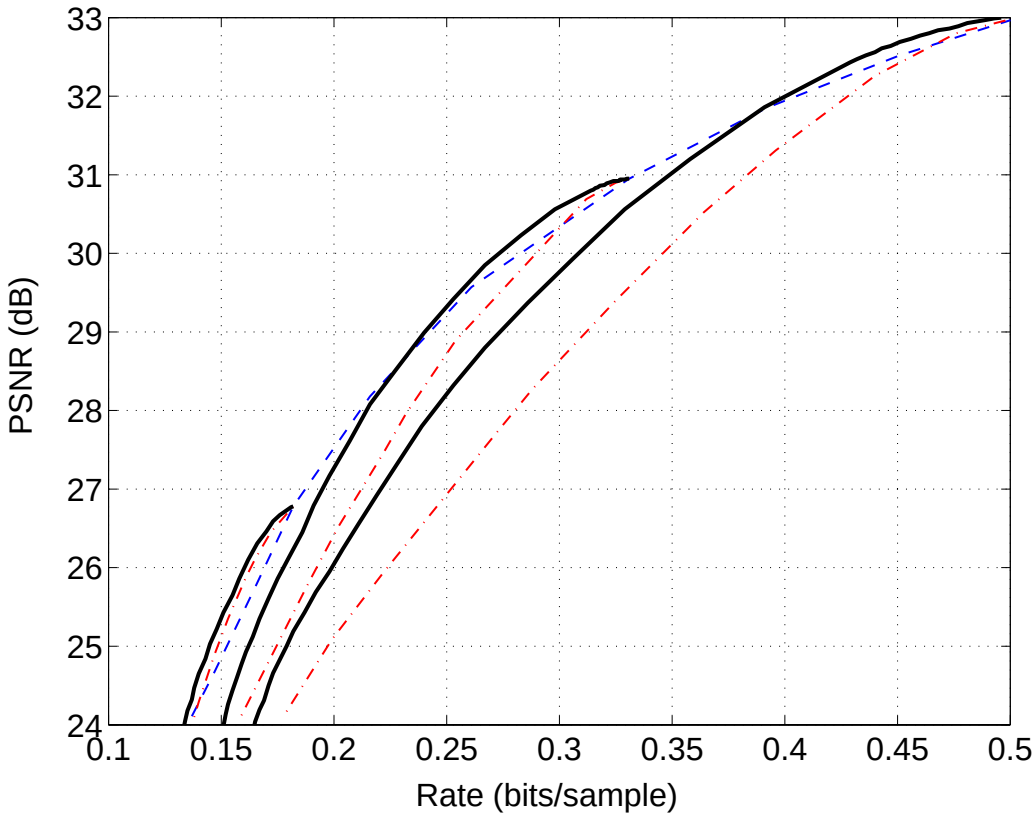


Figure 2.16: Comparison of R/D relations. The dashed curve is for the JPEG baseline, dash-dot curves are for the VCA with relative thresholds, and solid curves stand for the VCA with absolute thresholds.

separately, we can achieve better rate/distortion performance. Finally, we apply VCA to the JPEG baseline algorithm, where both the relative and absolute threshold method are used and compared. Experimental results show that our VCA approaches can provide better rate/distortion tradeoff, especially at low rates than the JPEG techniques.

Our work can be extended to DCT based coding for motion compensated residue images in video coding.

Chapter 3

Proxy-Based Approaches for IDCT Acceleration

3.1 Introduction

Today's Internet sees a trend of increased heterogeneity, both in terms of access devices (from high end PCs, hand held PDA's to cell phones) and of access bandwidth (from cable modems, DSL to 28.8k dial-up). Application adaptation can be provided by proxies, which are placed between clients and servers to perform computation and services on behalf of the clients [6, 19, 29, 56]. Existing examples of the proxy approach include transcoding techniques developed by such companies as Spyglass and IBM. In wire/wireless multimedia communication, the broad potential of the proxy-based processing has been recognized, which is demonstrated by on-going standardization efforts in the industry, for example, the Internet Content Adaptation Protocol(ICAP) [38], as well as the Wireless Application Protocol (WAP) [88].

On the wire/wireless interface, a considerable amount of work featuring the client-proxy-server model has been done. For instance, functions such as data-type specific lossy compression are performed at the proxy so that latency can be reduced in accessing

the web over the CDMA cellular networks [28]. Another example is in mobile video access, where adaptation in joint source and channel coding is applied at the proxy in order to ensure high quality video over error-prone wireless connections [83]. In these applications, the source to be transmitted is either modified or reformatted at the proxy before it is transmitted to the client.

In wireless applications where the power is at a premium for the portable image-decoding clients, it would be desirable for the proxy to be able to communicate with the client and enable fast, low-power decoding based on the client's power needs. Moreover, the proxy can trade-off different resources such as bandwidth, power consumption, latency and memory depending on the client needs or the user preferences.

In this work, we focus on the inverse discrete cosine transform (IDCT), which is used in image/video decoders of various international image and video coding standards such as JPEG, H.263, MPEG-1, MPEG-2, etc [65]. The IDCT is a major component in terms of computation and power consumption. However, since many DCT coefficients are quantized to zero at the encoder, there exist many methods to exploit the sparsely populated, non-zero transform DCT coefficients to speedup the IDCT computations. Hung and Meng [36, 37] identify and count zero values in coefficient matrices in order to select the best IDCT algorithms. Froitzheim and Wolf used reduced IDCT that saves computation for sparse inputs [21]. Nevertheless, it is not aimed at taking advantage of the full sparseness of the input. Lengwehasatit has introduced a tree-structured classification scheme, with the goal of achieving a minimum average complexity after

the zero test overheads has been taken into account [48, 50], but this tree structure was not designed for the proxy-client setting.

In this chapter, we present a novel proxy-based framework for accelerating IDCT computations: the zero patterns of DCT coefficients are tested and IDCT blocks are classified accordingly. The related class information is then sent from the proxy to the client, where IDCT is actually performed. The client can use this information to accelerate the IDCT computation and reduce power consumption. We assume that the power consumption of the proxy is unconstrained, but the client has a tight power budget. Since the proxy-client bandwidth is constrained, those IDCT algorithms that require large overhead bits for the block class information would not fit well into our framework. On the other hand, we would not be able to harness the full power of the proxy in classifying the blocks, as well as helping the client in reducing the complexity, if the block classification is overly coarse-grained. In order to achieve a good balance, a proxy-specific IDCT algorithm is called for. Based on the dyadic IDCT algorithm [47], we propose such an IDCT algorithm that is amenable to the proxy framework. We show that through adaptation, an active proxy can further decrease the client's complexity, subject to a slight bandwidth increase constraint.

The rest of the chapter is organized as follows. Section 3.2 presents the new proxy-based framework that is able to provide the useful trade-off between client-proxy bandwidth and client complexity. Section 3.3 describes a proxy-specific IDCT algorithm and its complexity. Next, Section 3.4 discusses a case where the IDCT operation can be made faster if we shift the block classification task from the client to the proxy.

Section 3.5 further demonstrates the effectiveness of the proxy framework. An active proxy can help lower the client's complexity to a greater extent when adaptation to the image statistics is introduced. Finally, conclusion is drawn in Section 3.6.

3.2 Proxy Processing to Speed Up IDCT at the Client

In the existing image/video coding standards such as JPEG, H.263, MPEG-1, MPEG-2, etc, the inverse discrete cosine transform (IDCT) has long been known to account for a significant portion of the total decoding time budget. Lowering the complexity of IDCT can contribute to the overall power consumption reduction of the client where image/video decoding is performed. There exist already many fast IDCT algorithms [10, 11, 30, 32, 52, 87], however, one can do even better by designing *variable complexity* algorithms that take advantage of the fact that many DCT coefficients are quantized to zeros at the encoder, especially when the image quality is low. Significant speed-up is reported to be realizable by exploiting the sparseness of the IDCT blocks [89]. These IDCT algorithms skip computing the zeros in the input blocks, and thus achieve a variable degree of complexity reduction, in accordance with the zero patterns of the IDCT input. Refer to Chapter 2 for a detailed discussion of VCA.

Nevertheless, in such algorithms, the classification of zero patterns in the IDCT input block, if performed at the client, turns out to be a nontrivial overhead. When the zero testing takes an excessive amount of complexity, the complexity reduction achieved by computing fewer inputs will be diminished. In a client-proxy setting, however, we may be much better off if the task of zero testing is moved to the proxy.

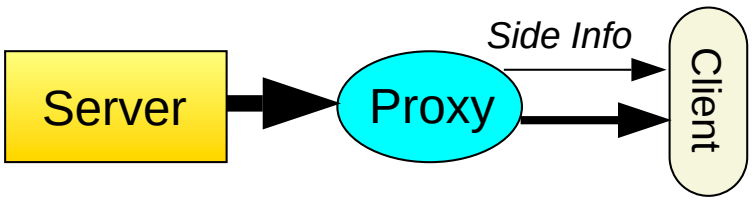


Figure 3.1: The proxy IDCT framework.

We thus propose a new framework based on the proxy. In Figure 3.1, the server stores the image/video data in compressed form. When the data is sent to the client for decoding, it passes through the proxy, which acts as an intermediary between the server and the client. The proxy can process the data for the purpose of helping the client decode faster. For example, in Section 3.4, zero patterns of IDCT inputs are tested and the resulting block class index is obtained by the proxy. These indices are then sent as side information alongside with the image data to the client, where IDCT is actually performed. The client decodes this side information and uses it to invoke the appropriate reduced IDCT subroutines. As a result, the complexity (power consumption) of the IDCT computation is reduced.

We can see that this framework introduces an interesting trade-off between a larger proxy-client bandwidth and the reduced client complexity. This trade-off becomes valuable when client power consumption is more of a concern than a slightly increased bandwidth. On the other hand, we want to design an IDCT algorithm that is more suitable to the distributed processing between client and proxy, so that the side information will be kept as low as possible. This issue is addressed in Section 3.3, where we discuss a proxy-specific fast IDCT algorithm.

Unlike common proxy schemes, we do not change or reformat the source data, and thus we do not introduce any impairment to the source data. Rather, such structural information as block class index is extracted from the data and sent to the client. Hence, the decoded image quality remains unchanged even if the speed of performing the IDCT changes. Note, however, that nothing would prevent the proxy from also changing the coded data while preserving compatibility with the decoder.

Note that here we assume that the power consumption of the proxy is unconstrained, but the client has a tight power budget. Depending on the amount of work it carries out, a proxy can be categorized as *simple* (light workload) or *active* (heavy workload). In Section 3.5, we show that an active proxy is capable of further reducing the IDCT complexity at the client, by performing a more aggressive processing.

3.3 Proxy-Specific Fast IDCT Algorithm

In the proxy-based IDCT framework we have described, IDCT inputs are tested and side information about their sparseness is sent from the proxy to the client. Thus we want to select an IDCT architecture that requires low side information overhead, while at the same time helps the client most in reducing the IDCT complexity.

In the case of extremely sparse IDCT inputs, the proxy can test the total number, N of non-zero entries in an 8×8 IDCT block. If N is below a threshold point B, then the client adopts the so called forward mapping IDCT [54, 57]. Otherwise, the client falls back to the baseline 8×8 IDCT algorithm. Thus 1 bit/block is needed for the proxy to convey this binary decision to the client. The spatial information of the non-zero

entries is not necessary since the client computes the IDCT of each non-zero coefficient independently. The intermediate IDCT results of the non-zero DCT coefficients are simply superimposed at the end. The threshold point B is determined by comparing in terms of complexity the forward mapping IDCT with a fast IDCT algorithm such as AAN[1] on a full 8×8 block. Our study shows that B is very small - around 8. This limits the practical use of this scheme.

There have been many finer-grained approaches in testing the zeros present in the IDCT input blocks [21, 36, 37, 48]. However, if these algorithms were incorporated in our proxy framework, the number of different classes would tend to place a heavy burden on the proxy-client bandwidth.

3.3.1 Dyadic 2D IDCT

The *dyadic* IDCT proposed by Lengwehasatit and Ortega [47] appears to be a good compromise. At the proxy, each 8×8 DCT block is classified into DC-only, low-2×2, low-4×4 and full 8×8 classes. The reason behind this classification is that high frequency components in a typical image are more likely to be quantized to zero, and non-zero coefficients tend to cluster around the DC component.

Only 2 bits/block are required to be transmitted to the client to indicate the class index. At the client, for each of the four possible classes, a corresponding 1D pruned IDCT is applied along the nonzero rows and then all columns.

The advantage of this scheme lies in the inexpensive 2D classification, as well as the low side information (bandwidth) requirement.

3.3.2 Proxy-Specific 2D IDCT

One disadvantage of the dyadic 2D IDCT is that it fails to yield satisfactory complexity reduction when a mismatch between the model and the statistics of data occurs. For example, for some sources, blocks of class 5×5 may be more likely to exist than those of class 4×4. But all 5×5 blocks are forced to map to the full 8×8 IDCT, thereby failing to capitalize the feasible complexity gain by using a 5×5 reduced (pruned) IDCT butterfly instead. We extend the dyadic IDCT to a proxy-specific 2D IDCT (PS-IDCT) scheme that is not only suitable for the proxy framework, but also provides a general scalable approach with excellent complexity performance. The details are given below.

For an 8×8 IDCT block $\mathbf{Y}$, where $\mathbf{Y}(i,j)$ is the IDCT input value at (i,j), with $i,j = 1,2,...,8$, we choose one out of 8 possible classes.

Definition A block $\mathbf{Y}$ is said to be of class k if $\mathbf{Y}(m,n) = 0$, $\forall\ m > k$ and $n > k$.

In PS-IDCT, a reduced $k \times k$ IDCT is applied to a block of class k (see Figure 3.2). A reduced $k \times k$ IDCT on class-k block can be decomposed into two separable operations, either row-wise followed by column-wise, or vice versa. In each operation, the same k-point reduced 1D IDCT is invoked. This is a nice attribute since the penalty for instruction cache misses can be avoided by adhering to the same subroutine in the locality of an IDCT program [89]. Note that although an IDCT algorithm can choose to carry out the row/column operations in different orders, depending on whether there are more zero columns or more zero rows, our PS-IDCT does not distinguish the

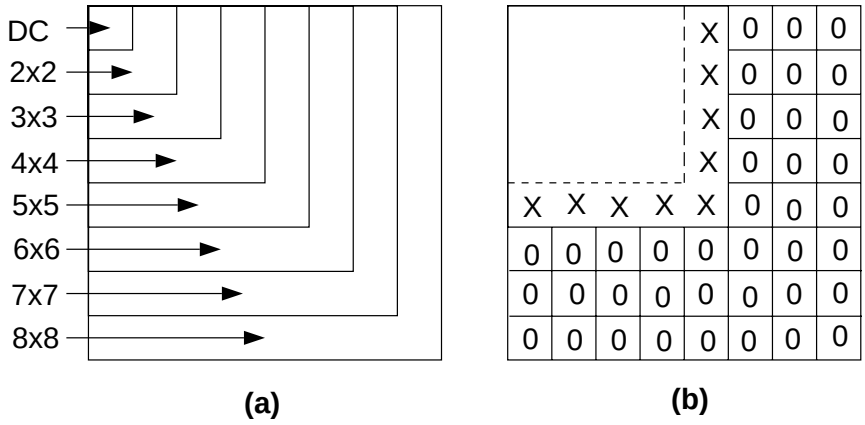


Figure 3.2: IDCT blocks of different classes. (a) Reduced $k \times k$ IDCT on class k block. DC is a special case, where all entries except the DC coefficient are zero, the output of the 1×1 IDCT is simply the scaled version of the DC coefficient. The other extreme is the full 8×8 IDCT, which is the least sparse case. (b) A block of class 5. Not all entries are zero in the boundary region marked by "X".

row/column order since it would incur extra side information overhead from the proxy to the client.

We choose the AAN 1D IDCT [1] as the baseline algorithm, which is supported in the independent JPEG software [44]. AAN IDCT is an efficient algorithm, requiring only 5 multiplications and 29 additions. It pre-scales all input DCT coefficients and combines the scaling factors with the de-quantization stage at the decoder. Figure 3.4.(a) shows the butterfly of the 8-point AAN IDCT. The k-point IDCT (where $y(m) = 0, \forall m > k$ and $m \leq 8$) can be obtained by pruning and merging some data paths of (a). A 2-point IDCT is illustrated as an example in Figure 3.4 (b).

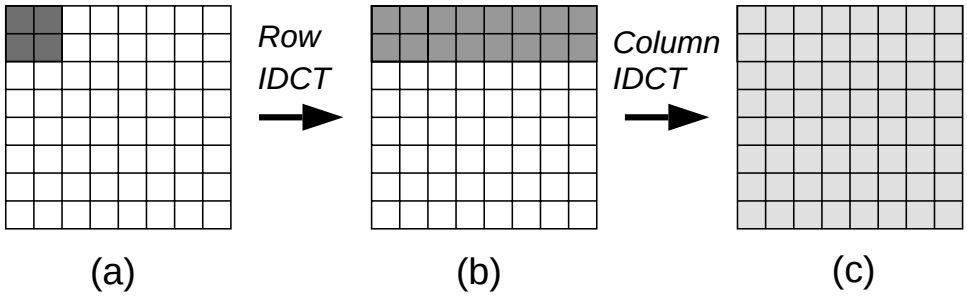


Figure 3.3: Separable 2D IDCT. Entries in non-shaded regions are all zero. (a)Class-2 block input; (b) A snapshot of the intermediate storage of the block, after the same 2-point 1D reduced IDCT is applied twice, one on each of the upper two rows. (c) After the same 2-point 1D reduced IDCT is applied 8 times, one for each of the column in (b). This completes the whole 2×2 reduced IDCT.

As an estimate, assume that the complexity of the k-point reduced 1D IDCT is C_k, then the $k \times k$ IDCT on class-k block has an overall complexity:

$$C_{k \times k} = (k + 8) \times C_k \tag{3.1}$$

The complexity of the PS-IDCT is given in Table 3.1, according to Eq.(3.1). Figure 3.3 gives the 2×2 IDCT as an example, which applies the 2-point 1D DCT 10 times. We can see that the PS-IDCT achieves finer granularity in complexity than the dyadic IDCT by allowing more IDCT input block classes. Furthermore, the PS-DCT can still maintain low side information: at most 3bits per block is necessary to distinguish all eight possible classes if we use fixed length code for the side information. On the other hand, the bandwidth can also be made scalable, based on the important observation that an $m \times m$ IDCT can be applied safely to a class k block as long as $m \geq k$. There exist many ways in which the proxy can map a set of eight classes to a smaller set

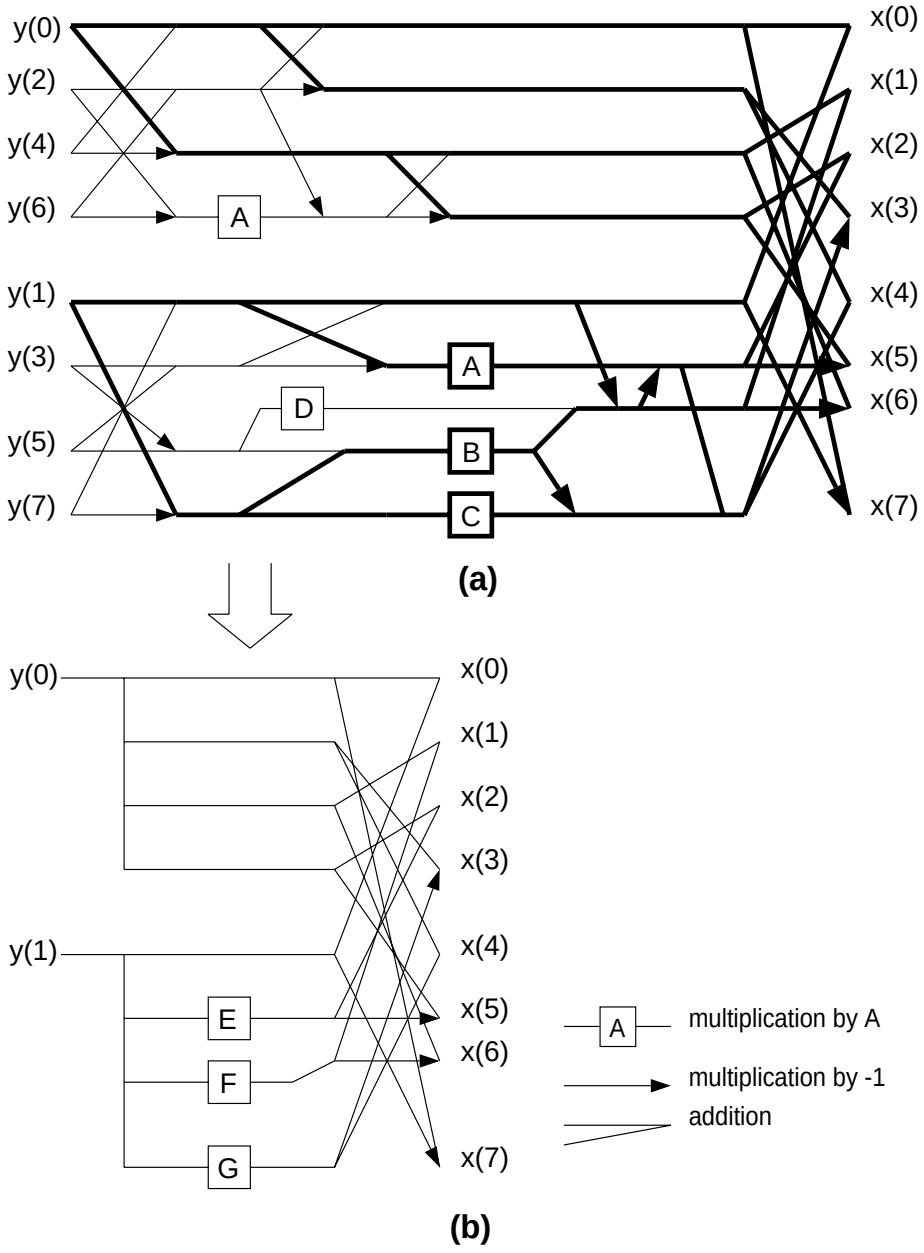


Figure 3.4: The pruned AAN IDCT algorithm. (a) Baseline full 8-point IDCT with 5 multiplications and 29 additions. The multipliers are: $A = 1.414213562$, $B = 1.847759065$, $C = 1.00823922$ and $D = -2.61312593$. If only $y(0)$ and $y(1)$ are non-zeros, then only the highlighted paths are contributing to the outputs $x(0)$ to $x(7)$; (b) A 2-point IDCT is obtained from (a), after we simplify the paths. Only 3 multiplications and 8 additions are required. The three multipliers are $E = A - B + 1$, $F = B - 1$ and $G = A - 2B + C + 1$.

containing, say, four classes, thereby reducing the side information overhead to around 2 bits per block. The operations performed by such an active proxy are presented in Section 3.5.

Table 3.1: Complexity of the IDCTs.

Class k	**k-point IDCT**		**$k \times k$ IDCT**	
	mult	**add**	**mult**	**add**
DC	0	0	0	0
2	3	8	30	80
3	4	12	44	132
4	5	20	60	240
5	5	23	65	299
6	5	25	70	350
7	5	27	75	405
8	5	29	80	464

3.4 Block Classification at the Proxy

One straightforward example to demonstrate the effectiveness our framework is to assign the block classification task to the proxy. In this fashion, the client is relieved of the burden to perform the zero testing, thereby further speeding up the IDCT operations.

3.4.1 Side Information

For the case of 8 block classes, 3 bits/block are required for coding the class index of each block. Only 2 bits/block are necessary for the 4-class case. Such side information overhead is relatively insignificant unless the coded image has very low bit rates.

We would be able to lower the side information by introducing entropy coding. However, it may be impractical to apply entropy coding directly to the block indices, since our study (see Figure 3.5 and Figure 3.6) shows that the class index statistics change from image to image and rate to rate, and thus one would need either to have a set of variable-length coding (VLC) tables for different images/rates or use an arithmetic coder to adaptively code the class indices.

One potential alternative is to use differential-coding of the class indices. In our study, the statistics of the index differences of the neighboring blocks show that the index differences tend to take small values (around zero), regardless of the bit rates and images. This would be more amenable to using a fixed VLC table rather than a set of tables.

Figure 3.7 illustrates the side information required when we perform differential entropy coding of the block indices. The side information is measured by the entropy E as follows.

$$E = \sum_{k=1}^{N} (-P_k \times \log P_k) \tag{3.2}$$

where P_k is the probability of symbol k occurring in an image, and N is the cardinality of the set of symbols. In PS-IDCT, where 8 classes of blocks are possible, the difference of block indices belongs to the set $[-7, 7]$ of 15 symbols. In the 4-class case such as the dyadic IDCT, we have the set $[-3, 3]$ of 7 symbols.

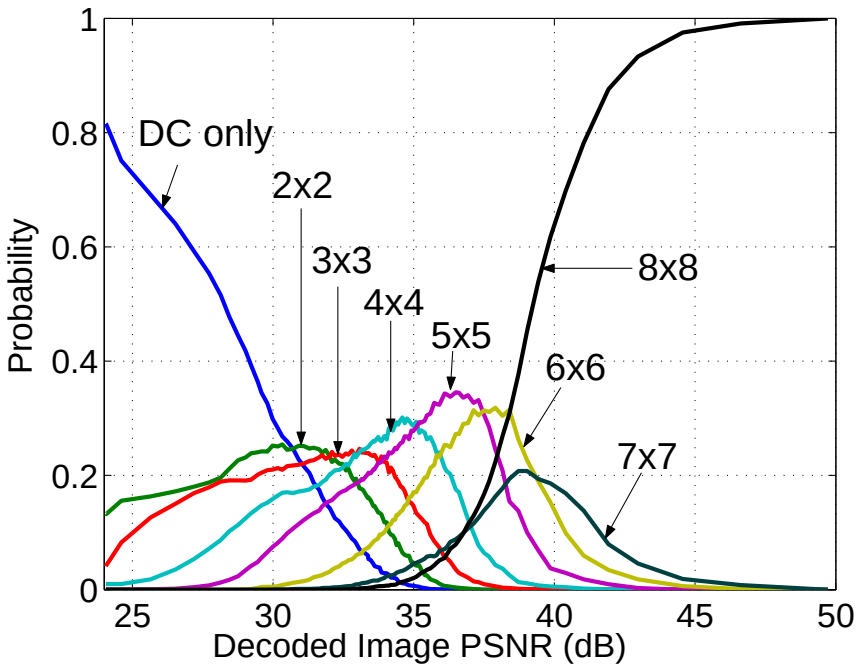
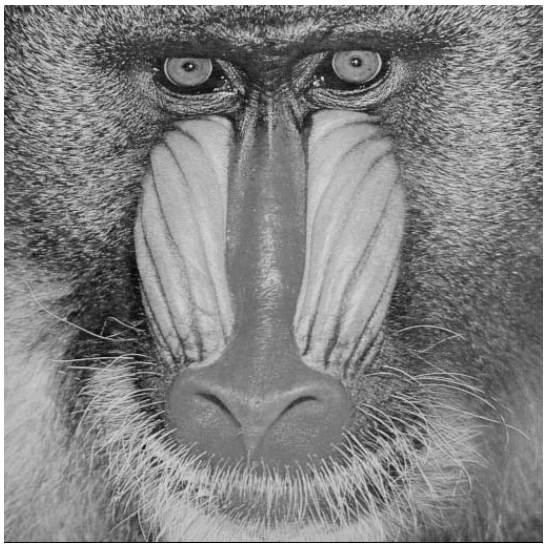


Figure 3.5: Statistics of classes of IDCT input blocks of "Lena". Probability of class 1 (DC only), the sparsest class, decreases with increasing image PSNR. The least sparse case, class 8 (full 8×8) is the opposite. In between are class 2 to class 7, which reach their peaks at PSNR values in ascending order. "Baboon" has fewer sparse IDCT blocks than "Lena" due to the fact that "Baboon" has more large high-frequency DCT coefficients.



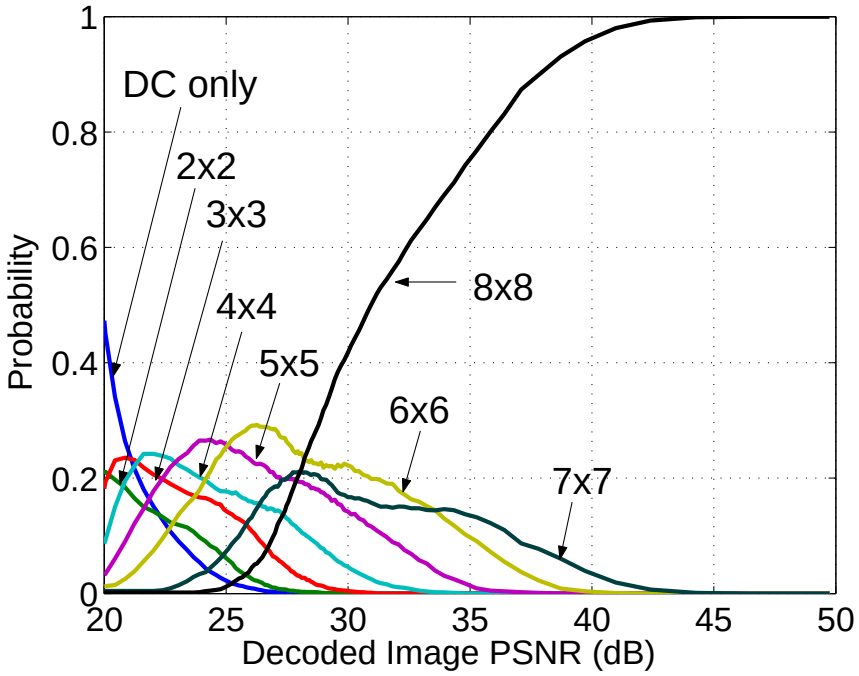


Figure 3.6: Statistics of classes of IDCT input blocks of "Baboon". There are fewer sparse IDCT blocks than "Lena" in Figure 3.5 because "Baboon" has more large high-frequency DCT coefficients.

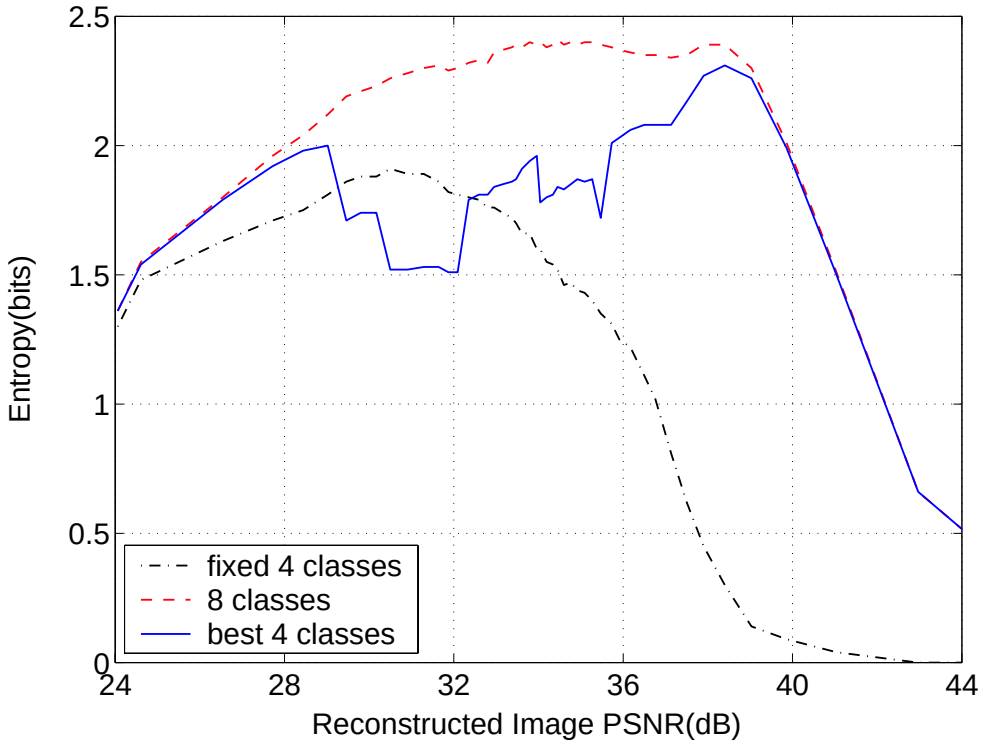


Figure 3.7: Side information from the proxy for "Lena". Comparison is made of three schemes: the proxy indicates to the client one class out of the set of classes $\{1, 2, 4, 8\}$ as in dyadic IDCT; or out of the set of the best 4 classes found by an active proxy described in Sec.3.5; or it uses the full classification including 8 classes.

3.4.2 Complexity Reduction by the Proxy

The task of block classification essentially involves testing the zeros present in the IDCT inputs. An efficient method to classify blocks is to use their EOB marker value, which can be obtained by parsing the compressed bit stream [89, 57]. A simple zero testing scheme is adopted in the IDCT subroutines of the IJPEG software [44], where the IDCT outputs are set to the DC term when all AC terms are found to be zero. For the purpose of comparison, we adopt the algorithm described below in pseudo code,

which starts zero testing from the outside (lower right) to the inside (upper left) of a block $\mathbf{Y}$.

```
k=8 ;
Repeat {
Test row Y(i,k) and column Y(k,j) against zero (i,j = 1,2,...,k );
If (at least one nonzero found) break ;
k = k − 1 ;
} Until (k == 1) ;
Return k as the block class.
```

This zero test algorithm is also a variable complexity algorithm. If block $\mathbf{Y}$ is not so sparse (of larger class index), it terminates earlier. On the other hand, if $\mathbf{Y}$ is very sparse, the algorithm will go all the way towards the upper left corner of the block, hence taking longer to finish.

We compare the complexity of the following three schemes:

1. IJPEG IDCT performed at the client;

2. PS-IDCT embedded into the IJPEG software and replacing the original AAN IDCT subroutines. The set of classes is chosen to be $\{1, 2, 4, 8\}$ as in the dyadic IDCT. Block classification is performed at the client and the complexity of these zero tests is included in the overall complexity computation; and finally

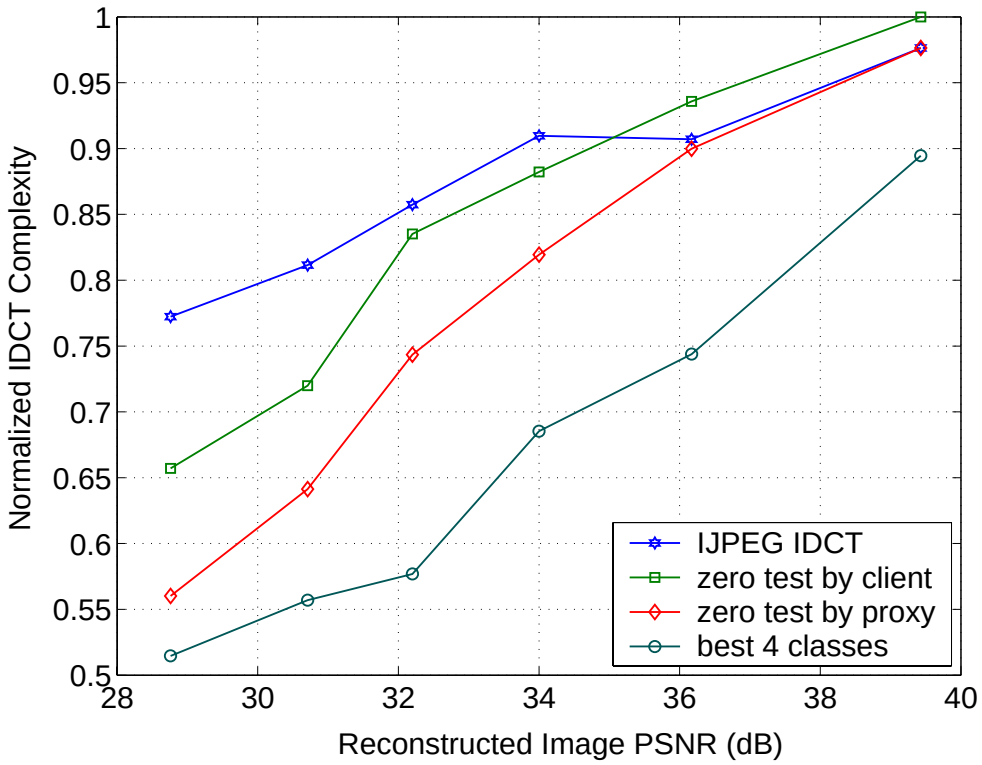


Figure 3.8: Comparison of the client complexity of the four schemes. The proxy scheme outperforms the IJPEG and the dyadic IDCT (without proxy) scheme consistently. The active proxy scheme achieves further reduction in complexity.

3. The same as scheme 2 except that zero test is carried out at the proxy. The test is thus not counted towards the complexity of the client.

Note that the client needs to spend time decoding the 2 bits/block side information, either obtained locally in scheme 2 or received from the proxy in scheme 3, so that one out of the four reduced IDCT AAN subroutines can be selected accordingly. The complexity of switching among the four reduced IDCT algorithms is also taken into account in both schemes 2 and 3.

The experimental results of the IDCT complexity at the client are shown in Figure 3.8. We can observe that the proxy scheme 3 has significantly lower complexity than the client scheme 2, over the whole range of PSNR values. This clearly demonstrates the effectiveness of our proxy framework, which incurs a slight increase of the proxy-client bandwidth, in exchange for an improved client complexity performance.

Scheme 2 outperforms the IJPEG when PSNR is smaller than 35 dB. This is expected since at low PSNR's, many blocks are of small index classes, we can still have an overall gain by invoking reduced IDCT of smaller classes, although more time is spent in the "greedy" zero testing in the client scheme than in the "shallow" zero testing in IJPEG. However, the return is diminishing with increasing PSNR values. The two curves crosses each other at 35dB, implying that the overhead of zero testing in the client scheme more than offsets the gain achieved by using reduced IDCT subroutines of larger class index. Nevertheless, the proxy scheme 3 outperforms the IJPEG consistently. They converge at very high image quality (PSNR = 40dB) because both schemes degenerate to the full 8×8 IDCT under such circumstances.

3.5 Active proxy under a bandwidth constraint

As mentioned in Section 3.4.1, making the set of classes smaller will help reduce the side information between the proxy and the client. In Section 3.4, a set of fixed classes is pre-determined and known by both the proxy and the client. In fact, the proxy can play a more active role in adaptively selecting a reduced number of classes based on the statistics of the image to be decoded at the client. For instance, the proxy can choose

the best four classes out of the eight possible classes. The side information about this selection can be sent to the client in the header field for the entire image. For instance, if the proxy finds that the best set of classes is $\{2, 3, 5, 8\}$, a total of 12 bits is attached in the image header destined to the client, which decodes the field and learns about the decision made by the proxy. The succeeding IDCT operations for the entire image at the client is then tied to these best four classes. We can see that the best-four scheme causes negligible increase in the bandwidth for the whole image, compared with its fixed-four class counterpart.

3.5.1 Problem Formulation

As an example, we formulate the best four class problem as follows. The more general problem of choosing the best N classes out of the total M $(N < M)$ classes can be formulated similarly.

Each element i $(i \in \{1, 2, ..., 8\})$ in the set $\mathcal{A}$ of cardinality 8 is to be mapped to an element j $(j \in \{1, 2, 3, 4\})$ of the set $\mathcal{B}$ of cardinality 4. The one-to-one mapping $i \mapsto j$ is given by $j = \min(n), \forall n \in \{1, 2, 3, 4\}$ that satisfy

$$f(n-1) < i \leq f(n) \tag{3.3}$$

where,

$$k = f(n) \tag{3.4}$$

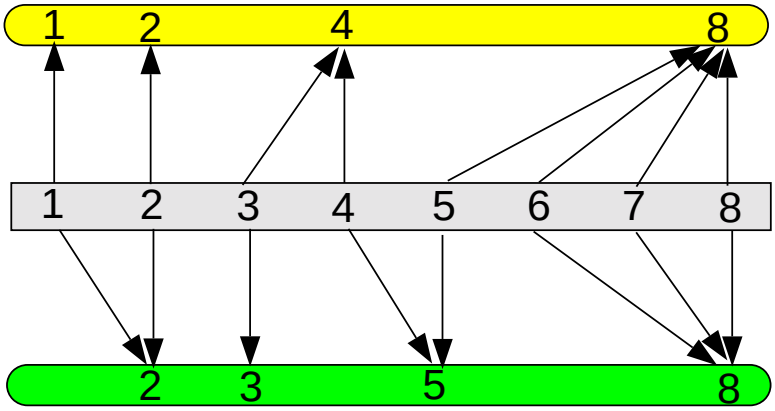


Figure 3.9: Two possible mapping schemes from a set of eight elements to a smaller set of four elements. In the mapping upwards, blocks of classes from 5 to 8 are merged and mapped to the fourth element in the smaller set on which the full 8×8 IDCT is applied, whereas in an alternative mapping downwards, blocks of class 5 are mapped to the third element that corresponds exactly to a reduced 5×5 IDCT.

i.e., the set $\mathcal{B}$ is chosen so that $k \times k$ reduced IDCT is applied to class n in set $\mathcal{B}$. Intuitively speaking, out of the eight classes of $\{1, 2, ..., 8\}$, the mapping is such that those that belong to set $\mathcal{A}$ are mapped to the nearest classes in set $\mathcal{B}$. See Figure 3.9 for two possible mapping schemes.

There are altogether $\binom{8}{4} = 70$ possible smaller sets the proxy can map the larger set to. The proxy can perform an exhaustive search and choose the best set that minimizes the following average complexity C_{avg},

$$C_{avg} = \sum_{j=1}^{4} C_{k \times k} \times P_k, \tag{3.5}$$

where $k = f(j)$, as defined in Eq.(3.4). And P_k is the probability given by $P_k = \frac{N_j}{N_t}$, where N_j is the number of blocks mapped to j, and N_t is the total number of blocks in the image. For a 512×512 image with block size being 8×8, N_t is 4096. $C_{k \times k}$ can

be approximated by Eq.(3.1). In our experiment, the number of operations listed in Table 3.1 are used. This information can be readily made available to the proxy prior to the transmission of source content. We can also measure the real running time (or power consumption) of each reduced IDCT subroutines at the client, so that the proxy can perform the optimization procedure based on more accurate estimations.

3.5.2 The Greedy Method

In order to reduce the complexity of exhaustively searching all the possible reduced sets in the optimization process, we propose a more efficient greedy algorithm that consists of four phases. The algorithm starts with a set of 8 classes $\{1, 2, ..., 8\}$. Each class p has its associated IDCT complexity $C_{p\times p}$, and the number of blocks falling into this class (occupation) N_p. In each phase, we examine any two neighboring classes p and q. Without loss of generality, we assume $q > p$. Note that $q = p + 1$ may not hold in other phases after the phase 1. We define the cost of merging the class pair (p, q) into one class q as ΔC_{pq}, which can be expressed as the increased complexity caused by the merging:

$$\Delta C_{pq} = (N_p + N_q) \times C_{q\times q} - (N_p \times C_{p\times p} + N_q \times C_{q\times q}) = N_p \times (C_{q\times q} - C_{p\times p}) \quad (3.6)$$

In Eq.(3.6), the new occupation after merging the class pair (p, q) is the sum of the occupations of class p and q respectively, while the new complexity of the merged pair is taken to be that of class q.

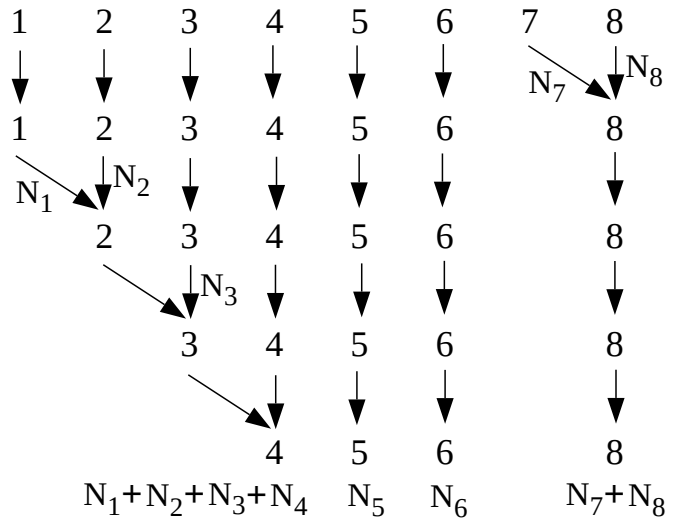


Figure 3.10: An example of merging 8 classes using the greedy approach.

Among all possible merging operations in a phase, we choose the pair of neighboring classes (p, q) found to have the minimal cost. As a result, class p is removed from the set, while class q is updated to a new class by increasing its occupancy to $(N_p + N_q)$. The class complexity $C_{q \times q}$ remains unchanged. It follows that the next phase will operate on a smaller set that has one fewer class than the current phase. After four phases of merging the minimal cost pairs, there are only four classes left in the set. Figure 3.10 illustrates the four phases of merging the original eight classes to a reduced set of four classes. Our simulation results show that they are identical to the best four classes found by the exhaustive search method.

As for the complexity, in phase i, where $i = \{1, 2, 3, 4\}$, at most $(8 - i)$ subtractions, multiplications and comparisons are required. The subtraction operation can be replaced by the table lookup if we pre-compute all the class complexity difference

$(C_{q\times q} - C_{p\times p})$. Hence the total complexity of the greedy algorithm can be reduced to 22 $(= 7 + 6 + 5 + 4)$ multiplications and 18 comparisons. By contrast, the foregoing exhaustive search method has to compare the cost of $\binom{8}{4} = 70$ possible smaller sets, with each set involving 4 multiplications. Thus a total of up to 280 multiplications and 69 comparisons are required. The advantage of the greedy method is self evident. Moreover, the existence of a greedy algorithm makes it conceivable to perform this class selection among more than 8 classes, without the significant increase in computation required by an exhaustive search.

3.5.3 Results

Table 3.2 gives the best 4 classes generated by the active proxy algorithm. Two observations can be made: (1) The classes shift to the right (larger index) as PSNR value increases. This is due to blocks becoming less and less sparse as the quantization gets finer; (2) When the image quality is low, i.e., 30dB or lower, the blocks are so sparse that even the class-7 and class-8 blocks are absent, i.e., it is not cost efficient to compute all the DCT coefficients. Hence the proxy can instruct the client to apply 6×6 IDCT to class-5 blocks rather than the wasteful full 8×8 IDCT, as dictated by the fixed-four scheme of $\{1, 2, 4, 8\}$.

Simulation results agree well with our theoretical prediction. The complexity of the best-four class scheme is remarkably lower than the fixed-four class scheme, regardless of the reconstructed image quality (Figure 3.8). This clearly demonstrates the client performance improvement made possible by the adaptation at an active proxy.

Table 3.2: Best 4 classes for "Lena" under several decoded image PSNR values.

PSNR(dB)	**Classes**							
28.76	1	2	3			6		
30.71	1		3		5		7	
32.20		2	3		5			8
34.68			3	4		6		8
36.17				4	5	6		8
37.94					5	6	7	8
39.43					5	6	7	8

Both the static and dynamic schemes require 2 bits/block overhead. If differential entropy coding of block indices is used, the static scheme consumes less side information bits than the dynamic one in most cases (Figure 3.7). This, again, exhibits an interesting trade-off between complexity reduction and side information increase.

The optimization process requires a global view of the statistics of the whole image, hence the processing delay, when combined with the classification complexity, would be unacceptable if the optimization is performed by the client device. The proxy relieves the burden of the client since the proxy typically has access to powerful computational resources.

3.6 Conclusions and Future Work

We have proposed a novel proxy framework in wire/wireless multimedia communication, which is capable of help accelerating the IDCT operations performed at the client. We have introduced a fast IDCT algorithm that is conducive to the client-proxy-server

architecture. We have demonstrated the effectiveness of the proxy in providing the useful trade-off between client-proxy bandwidth and IDCT complexity at the client. We have also shown that through adaptation, an active proxy can further improve the client complexity performance to a great extent, without violating the bandwidth constraint.

One potential extension of this work is to selectively set some high-frequency DCT coefficients to zero, with the goal of optimizing the decoded image quality subject to either the client complexity.

Chapter 4

Adaptive Computation Control of Variable Complexity Fano Decoders over Memoryless and Fading Channels

4.1 Introduction

The popularity of cellular telephone and satellite paging systems is driving the trend of mobile communications systems. Third generation (3G) cellular/PCS systems are attempting to integrate mobile data services such as mobile telephony, wireless LAN, home networks and mobile web access. The primary design challenge imposed by mobility is the need for designs that efficiently use the battery power of the mobile communication terminals. For example, a user that is located close to a base station should be able to operate at lower power than users roaming further afield. Similarly, low power decoding is desired when the available power is low or the user foresees the need for an extended use before re-charging. If we assume that the computation complexity of a decoding algorithm is a good approximation of the power consumption of the system running the algorithm, then *variable complexity algorithms* (VCA) can provide trade-offs that lead to variable power consumption or processing speed. By

employing VCA in mobile communications, we can achieve potential gains in average performance due to the changing nature of transmission channels. In order to make these VCA-systems operate in practical scenarios it is essential to develop techniques that will enable the channel decoder to adjust to the changing environment. For instance, algorithms will be needed to make the decoder operate in "high power" mode whenever channel conditions worsen (e.g., the SNR is lower). We call these techniques *Adaptive Computation Control*, because their goal is to match the computation level to channel conditions. We will demonstrate the use of computation control for the decoding of convolutional codes.

4.1.1 Convolutional Codes

Convolutional codes have found widespread application in wireless communication systems such as GSM and CDMA [71][80]. A convolutional code is generated by passing the information sequence to be transmitted through a linear finite-state shift registers. In general, the shift register consists of K stages and n linear algebraic function generators (e.g., modulo-2 adders). The binary input data to the encoder is shifted into and along the shift register k bits at a time. The number of output bits for each k-bit input sequence is n bits. Consistent with the definition of the code rate for a block code, the code rate of a convolutional code is defined as $R_c = k/n$. We call the parameter K the *constraint length* of the convolutional code.

Figure 4.1 depicts one of the convolutional codes used in our study. Tree diagram and trellis diagram are among the common methods to describe a convolutional code.

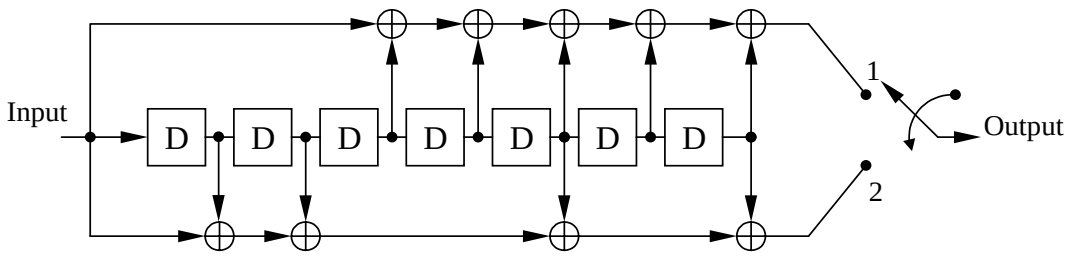


Figure 4.1: Convolutional code.

In the following, we shall give an introduction to two decoding algorithms, the Viterbi algorithm and the Fano algorithm. Viterbi algorithm is inherently a fixed complexity algorithm, while the Fano algorithm is a variable complexity algorithm. In a mobile communications system, it is often desirable to efficiently use the battery power of the mobile communication terminal. For example, a user that is located close to a base station should be able to operate at lower power than users roaming further afield. Therefore, Fano decoders are of our interest since their complexity can vary with different channel conditions. Nonetheless, buffers are needed due to the variable processing delay of Fano decoders.

4.1.2 Viterbi Algorithm

In the decoding of a block code for a memoryless channel, we compute the distances (Hamming distance for hard-decision decoding and Euclidean distance for soft-decision decoding) between the received code word and the 2^k possible transmitted code words. Then we select the code word that is closest in distance to the received code word. A convolutional code is essentially a finite-state machine. Hence the optimum decoder is a maximum-likelihood sequence estimator that involves a search through the trellis for

the most probable sequence. In order to measure the closeness of the received sequence to the coded sequence, metrics are defined for different branches and paths in a trellis. The criterion for deciding between two paths through the trellis is to select the one having the larger metric, which is called a survivor.

The Viterbi algorithm is known for optimum decoding of the convolutionally encoded information sequence. In general, when a binary convolutional code with $k = 1$ and constraint length K is decoded by means of the Viterbi algorithm, there are 2^{K-1} states. Therefore, there are 2^{K-1} surviving paths at each stage and 2^{K-1} metrics, one for each surviving path. Furthermore, a binary convolutional code in which k bits at a time are shifted into an encoder generates a trellis that has $2^{k(K-1)}$ states. The Viterbi algorithm keeps track of such $2^{k(K-1)}$ surviving paths and $2^{k(K-1)}$ metrics. At each stage of trellis, of the 2^k paths that merge at each node, only one survives as the most-probable path. Consequently, the decoding complexity at each stage increases exponentially with k and K, thus limiting the use of the Viterbi algorithm to relatively small values of k and K. Moreover, the memory required to store the entire length of surviving sequences is large and expensive. On the other hand, Viterbi decoder is a fixed complexity algorithm in that it compares all possible paths against the received sequence and picks the closest match, regardless of the channel condition. On one hand, the Viterbi decoder is good for hardware implementation because its computation is very regular. One the other hand, it cannot provide us with the desired capability of faster decoding when the channel is of high Signal-to-noise-ratio (SNR) and thus the received sequence tends to be less corrupted by the noise and easier to decode. That

is why a class of variable complexity algorithms called sequential algorithms are of interest.

4.1.3 Fano Algorithm

Sequential decoding was introduced in 1961 by Wozencraft as a method of maximum likelihood sequence estimation with typically lower computational complexity than the Viterbi algorithm [51]. Of several versions of sequential decoding algorithms, the Fano algorithm is generally considered to be the most practical to implement [16, 22, 84, 90].

A Fano decoder explores one hypothetical data sequence at a time by locally encoding it and comparing it with the noisy encoded version that is actually received. The decoder examines the metric of a potential path. If the metric value dips below a threshold T, the decoder backs up and begins to examine other paths. If no path can be found whose metric value stays above the threshold, the threshold is then loosened ($T \leftarrow T - \Delta$) and the decoder moves forward again with a lower threshold. As long as the decoder moves forward to a node as a first visit, the threshold is tightened ($T \leftarrow T + \Delta$) to ensure no endless loop occurs and the decoder eventually reaches the end of the tree.

The complete flowchart of the Fano algorithm is shown in Figure 4.2. The decoder starts at the origin node with the threshold $T = 0$ and the metric value $M = 0$. It then looks forward to the best of the 2^k succeeding nodes that has the largest metric. If $M_F \leq T$, where M_F is the metric of the forward node being examined, the decoder moves to this node. If this node has been examined previously, no threshold tightening is performed. If $M_F > T$, the decoder then looks backward to the preceding node.

If $M_B \leq T$, where M_B is the metric of the backward node being examined, then the decoder moves back to the preceding node P. If this backward move was not from the worst of the 2^k nodes succeeding node P, the decoder then looks forward to the next best of the 2^k nodes succeeding node P. Otherwise, the decoder again looks back to he node preceding node P. If $M_B > T$, then T is lowered by Δ and the look forward to the best node step is repeated. Ties in metric values can be resolved arbitrarily without affecting average decoder performance.

The number of computations performed by the Fano algorithm depends on how the threshold increment Δ is selected. In general, if Δ is small, a large number of computations tend to result. Making Δ larger will reduce the number of computations. However, Δ cannot be raised indefinitely without affecting the error probability. For the Fano algorithm to find the maximum likelihood path, T must at some point be lowered below the minimum metric along the maximum likelihood path. If Δ is too large, when T is lowered below the minimum metric of the maximum likelihood path, it may also be lowered below the minimum metric of several other paths, thereby making it possible for some of these suboptimal paths to be decoded before the maximum likelihood path [51]. Therefore, Δ is a parameter that controls the tradeoff between the bit error rate (BER) and the decoder complexity, which translates to power consumptions of the decoder. It will be shown that in general, faster decoding means higher bit error rate and vice versa. An analogy can be made between Δ and the quantization parameter (step size) in image/video coding, since the quantizer controls the tradeoff between decoded image distortion and bit rate [58].

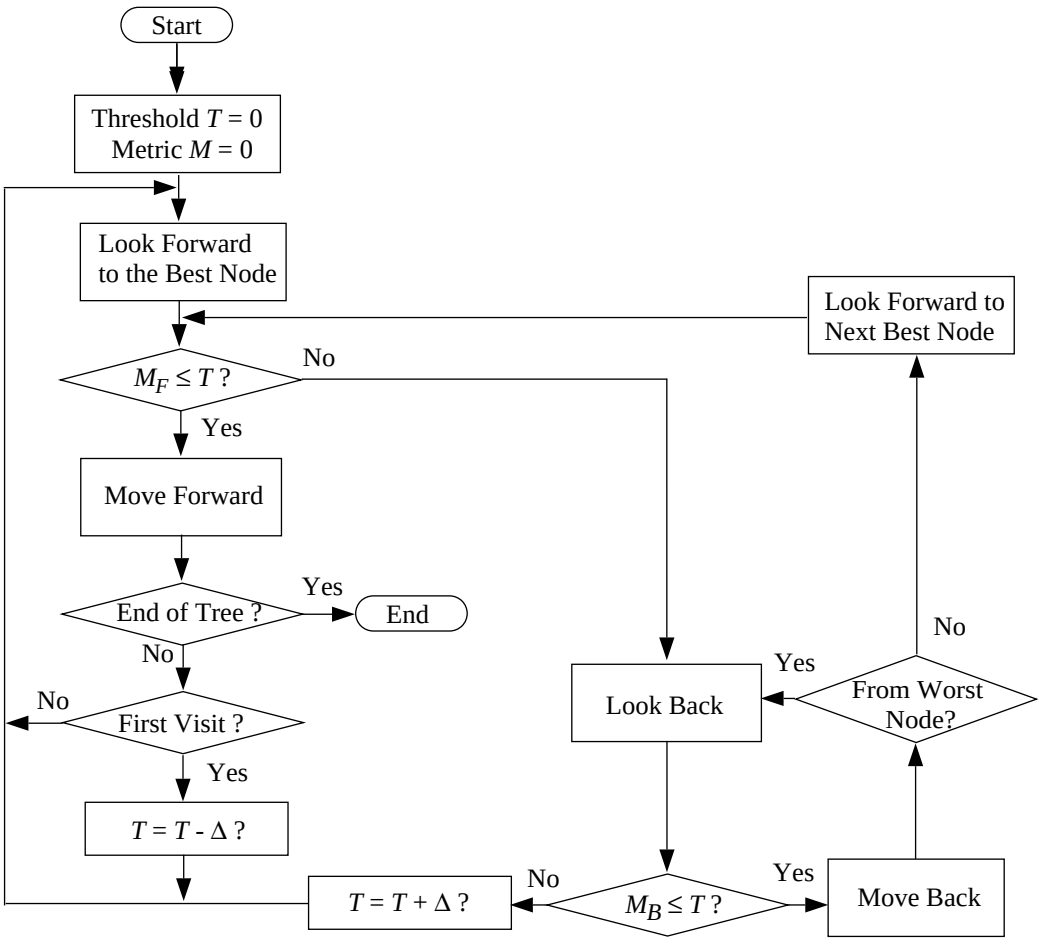


Figure 4.2: Flowchart of the Fano algorithm.

4.1.4 Metric Calculation

In Viterbi decoding, the path metrics are directly proportional to the likelihood functions since all paths of the same length are compared in parallel. The idea behind the Fano decoding is to explore at any time instant the most promising branches of a subtree of the entire code tree. As a consequence, a metric is required that enables us to compare paths of different lengths. The well known Fano bit metric is defined as [16, 51]:

$$M(r_i|v_i) = \log_2 \frac{P(r_i|v_i)}{P(r_i)} - R, \tag{4.1}$$

where $P(r_i|v_i)$ is a channel transition probability, R is the coding rate, and $P(r_i)$ is a channel input symbol probability given by

$$P(r_i) = \sum_{m=1}^{k} \left[P(r_i|v_m) \times P(v_m) \right], \tag{4.2}$$

where k is the cardinality of the set of the symbols transmitted.

The partial path metric for the first l branches is

$$M([\mathbf{r}|\mathbf{v}]_{l-1}) = \sum_{i=0}^{nl-1} M(r_i|v_i) = \sum_{i=0}^{nl-1} \log_2 \frac{P(r_i|v_i)}{P(r_i)} - nlR, \tag{4.3}$$

where the second term nlr increases with the path length l.

For a BSC (binary symmetric channel), the Fano bit metric in Eq.(4.1) can be simplified as

$$M(r_i|v_i) = \begin{cases} \log_2 2P - R, & \text{if } r_i \neq v_i; \\ \log_2 2(1-P) - R, & \text{if } r_i = v_i. \end{cases} \tag{4.4}$$

where $P = P(r_i|v_i)$, with $r_i \neq v_i$.

4.1.5 Computation Control Problem

In a mobile communications system, variable complexity fano decoders are beneficial since they enable efficient use of the battery power of the mobile communication terminal when the channel is good. In [79], a chip design based on the Fano algorithm was shown to achieve significantly lower energy consumption for an *additive white Gaussian noise* (AWGN) channel with high SNR ($\geq$ 6dB), compared to the Viterbi decoder.

On the other hand, the Fano decoder has inherent non-deterministic processing delay, which necessitates buffering of data before and after the decoder in practical real-time decoding systems. Fano decoders are not the only systems having variable delay characteristics. As an example, commercially available LDPC (low-density parity-check) processors require buffers to accommodate the varying processor throughput [18]. Therefore we are faced with an interesting tradeoff — if the decoder runs faster by choosing a large Δ in order to avoid buffer overflow, the "coarser" decoding will cause more blocks to be decoded in error, which will be considered lost. Alternatively, if the decoder uses a small Δ in order to achieve "finer" decoding, then the decoded blocks will tend to contain few bit errors. However, the decoder will be slowed down, and the buffer will tend to fill up. If the buffer becomes full, blocks have to be dropped. Thus our objective is to design a computation control algorithm that can minimize the average number of lost blocks under the limited buffer size constraint.

There has been extensive research on buffer control techniques in the source coding literature [2, 12, 34, 72, 74, 91]. However, to the best of our knowledge, no solution has been proposed to the specific buffer control problem under consideration here. In

this chapter, we present our solutions for two scenarios. In the case of memoryless, AWGN channels, our approach is to pick the best Δ that minimizes the probability of block loss, based on the joint distribution of the decoding complexity and the BER [61]. For slow, flat Rayleigh fading channels, we use a first state, two-state Markov model to capture the memory behavior of the channel. We demonstrate that by taking advantage of the memory of the channel, we can achieve better Δ control performance than assuming the channel is memoryless. Note that buffer sizes cannot be too long in interactive applications.

The chapter is organized as follows. In Section 4.2, we discuss the Δ control in the buffer framework. Next, the relations between the buffer occupancy, complexity and block errors are established in Section 4.3. Section 4.4 presents the solution to the Δ control problem for memoryless, AWGN channels. We then describe a simple table lookup algorithm, followed by its simulation results. Section 4.5 discusses the finite-state Markov modeling of the flat, slow fading channel and the calculation of the Fano metric tables for the decoding over fading channels. Section 4.6 extends the table lookup algorithm to cope with channels with memory. Prediction algorithms are introduced to exploit the memory property of the consecutive blocks. Simulation results show that we can further improve the control performance by using two tables, as compared to using just one table by assuming that the channel is memoryless.

4.2 Buffer for the Fano Decoder

The variable complexity advantage of the Fano decoder comes with the price of requiring buffers in a practical system. As shown in Figure 4.3, it is necessary to introduce a buffer memory to store and queue the incoming data blocks until the Fano processor can decode them. We assume that the data from the channel is divided into blocks. A block consists of inner and outer codes if we use concatenated error-control codes.

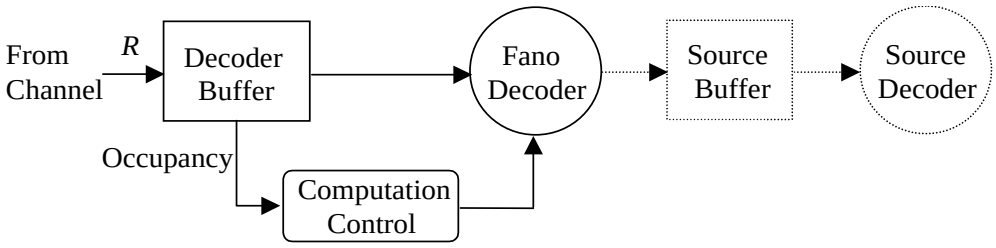


Figure 4.3: Fano decoder buffer. From the channel, blocks of data are input at a constant rate R to the Fano decoder buffer, where they will be removed at a variable rate since the decoding complexity is variable. If the decoded data is consumed at a constant rate by a source decoder (e.g., voice or video decoder), then the source buffer will be needed. Note that our work focuses on the decoder buffer.

In our experiments, we use concatenated error-control codes, which are prevalent in nowadays' GSM systems. The inner code is a constraint length 7, rate 1/2 convolutional code with outer code being the Reed-Solomon (RS) code with certain degree of error correction capability.

In the Fano decoder, Δ is a control parameter. Some simulation results are shown in Figure 4.4 and Figure 4.5. In Figure 4.4, we can see that the average complexity of Fano decoders decreases when we increase Δ. Also, for the same Δ, the decoder runs faster when the channel is less noisy (of higher SNR). In Figure 4.5, we can see that

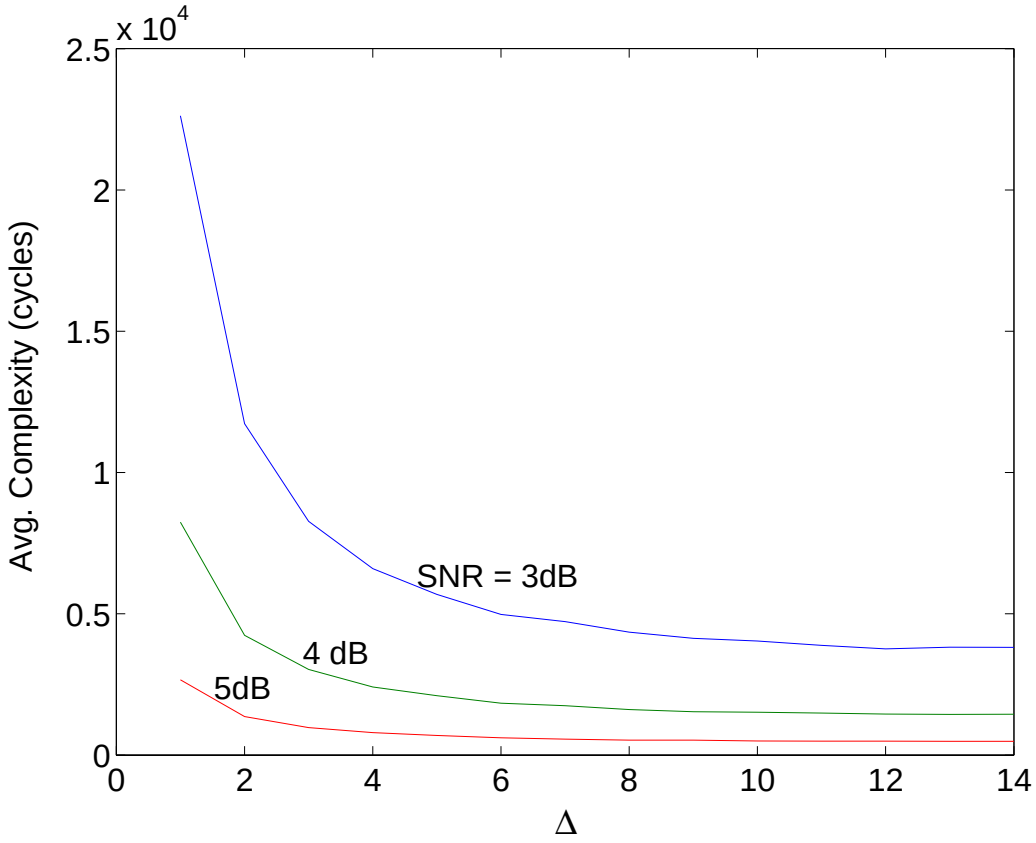


Figure 4.4: Average complexity as a function of Δ and channel SNR.

the average bit error rate (BER) of the decoded block tends to increase when we use a larger Δ. As can be expected, for the same Δ, a more noisy channel tends to cause a larger BER.

Therefore, if Δ is chosen to be small, a block is less likely to be decoded in error, however, the decoding complexity increases. As shown in Figure 4.3, data blocks keep coming into the buffer at a constant speed from the channel, while the data in the buffer are taken away from the buffer and decoded by the decoder. If the decoder is emptying the buffer slowly, some incoming blocks might be dropped due to buffer overflow.

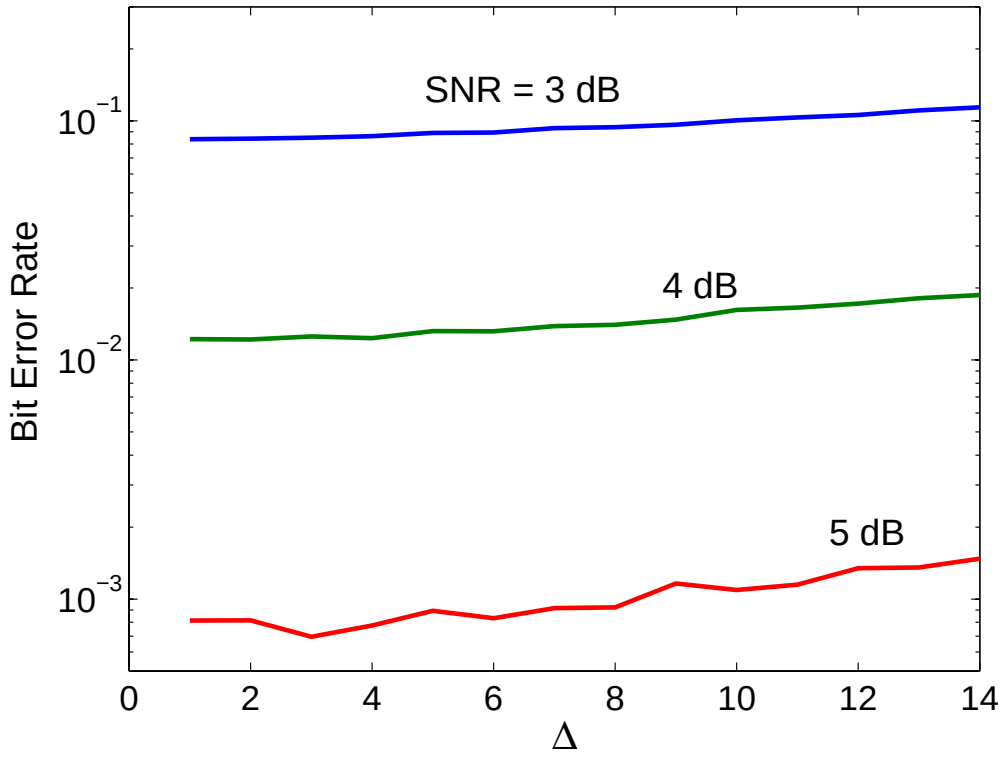


Figure 4.5: Average BER as a function of Δ and channel SNR.

Conversely, if the Δ is chosen to be large, the decoding complexity will decrease, and the probability of overflow decreases due to faster decoding operations. Nevertheless, decoded blocks are likely to contain greater number of bits in error because of higher BER's. If a data block contains too many error bits to be corrected by the outer RS code, the entire blocks will be rendered useless for the source decoder, therefore, this block shall also be considered lost.

Hence, our objective is to design a Δ control policy such that the overall probability of block loss is minimized, subject to the constraints of a finite buffer size.

4.3 Buffer Occupancy, Complexity and Block Errors

In this section, we aim to describe in detail the relations among the buffer occupancy, Fano decoding complexity and the block error rates. This serves as the basis for our design of an appropriate buffer control scheme.

4.3.1 Buffer Occupancy and Complexity

In Figure 4.6, at time t, the Fano processor has to fetch the next block from the buffer for decoding. Assume that the Fano processor runs at a clock frequency of f Hz, and that decoding a given block takes c cycles (note that c can vary from block to block), then during the decoding time $T_d = (c/f)$, one block (L bits) will be removed from the decoder buffer by the processor, while $(T_d \times R)$ bits will be fed into the buffer from the channel. Given the buffer occupancy $O(t)$, and the buffer size B_{max}, we can determine a threshold complexity C_t, such that if the processor is to decode the block with a complexity $c \leq C_t$, then no buffer overflow will occur.

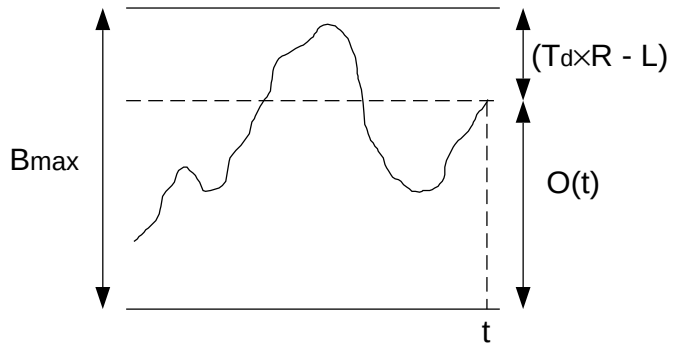


Figure 4.6: Mapping of the buffer occupancy to the decoder complexity.

To avoid overflow, the buffer needs to have enough space to store all incoming bits while decoding of the current block proceeds, i.e.,

$$\frac{c}{f} \times R - L + O(t) \leq B_{max}, \tag{4.5}$$

or equivalently,

$$c \leq C_t := [B_{max} + L - O(t)] \times \frac{f}{R}. \tag{4.6}$$

Since the decoding time of the Fano processor is variable, decoding of a certain block may require more than C_t cycles, thus leading to a buffer overflow. We can further determine a family of complexity thresholds C_m, where m is an integer. C_m is the decoding complexity such that if the actual complexity $c < C_m$ then at most m blocks are lost due to buffer overflow. If $m = 0$, then $C_0 = C_t$, i.e., no buffer overflow. We have

$$\frac{C_m}{f} \times R - L + O(t) = B_{max} + m \times L, \tag{4.7}$$

and therefore

$$C_m = C_{m-1} + \delta, \tag{4.8}$$

where

$$\delta = L \times \frac{f}{R} \quad , \tag{4.9}$$

which is independent of the buffer occupancy.

Note that if an incoming block gets dropped because the buffer is already full, then the entire block gets dropped. Hence, any decoding complexity that lies with the region $(C_{m-1}, C_m]$ translates to m blocks dropped (Figure 4.7).

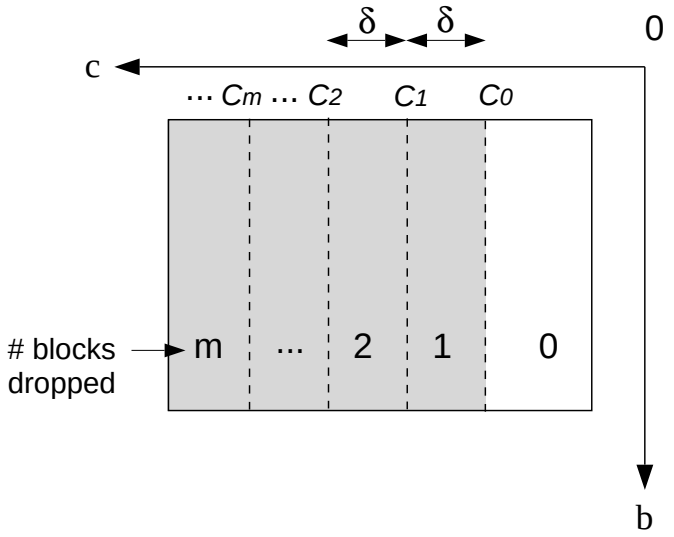


Figure 4.7: A family of complexity thresholds C_i for determining the number of blocks dropped due to buffer overflow. Note that C_i is a function of $O(t)$, similar to C_t in Eq.(4.6).

4.3.2 Joint Distribution of Bit Errors and Complexity

In order to devise an adaptive Δ control policy, we study the joint distribution of the number of bit errors in a block b after it is decoded by the Fano decoder, and its decoding complexity c. We use the same complexity model as in [79].

The two-dimensional sample space (b, c) can be partitioned into four quadrants (Figure 4.8) by the complexity threshold C_t and the error bit number threshold E_t.

In a practical system, E_t is determined by the error correcting capability of the error-correcting code (outer code) in use. For example, for a Reed-Solomon code specified as RS (n,k) with s-bit symbols, the encoder takes k data symbols and adds parity symbols to make an n symbol codeword. There are $(n-k)$ parity symbols. The Reed-Solomon decoder can correct up to t symbols that contain errors in a codeword, where $2t = n-k$. If $s = 1$, then $E_t = t$.

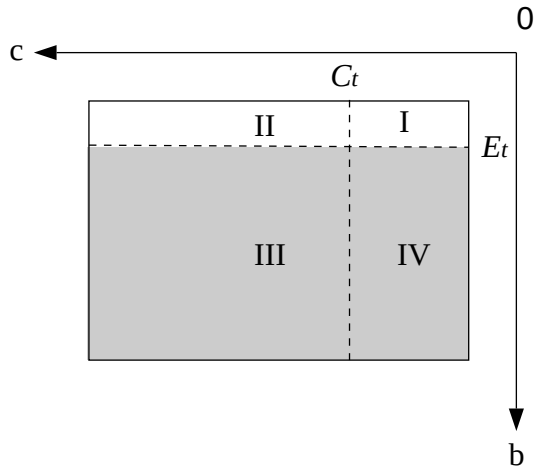


Figure 4.8: Partition of the sample space (b, c). C_t is a threshold such that any complexity above C_t will cause buffer overflow. The shaded area ($b > E_t$) represents blocks that are declared in error after the Fano decoding, since the succeeding RS code cannot correct over E_t error bits in a block.

Characteristics of each quadrant are summarized in Table 4.1. For example, any sample (b,c) in quadrant I ($c \leq C_t$, and $b \leq E_t$) corresponds to the case where the decoder runs so fast that overflow is avoided. Meanwhile, the decoded blocks contain E_t (E_t can be greater than 1) or fewer bit errors, which can still be corrected by the RS code. Note that outer than the RS code, a CRC (Cyclic Redundancy Check) code can be used for the purpose of error detection.

Table 4.1: Partition of the space.

Quadrant	Overflow	Block Error
I	No	No
II	Yes	No
III	Yes	Yes
IV	No	Yes

An example of a resulting set of 2-D histograms is shown in Figure 4.9. The main conclusion to be drawn from the histogram is that the distributions of b and c are far from being independent. Thus, in what follows, we will use a joint distribution for b, c in our optimization. Note that our simulations are based on the same b, c distribution, i.e., there is not any mismatch between the model and experimental data.

4.4 Optimal Δ Control for Memoryless channels

4.4.1 Formulation

The adaptive channel decoder buffer control problem can be formulated as follows:

Given a certain channel SNR and a finite buffer size B_{max}, at a given point in time t, the Fano decoder is to decode a block already in the buffer. The buffer occupancy $O(t)$ is:

$$O(t) = \begin{cases} N(t) & \text{if } t \leq \triangle T_d; \\ min(N(t) - R_p(t - \triangle T_d), B_{max}) & \text{if } t > \triangle T_d. \end{cases} \tag{4.10}$$

In Eq.(4.10), $N(t)$ is the number of bits input to the buffer up to time t, and $R_p(t)$ represents the number of bits decoded by the Fano decoder. Note that since the Fano

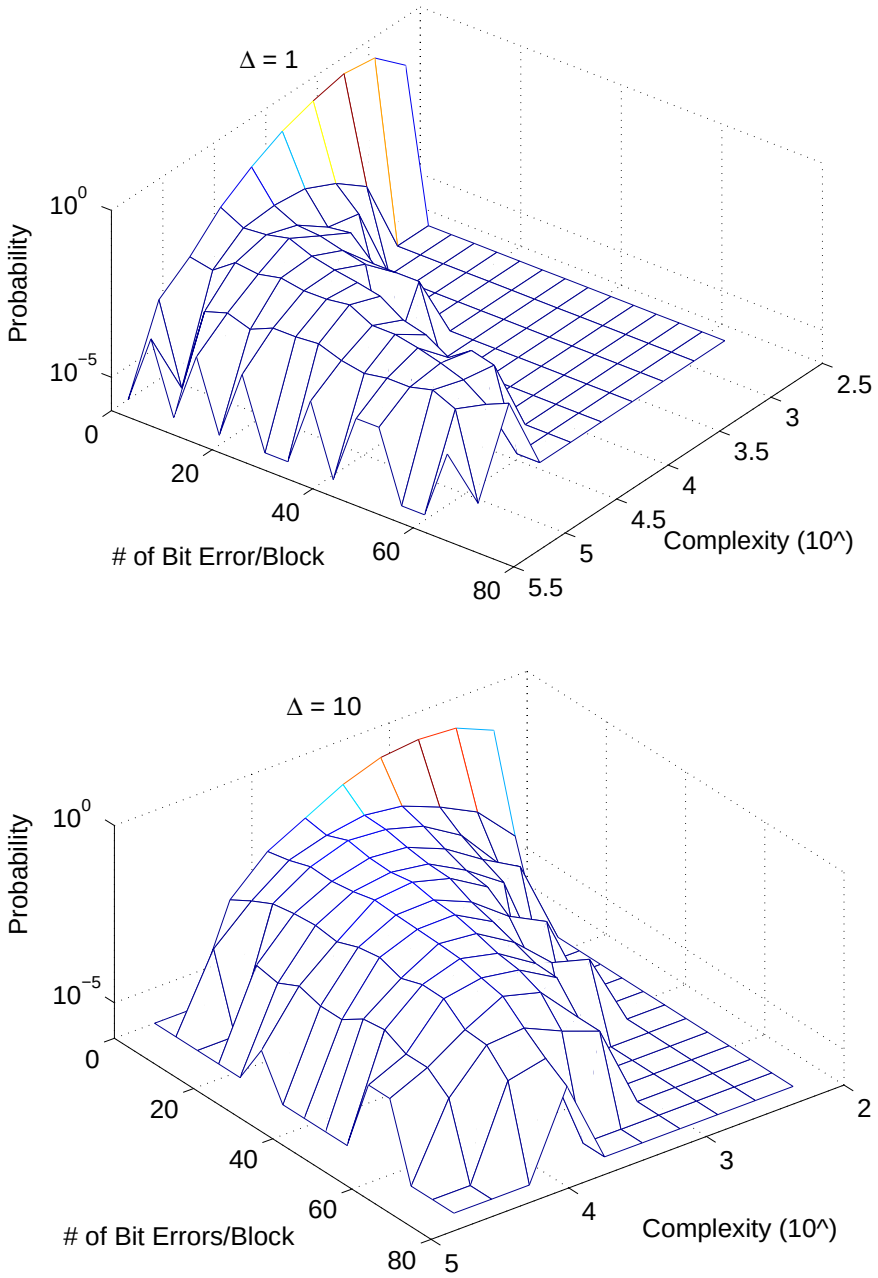


Figure 4.9: Histograms of (b,c) pairs, for SNR= 3dB, and Δ=1 and 10.

decoder decodes blocks rather than bits, $R_p(t)$ does not change until after another block gets decoded, when its value is incremented by the block size L. ΔT_d is the initial delay (for pre-loading the buffer with some blocks in order to avoid the underflow at the very beginning of the decoding process). Therefore, ΔT_d is the latency. It is also related to the buffer size.

The optimal control function $\Delta = f(O(t))$ allows the decoder to choose one element from a discrete set of n admissible values, $[\Delta_1, \Delta_2, ..., \Delta_n]$, so as to minimize the overall *Probability of Block Loss* (PBL).

$$PBL = \lim_{t \to \infty} \frac{[D(t) + E(t)] \times L}{N(t)}, \tag{4.11}$$

where L denotes the number of bits in a block, $D(t)$ represents the number of blocks dropped due to buffer overflow, and $E(t)$ represents the number of decoded blocks having excessive number of bit errors that are uncorrectable by the RS code. Obviously, $N(t) = R \cdot t$.

In order to achieve this goal, we need to select a sequence of Δ's that minimize *average number of lost blocks*. To this end, we shall minimize the expected number of blocks dropped due to buffer overflow, as well as the probability of blocks decoded in error. Thus for a given SNR, if we assume that the joint conditional probability mass function (PMF), $P(b, c|\Delta)$, is known, we can find a Δ^* as

$$\Delta^* = \arg \min_{\Delta_i \in [\Delta_1, \Delta_n]} \left\{ \sum_{m=1}^{M} [m \times \sum_{c \in (C_{m-1}, C_m]} P(b, c|\Delta_i)] + \sum_{b > E_t} P(b, c|\Delta_i) \right\}, \tag{4.12}$$

where M is such that $C_{max} \in (C_{M-1}, C_M]$, with C_{max} being the highest possible decoder complexity.

Eq.(4.12) selects the Δ that minimizes the sum of two terms: the first term is the *expected* number of blocks dropped due to buffer overflow, whereas the second term is the probability of the current block decoded in error.

A sequence of Δ^* based on Eq.(4.12) constitutes a *greedy* control policy statistically since it ensures that each block is decoded at a speed that causes the smallest number of blocks lost *on average* during the decoding of the current block. Thus this control policy is optimal at each decoding stage if the current buffer occupancy is given. Since the channel is memoryless, and Δ is coarsely quantized, our solution is a good *approximation* to the optimal control policy over the entire decoding process.

4.4.2 Algorithm

In practice, rather than computing Eq.(4.12) on the fly, we can pre-compute the optimal Δ^* for each possible decoder complexity $C_t \in [C_{min}, C_{max}]$, by substituting C_0 with C_t in Eq.(4.12). A lookup table T that stores the (C_t, Δ^*) pairs can be constructed in this way. Since the buffer occupancy $O(t)$ is related to C_t by Eq. (4.6), constructing the lookup table is equivalent to computing an optimal Δ^* for each $O(t)$.

Thus the buffer control algorithm can be stated as follows:

- *(step 1) To decode a block already in the buffer at time t, use Eq. (4.6) to obtain the target decoding complexity C_t based on the current buffer occupancy $O(t)$.*

- *(step 2) Find the Δ^* value to be used for decoding this block in the lookup table T. Finish decoding this block and repeat the first step for the decoding of the next block.*

4.4.3 Simulation Results

In the buffer simulation, we investigate the relation between buffer sizes and the probability of block losses for both fixed and adaptive Δ control algorithms. Δ is chosen from 1 to 14. Note that *Delta* is taken to be an integer because integers are used for Fano metrics in practical implementations.

As can be seen in Figure 4.10, when the buffer size is small, the dominant factor contributing to the block loss is the buffer overflow caused by insufficiently fast Fano decoding. Therefore, curves ($\Delta = 6, 10, 14$) exhibit lower PBL than the others ($\Delta = 1, 2, 3$). On the other hand, as the buffer size grows larger, e.g., above 40 blocks, the buffer is less likely to overflow, and the interaction between Δ and BER becomes more pronounced — a smaller Δ achieves a smaller PBL, except when $\Delta = 1$, which is worst in PBL even with large buffer size since it entails an excessive amount of complexity. Note that in Figure 4.4 the average decoding complexity becomes halved when Δ goes from 1 to 2. Therefore, there exists an optimal Δ for a given buffer size. However, no single fixed Δ control policy can achieve the lowest PBL across the whole range of buffer sizes. By contrast, our adaptive control policy consistently outperforms all other fixed Δ control policies. For example, the optimal Δ scheme can lower the PBL for $Delta = 6$, which is the closest in performance to the optimal Δ scheme, by about 30% when considering a given fixed buffer size.

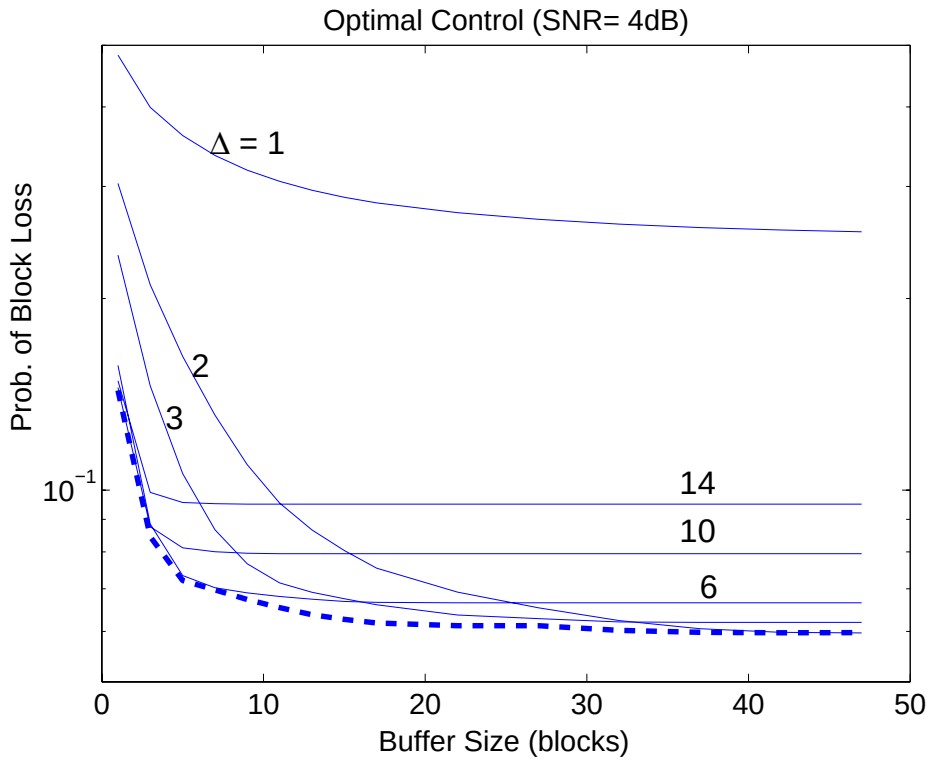


Figure 4.10: Comparison with fixed Δ control policies. The thicker, dashed curve represents the proposed Δ^* control policy based on the lookup table in Figure 4.11.

There have been numerous well-known adaptive rate control techniques employed in video source coding context, where the goal of rate control is to effectively control the quantization step size depending on the buffer occupancy [77]. These techniques can be classified into

- *Linear* [43]:

$$\Delta = (INT)\left[\Delta_{min} + (\Delta_{max} - \Delta_{min}) \times NO(t)\right], \tag{4.13}$$

where (INT) denotes the operation of rounding up to the nearest integer.

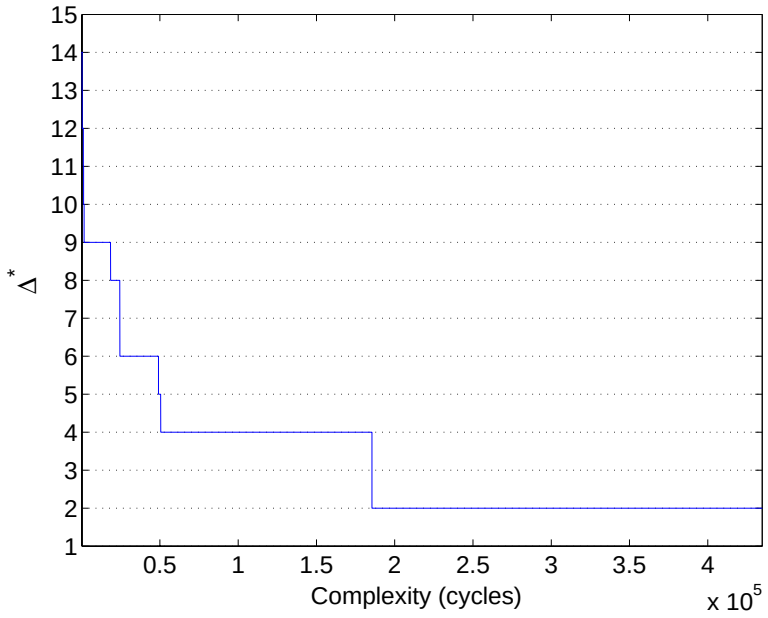


Figure 4.11: The lookup table (C_t, Δ^*) for SNR= 4dB.

- *Piecewise Linear* [40]

$$\Delta = (INT)\begin{cases} \left[\Delta_{min} + (d_1 - \Delta_{min}) \times \frac{NO(t)}{p_1}\right], & \text{if } 0 \leq NO(t) \leq p_1 \ ; \\ \left[d_1 + (d_2 - d_1) \times \frac{NO(t)-p_1}{p_2-p_1}\right], & \text{if } p_1 < NO(t) \leq p_2; \\ \left[d_2 + (\Delta_{max} - d_2) \times \frac{NO(t)-p_2}{1-p_2}\right], & \text{if } p_2 < NO(t) \leq 1, \end{cases} \tag{4.14}$$

where (d_1, p_1) and (d_2, p_2) are the turning points of adjoining pieces of straight lines. and *nonlinear*, which include

- *Sigmoidal* [45]:

$$\Delta = (INT)\begin{cases} \left[\alpha \times \left(\frac{1}{\alpha} \times NO(t)\right)^{R_b}\right], & \text{if } 0 \leq NO(t) \leq \alpha; \\ \left\{1 - (1-\alpha) \times \left[\frac{1}{1-\alpha} \times (1 - NO(t))\right]^{R_b}\right\}, & \text{if } \alpha < NO(t) \leq 1, \end{cases} \tag{4.15}$$

where α and R_b are parameters that control the shape and curvature of the curve.

- *Logarithmic* [76]:

$$\Delta = (INT)\left\{\Delta_{min} + \frac{1}{\log(a+1)} \times [\log(a \times NO(t)+1)] \times (\Delta_{max} - \Delta_{min})\right\}, \tag{4.16}$$

where a is also a curvature controlling parameter.

- *Exponential* [76]:

$$\Delta = (INT)\left[\rho(a) \times e^{a \times NO(t)} - \rho(a) + \Delta_{min}\right], \tag{4.17}$$

where

$$\rho(a) = \frac{\Delta_{max} - \Delta_{min}}{e^a - 1}, \tag{4.18}$$

and a is also a curvature controlling parameter.

The nonlinear control can adaptively choose a quantization scale value to compensate for the dramatic changes in the buffer occupancy. As shown in Figure 4.12, clearly, our proposed control policy outperforms all other well-known adaptive control policies. Although buffer size is fixed in many cases, there are situations where one wants to adapt buffer size to the application. Therefore, our method is advantageous since it always achieves the best performance for each different buffer size.

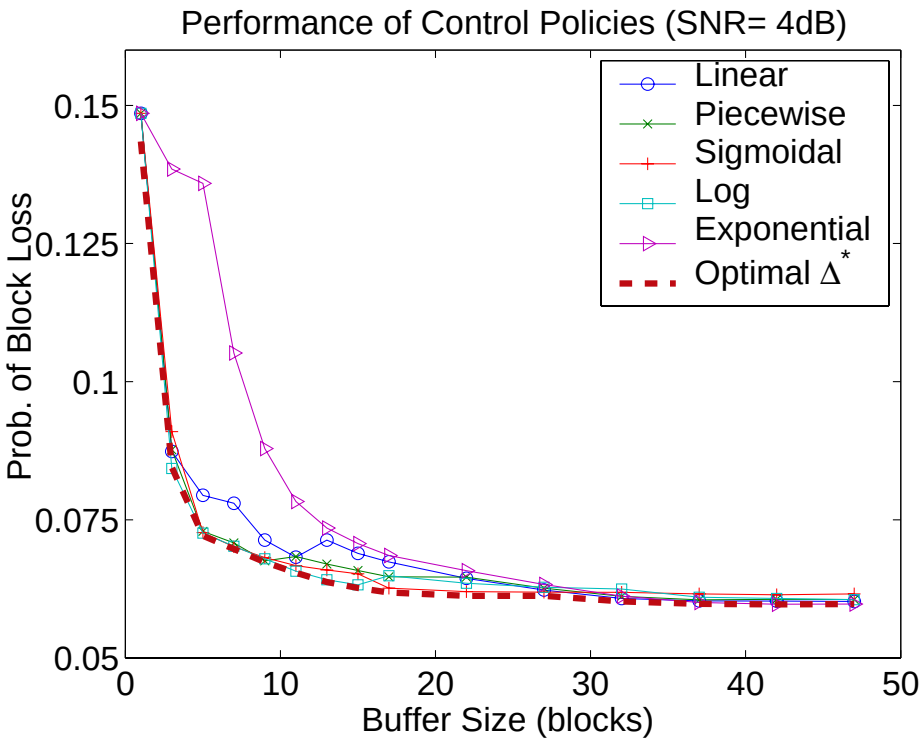


Figure 4.12: Comparison with several adaptive control policies in Figure 4.13.

4.5 Fano Decoding over Channels with Memory

A mobile wireless communication channel is characterized by time varying amplitude and phase fluctuations, in addition to the ubiquitous white Gaussian noise. Scattering causes the transmitted signal to arrive at the receiver in the form of attenuated rays with different time delays. Moreover, receiver mobility introduces a time varying channel response which superimposes a frequency modulation on the signal. In the latter case, the lower the speed of the mobile station, the higher the correlation among the fade magnitudes. A deep fade affects a number of the transmitted symbols. As a result, errors tend to occur in blocks, which is a manifestation of the channel memory. Many techniques have been developed to eliminate channel memory, e.g., interleaving.

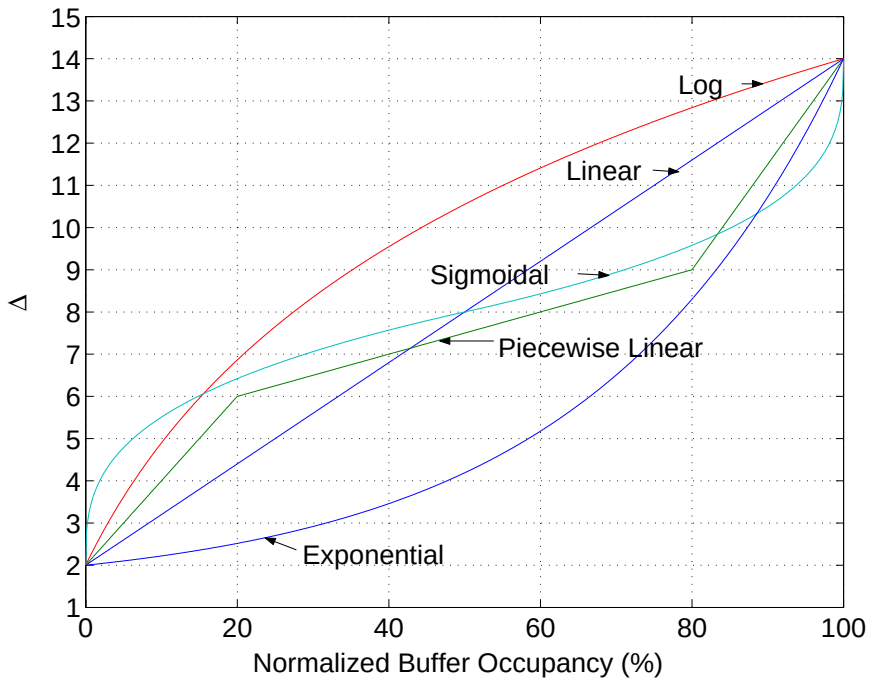


Figure 4.13: Adaptive control policies. Note that although the rounding operation is not shown in this figure, it is necessary since the actual Δ control output should be an integer.

However, interleaving introduces complexity and delay, although there may be interleaving within a block. Perfect interleaving is not always possible in practical systems, especially for real-time multimedia services. In many mobile communications systems such as GSM, the transmission of user data, short messages, digitized voice, and system control messages are organized in short blocks without or with minimal interleaving. Therefore, non-interleaved packet transmission on fading channels has become increasingly important [8]. Rather than destroying the channel memory, a newer approach is to take advantage of the memory to obtain better performance. Note that our proposed approach is not restrited to the Rayleigh fading channels, its framework can be readily applied to other types of channels with memory as well.

In our work, Δ is assumed to be fixed during the entire process of decoding a block. Hence, in order to devise a Δ control scheme that utilizes the memory property of the channel, channel memory should be manipulated using a block-oriented approach. Knowledge of the channel condition during transmission of the previous block can be used to provide a probabilistic estimate of future channel behavior, and therefore improve the performance of the computation control, as compared to a control based on memoryless channel assumptions. In the following, we use fading channels as an example to demonstrate our proposed framework, which can be applied to other possible cases like shadowing.

4.5.1 Rayleigh Fading Channel

We focus on slow, flat Rayleigh fading channels which describes the ISI-free mobile communications in urban environments. Let $x(t)$ be a low-pass equivalent of the transmitted signal. Consider a frequency-non-selective fading channel with additive noise $n(t)$. This channel can be modeled as [7, 66]:

$$y(t) = c(t)x(t) + n(t), \tag{4.19}$$

where $y(t)$ is the received signal and fading is modeled by the complex random process $c(t)$. Usually it is assumed that $n(t)$ is zero-mean complex additive white Gaussian noise (AWGN), and c(t) is a complex stationary random process. If the mean $E[c(t)] = 0$ when the LOS (line of sight) component is absent, then the envelope $|c(t)|$ has a Rayleigh distribution.

The covariance function

$$K(\tau) = E[c(t+\tau)^* c(t)] = J_0(2\pi f_D|\tau|), \tag{4.20}$$

where $J_0(.)$ is the zero-order Bessel function of the first kind [66, 80]. f_D is the maximum Doppler frequency given by

$$f_D = \frac{v f_0}{c}, \tag{4.21}$$

where v is the speed of the mobile station, c is the speed of light, and f_0 is the carrier frequency.

The Fourier transform of $K(\tau)$ yields the Doppler power spectrum, given by

$$S(f) = \begin{cases} \frac{S(0)}{\sqrt{1-\left(\frac{f}{f_D}\right)^2}} & \text{if } |f| < f_D; \\ 0 & \text{otherwise.} \end{cases} \tag{4.22}$$

Note that the correlation property of the fading process depends only on $f_D|\tau|$. When $f_D|\tau|$ is small, the process is very correlated (slow fading). This is characterized by the *maximum normalized Doppler frequency* ν_d given by

$$\nu_d \equiv f_D|\tau| \equiv \frac{v}{c} f_0 T L, \tag{4.23}$$

where $|\tau| = T \times L$ is the duration of a block of L symbols.

Let $r = \frac{1}{T}$ be the channel data symbol rate, we can then find the speed of the mobile station as

$$v = \frac{\nu_d r c}{f_0 L} \tag{4.24}$$

4.5.2 Fano Metrics for the Fading Channel

To calculate the Fano metric for a Rayleigh fading channel, we need to know the probability density function (pdf) of the received signal. Assuming that we have BPSK modulation, and the multiplicative fading channel model as described in Eq.(4.19), then the PDF of the received signal y, given that symbol "1" was transmitted ($x(t) = 1$), is [59]:

$$f_Y(y|1) = f_C(c) * f_N(n), \tag{4.25}$$

where $*$ denotes the convolution operation.

The PDF of the random variable Y has a closed-form expression (see Appendix C). The P in Eq.(4.4) for hard decision is given by

$$P = \int_{-\infty}^{0} f_Y(y) dy. \tag{4.26}$$

By the change of the integration limits, the channel transition probability for soft decisions can be obtained in a similar fashion. As a result, the Fano metric table can be tabulated for fading channels.

4.5.3 Markov Modeling for the Rayleigh Fading Channel

Our goal is to take advantage of the channel memory to obtain better performance in the computation control of the Fano decoder. The natural approach is to approximate the Rayleigh fading channel by means of a Markov model. Finite-state Markov chains (FSMC) have been proposed to model the flat-fading channel, with states representing discrete, non-overlapping intervals of a parameter chosen for the channel estimation. One popular choice is the amplitude of the received signal envelope. For example, Wang and Chang [85] demonstrate that a first-order amplitude-based FSMC is sufficient to model very slowly fading channels for any applications, based on a mutual information measure. The first-order Markovian assumption implies that, given the information of the state immediately preceding the current one, any other previous state should be independent of the current one. In the literature, the application of the FSMC to the modeling of flat fading channels can be classified into two categories: either symbol level models [3, 53, 81, 86, 85, 92], or block-level models [4, 35, 46, 73, 75, 93]. The symbol-based model is concerned with the magnitude of the signal envelope (or the instantaneous SNR, which is proportional to the the square of the envelope). However, more often, parameters (e.g., number of bit errors contained in a decoded block) that describe the behavior of the block transmission are of primary importance, and are the only information available to higher-level (than the physical layer) protocols. In [93], it is shown that the first-order Markov model is a good approximation for the block error process (possibly degenerating into an i.i.d. process for sufficiently fast fading) for a broad range of parameters, and that the relationship between the marginal error rate

and the transition probability depends only on an appropriately normalized version of the Doppler frequency.

In this work, we aim to predict the complexity (c) of decoding the future blocks, and the number of bit errors (b) contained in the blocks to be decoded. This motivates us to adopt the block-level Markov model. One is tempted to follow a similar approach in [93], where a binary (success and failure) process β_i is used to characterize the channel. A block is correctly received ($\beta_i = 0$) if it contains n_e or fewer symbol errors, and in error ($\beta_i = 1$) otherwise. Sequence of either complexity (c) or bit error number (b) samples can serve this purpose in our case. However, since both b and c are sensitive to the change of Fano metric threshold increment Δ used (this is exactly where the computation control comes from), we may have different interpretation of the channel condition if we use a different Δ. Therefore, neither b nor c, if used directly, is suitable for channel state estimation for Markov models. As an alternative, the more complex hidden Markov models (HMM) [68] have been used to characterize fading channels [9, 23, 78, 82]. In this paper, we adhere to the simple yet effective finite state Markov models. We introduce a novel use of the terminal Fano metric (we refer to it as "Fano metric" hereafter for brevity) to estimate the channel state, which is the metric value of the leaf node in the code tree, reached by the Fano algorithm when it finishes decoding a block. In the Fano algorithm, the time required to reach a leaf node of the code tree can be controlled by the Δ in use. However, the terminal Fano metric value does not change significantly. As can be seen in Figure 4.14, the Fano metric is a good parameter to estimate the channel state.

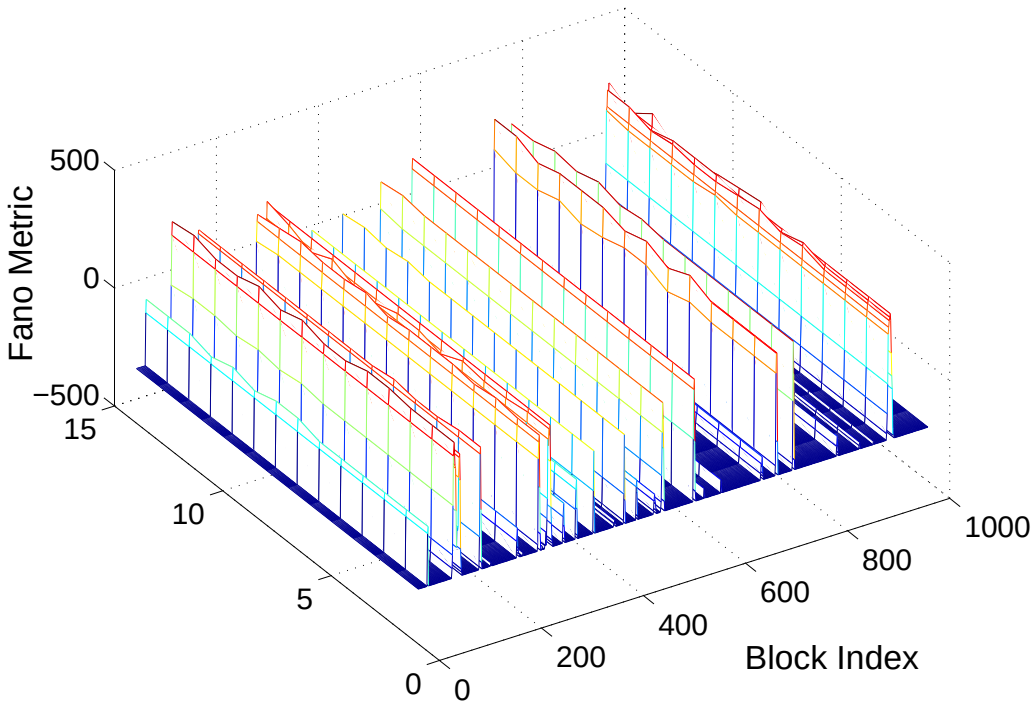


Figure 4.14: A trace of terminal Fano metrics over blocks. The Fano metric is insensitive to the Δ in use. A smaller metric value implies that the channel is less noisy.

Previous studies show that the two-state Markov model by Gilbert [25] and Elliot [15], can provide a good approximation in modeling the error process at the block (packet) level in fading channels [20, 33]. Here we use such a simple two state models, but other N-state ($N > 2$) models [34, 85, 53, 73] could also be used within our framework. In this model, the channel switches between a "good" state (G) and a "bad" state (B), respectively. Note that in [34], state G corresponds to packets transmitted correctly and state B corresponds to errors occurring. Our situation is different. Errors can occur in both states, except that the decoding complexity (c) and bit errors (b) will be small with higher probability for state G than state B since the block tends to

be less corrupted. In Figure 4.15, P_{ij} for $i, j \in \{0, 1\}$ are the transition probabilities. The transition probability matrix for this channel model then can be set up as:

$$P = \begin{bmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{bmatrix}. \tag{4.27}$$

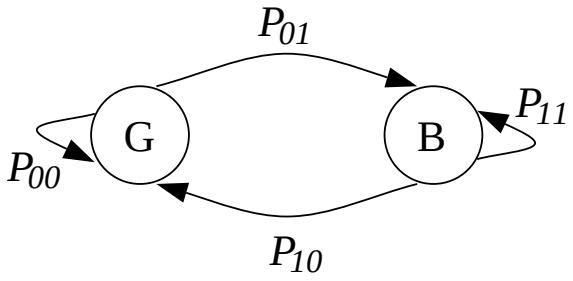


Figure 4.15: Two-state Markov channel model.

4.5.4 Partition of the Fano Metrics

Given a 2-state Markov model, we need to partition the Fano metrics into two intervals (see Figure 4.16), using a threshold Γ to separate state G and B.

The most straightforward way is to use uniform quantization, i.e., $\Gamma = 0.5 \times (Min + Max)$. In [86], the equal probability method is used so that $P(G) = P(B) = 0.5$. Another option is to obtain the threshold based on the Lloyd-Max quantizer that achieves the minimum mean-square error [24]. However, the Lloyd-Max quantizer does not necessarily yield the optimal solution to our specific problem although it seeks to represent the actual Fano metric with the least distortion. The reason lies in the fact that these methods are based solely on the univariate random variable, and do not consider the dynamics of the fading process [92]. In [75], it was found that the

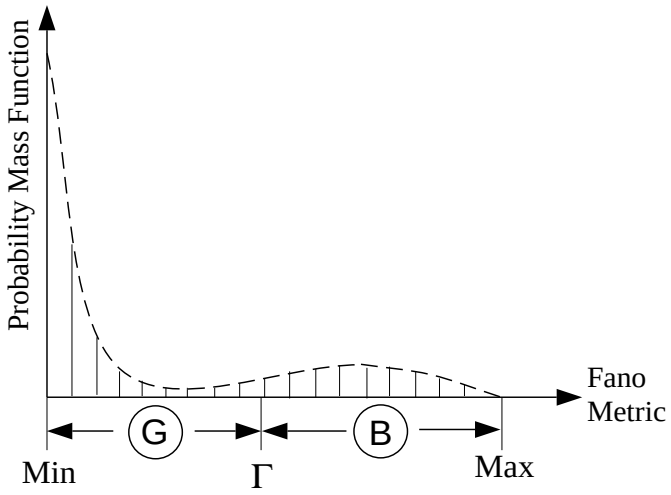


Figure 4.16: Partition (quantization) of the Fano metric m. If $m \leq \Gamma$, then $Q(m) = 0$, and the channel is in state $G(S_0)$; otherwise, $Q(m) = 1$, channel is in state $B(S_1)$. Γ is the threshold (dividing point) of the two partitions.

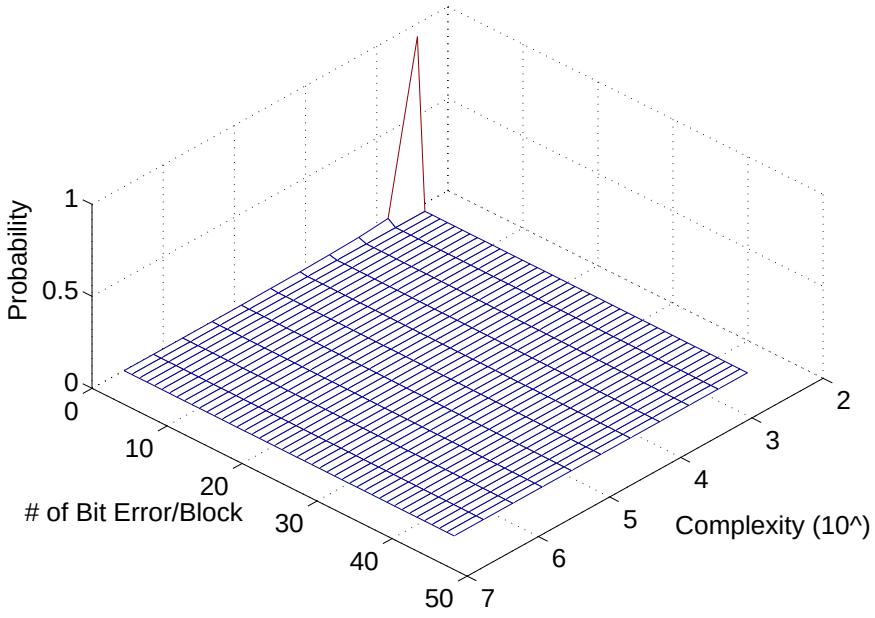
smaller fade amplitude levels are more critical in the memory evolution and therefore a finer quantization at these levels is more effective. In [92], more accurate channel models are obtained by considering also the state duration time in the partition of the instantaneous SNR. Hence, the quantization scheme should be optimized based on an objective function specific to the problem under consideration [75]. One difficulty is that the Fano metrics are not directly (one-to-one mapping) related to the complexity (c) of decoding a block, and the number of bit errors (b) contained in the block decoded. The connection is through the conditional PMF $P(b, c|\Delta, S)$, where the state $S \in \{G, B\}$ is estimated by the Fano metric (see Figure 4.17 for an example). Our goal is to utilize the memory property of the channel to further improve the computation control performance achievable based only on the memoryless assumption. Hence, the quantization of the Fano metric should be performed in a way such that both the $P(b, c|\Delta, S)$ and the Markov probability matrix are considered. We propose using the

information-theoretic *Kullback-Leibler distance* (also known as *Relative Entropy*) in determining the best threshold to partition the Fano metrics.

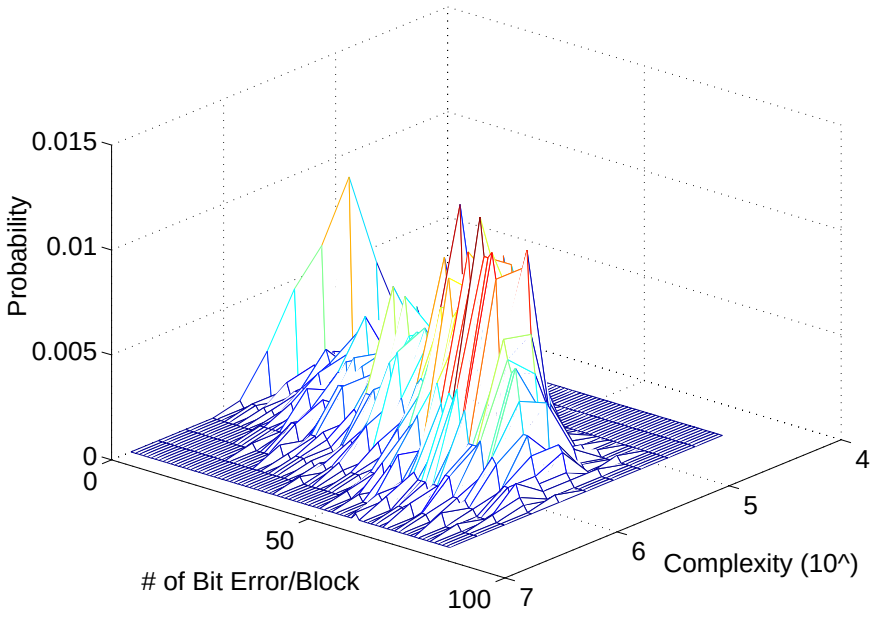
The relative entropy or Kullback-Leibler distance between two-dimensional PMF's $p(x, y)$ and $q(x, y)$ is defined as [13]

$$D(p||q) = \sum_{x \in \mathcal{X}} \sum_{y \in \mathcal{Y}} p(x, y) \log \frac{p(x, y)}{q(x, y)}, \tag{4.28}$$

$D(p||q)$ is a measure of the inefficiency of assuming that the distribution is q when the true distribution is p. Namely, let $H(p)$ be the entropy of the random variable p, then we can construct a code to describe p with average description length $H(p)$ bits. In order to describe q, we would need $H(p) + D(p||q)$ bits on the average. It can be shown that K-L distance is always non-negative and is zero if and only if $p = q$. Note that $D(p||q) \neq D(q||p)$ in general.



(a)



(b)

Figure 4.17: Conditional PMF's ($\Delta = 1$, average $SNR = 14$ dB, $\Gamma = 80$). (a) $P(b, c|\Delta, G)$. (b) $P(b, c|\Delta, B)$. Blocks are classified into two categories by the channel states. For the state G, vast majority of the blocks have $b = 0$ and minimal c values. By contrast, the state B corresponds to a large proportion of blocks (peak region) having large (b,c) values.

4.5.5 Estimation of the Markov Transition Probabilities

Let $X_i, i = 1, 2, 3, \ldots$ be a stationary and ergodic discrete time Markov process taking values in the set $\mathcal{X} = \{0, 1\}$. Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ be an observed sequence from an Markov source. Then the first order state transition probabilities of the Markov chain in Eq.(4.27) can be estimated as follows.

Let $l_{\mathbf{x}}(a, b)$ be the number of times state b is followed by state a in the sample sequence.

$$l_{\mathbf{x}}(a, b) = \sum_{i=1}^{n} I\{x_i = a, x_{i-1} = b\}. \tag{4.29}$$

Let $l_{\mathbf{x}}(b)$ be the number of times state b is seen.

$$l_{\mathbf{x}}(b) = \sum_{i=1}^{n} I\{x_i = b\}, \tag{4.30}$$

where $I(\cdot)$ is the indicator function of an assertion:

$$I\{A\} = \begin{cases} 1 & \text{if assertion A is true;} \\ 0 & \text{otherwise.} \end{cases} \tag{4.31}$$

Then we have

$$p_{ij} \equiv p_{\mathbf{x}}(a = j | b = i) = \frac{l_{\mathbf{x}}(a, b)}{l_{\mathbf{x}}(b)}. \tag{4.32}$$

4.6 Control Algorithms for Memory Channels

Our goal is the same as stated for the case of memoryless channel in Section 4.4, i.e., to minimize the overall *Probability of Block Loss* (PBL). Prediction algorithms are introduced here to further improve the control performance. In order to achieve this goal, we need to select a sequence of Δ's that minimize the additive cost — *average number of lost blocks*. For a given average SNR of the fading channel, if the joint conditional probability mass function (PMF), $P(b, c|\Delta)$, is given, many results in Section 4.4 can be re-used, except that we are now working on new PMF's that are shaped by the Markov transition probability matrix capturing the memory property of the channel. We propose the following algorithms for one-block and two-block ahead prediction.

4.6.1 One-Block Ahead Prediction

The idea here is to predict the (b, c) values associated with decoding the future block based on the channel state k of the current block. To this end, the conditional PMF's $P(b, c|\Delta)$ on which the optimization is carried on for the future block should take into account two more factors: 1) the current state of the channel as estimated by the Fano metric after decoding of the current block; 2) the Markov transition probability matrix P in Eq.(4.27). As illustrated in Figure 4.18, we now have two conditional PMF's as given by

$$P_k(b, c|\Delta_i) = \sum_{j=0}^{1} \left[P(b, c|\Delta_i, S_j) \times P_{kj}\right], \tag{4.33}$$

where $k \in \{0, 1\}$.

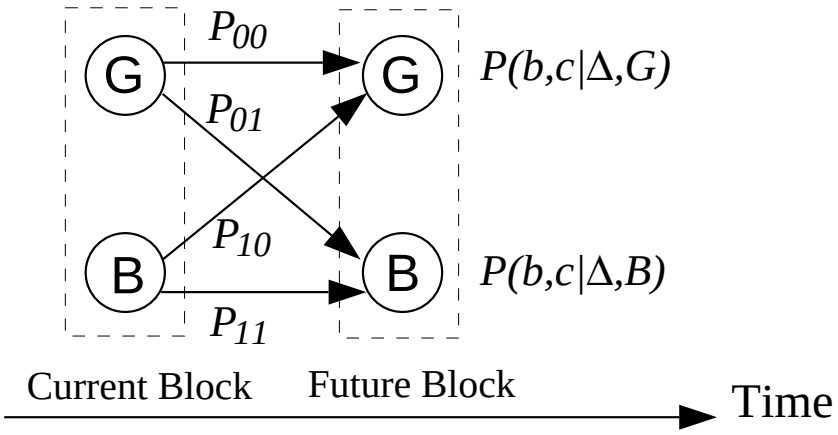


Figure 4.18: One-block ahead prediction. For example, if the channel state is G for the current block, the transition to the future block can either be "GG" or "GB". The conditional PMF, given that the current state is G, is $P_0(b, c|\Delta) = P(b, c|\Delta, G) \times P_{00} + P(b, c|\Delta, B) \times P_{01}$.

For the decoding of the next block, we choose Δ_k^* that satisfies

$$\Delta_k^* = \arg \min_{\Delta_i \in [\Delta_1, \Delta_n]} \left\{ \sum_{m=1}^{M} [m \times \sum_{c \in (C_{m-1}, C_m]} P_k(b, c|\Delta_i)] + \sum_{b > E_t} P_k(b, c|\Delta_i) \right\}, \quad (4.34)$$

where $k \in \{0, 1\}$. M is such that $C_{max} \in (C_{M-1}, C_M]$, with C_{max} being the largest possible decoder complexity.

Hence, two tables (C_t, Δ_0^*) and (C_t, Δ_1^*) are formed. Previously, there is only one table (C_t, Δ^*) for the memoryless channel in Section 4.4.2.

Thus the computational control algorithm becomes:

- *(step 1) At time t, finish the decoding of the current block in time t_d. Read in the terminal Fano metric m, compare it against the decision threshold Γ and obtain the channel state $k = Q(m)$, where $k \in \{0, 1\}$.*

- *(step 2) At time $t + t_d$, decode the next block already in the buffer, use Eq.(4.6) to obtain the target decoding complexity C_t according to the current buffer occupancy $O(t + t_d)$.*

- *(step 3) Find the ${\Delta_k}^*$ in the lookup table $(C_t, {\Delta_k}^*)$. Repeat by going back to the step 1 to decode this block by using ${\Delta_k}^*$).*

Let us revisit the problem of partitioning the Fano metrics. The choice of threshold Γ will affect both the P_{ij} in Eq.(4.32), and the two conditional PMF's $P_k(b, c|\Delta_i)$ in Eq.(4.33), from which the optimal Δ_k is selected by Eq.(4.34). Intuitively, if conditional PMF's $P_k(b, c|\Delta_i)$ are not sufficiently different from the $P(b, c|\Delta_i)$ used in Eq.(4.12) for the memoryless channel, it will follow that the two lookup tables (C_t, Δ_0^*) and (C_t, Δ_1^*) will yield the same optimal Δ as the one-table algorithm in the memoryless case. This implies that we cannot take advantage of the memory property of the channel. Therefore, a reasonable objective function for choosing the partition threshold is the "distance" between the PMF $P(b, c|\Delta_i)$ and $P_k(b, c|\Delta_i)$. An optimal threshold Γ will maximize the following K-L distance (D_1):

$$D_1 = \frac{1}{2} \sum_{k=0}^{1} D\left(P_k(b, c|\Delta_i)||P(b, c|\Delta_i)\right) \tag{4.35}$$

or the distance in reversed order:

$$D_2 = \frac{1}{2}\sum_{k=0}^{1} D\left(P(b,c|\Delta_i)||P_k(b,c|\Delta_i)\right) \tag{4.36}$$

Simulation results show that maximum points of D_1 and D_2 agree very well for all Δ's (Figure 4.19).

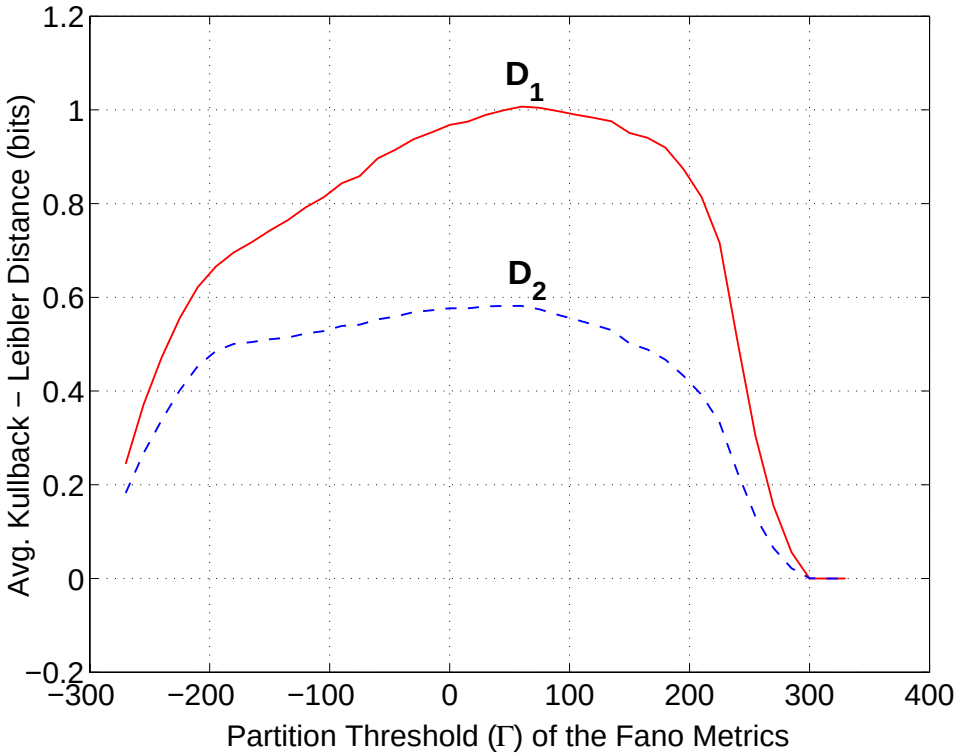


Figure 4.19: The average Kullback-Leibler distances in Eq.(4.35) and (4.36). We choose the partitioning threshold (Γ) that maximizes the K-L distances D_1 and D_2.

4.6.2 Two-Block Ahead Prediction

We can improve the prediction performance by looking further into the future. Given the Markov transition probabilities in Eq.(4.27), we can determine the expected value of the duration τ in state i:

$$\bar{\tau_i} = \sum_{\tau=1}^{\infty} \tau (P_{ii})^{\tau-1}(1 - P_{ii}) = \frac{1}{1 - P_{ii}}. \tag{4.37}$$

Since it is the "bad" state that will cause either the buffer overflow or blocks decoded in error, we will focus our prediction scheme on state B. For example, if we obtain a threshold $\Gamma = 80$ based on the K-L distance, then we obtain from simulation data by Eq.(4.32) a Markov matrix:

$$P = \begin{bmatrix} 0.987290 & 0.012699 \\ 0.391097 & 0.608903 \end{bmatrix}. \tag{4.38}$$

Therefore, the state B will last approximately 3 blocks ($\frac{1}{1-0.608903}$). Given the channel state of the current block, we can calculate the probabilities associated with all state transitions (see Figure 4.20) in the next two blocks $P_k(S_m S_n)$, where $k, m, n \in \{0, 1\}$, as shown in Table 4.2.

Based on the channel state k ($k \in \{0, 1\}$) of the current block, similar to Eq.(4.33), we can determine the two conditional PMF's as follows.

$$P_k(b, c|\Delta_i) = \sum_{m=0}^{1} \sum_{n=0}^{1} \left[P(b, c|\Delta_i, S_m S_n) \times P_k(S_m S_n) \right], \tag{4.39}$$

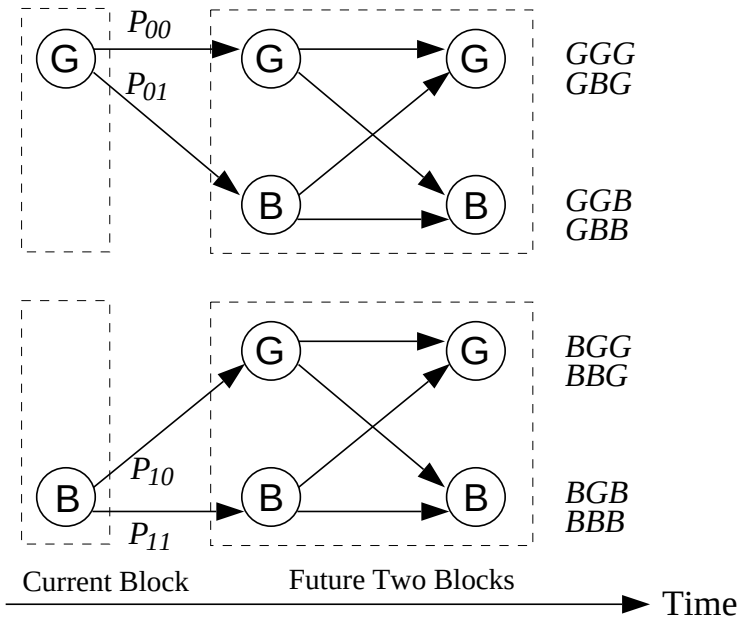


Figure 4.20: All eight possible state transitions.

For the future two blocks, if the first block has (b_1, c_1) and the second block has (b_2, c_2), then we have (b, c) for the two blocks as a whole, where $b = b_1 + b_2$ and $c = c_1 + c_2$. If we make two assumptions: 1) these two consecutive blocks will use the same Δ in the decoding (otherwise there will be $14 \times 14 = 196$, an impractical number of possible Δ combinations) ; 2) (b_1, c_1) and (b_2, c_2) are independent. By the principle "if two random variables are independent, then the density of their sum equals the convolution of their densities" [64], $P(b, c|\Delta_i, S_m S_n)$ in Eq.(4.39) can be determined as follows:

$$P(b, c|\Delta_i, S_m S_n) = P(b, c|\Delta_i, S_m) * P(b, c|\Delta_i, S_n), \tag{4.40}$$

Similarly to the one-block ahead prediction, we choose Δ_k^* that satisfies

Table 4.2: Probabilities of State Transitions

Current State	Future States	Prob	Value
G	GG	$P_0(S_0S_0)$	0.967256
	GB	$P_0(S_0S_1)$	0.020014
	BG	$P_0(S_1S_0)$	0.012442
	BB	$P_0(S_1S_1)$	0.000257
B	GG	$P_1(S_0S_0)$	0.244152
	GB	$P_1(S_0S_1)$	0.146947
	BG	$P_1(S_1S_0)$	0.380118
	BB	$P_1(S_1S_1)$	0.228780

$$\Delta_k^* = \arg \min_{\Delta_i \in [\Delta_1, \Delta_n]} \left\{ \sum_{m=1}^{2M} [m \times \sum_{c \in (C_{m-1}, C_m]} P_k(b, c|\Delta_i)] + \sum_{b > E_t} P_k(b, c|\Delta_i) \right\}, \quad (4.41)$$

Eq.(4.41) is the same as Eq.(4.34), except that C_{max} is due to the convolution operation. Likewise, two tables (C_t, Δ_0^*) and (C_t, Δ_1^*) can be constructed off-line. Note that although we use the two blocks ahead in the attempt to determine the best Δ, we use this Δ_k for the actual decoding of the next one block only. Therefore, the computational control algorithm remains the same as that for the one-block ahead case, although the lookup tables are different.

4.6.3 Simulation Results

Each simulation involves 100,000 blocks. Block size is 270 bits. The Fano metrics are partitioned by using the threshold that maximizes the Kullback-Leibler distance. Simulations show that if the normalized Doppler frequency $\nu_d > 0.027$, then we cannot

do better in terms of PBL than treating the traces of (b, c) values of each decoded block as uncorrelated sequence. This fact is evidenced by the two tables $(C_t, {\Delta_0}^*)$ and $(C_t, {\Delta_1}^*)$, which are generated by the prediction algorithms in Section 4.6.1. They are almost indistinguishable from the one table (C_t, Δ^*) in Section 4.4 as if the channel were memoryless. This observation confirms the study carried out in [93] regarding the relation between the correlation of the block error process and the normalized Doppler frequencies. If $r = 100$ kb/s is taken as a reference value for the channel data rate, the carrier frequency $f_0 = 900$ MHz, then from Eq.(4.24), the mobile speed $v = \frac{0.027 \times 10^5 \times 3 \times 10^8}{9 \times 10^8 \times 270} \approx 3$ m/s $= 10.8$ km/hour, which corresponds to low / medium degree of mobility. For this example, if the mobile station is moving at a higher speed, the dependence between the (b, c) values of consecutive blocks can be neglected, and channel degenerates into a memoryless one as far as computation control is concerned, hence we can fall back to the single-table algorithm in Section 4.4.

On the other hand, for slow fading ($\nu_d \leq 0.027$), there exists correlations between the (b, c) samples of the neighboring blocks. Our prediction algorithms presented in Section 4.6 can offer gains in control performance. The simulation results are illustrated in Figure 4.21. By exploiting block memory, over the entire range of buffer sizes our prediction can allow for better selection of Δ for the decoding of the next block, thus leading to lower block loss rate than the memoryless scheme based on the assumption of independent blocks. Moreover, the two-block ahead prediction scheme outperforms the one-block ahead scheme. The improvement is more obvious, albeit slight, in the middle range of buffer sizes. The reason is that if the buffer size is either very small (or

very large), two schemes will take the same largest Δ (or the smallest Δ). It is within the range of buffer sizes from 20 to 50 blocks, that the alternating use of the two Δ's are necessary (see tables in Figure 4.22). Comparing the two prediction schemes, We believe that the two-block ahead prediction can utilizes the memory property of the channel to a fuller extent and thus achieve slightly better performance.

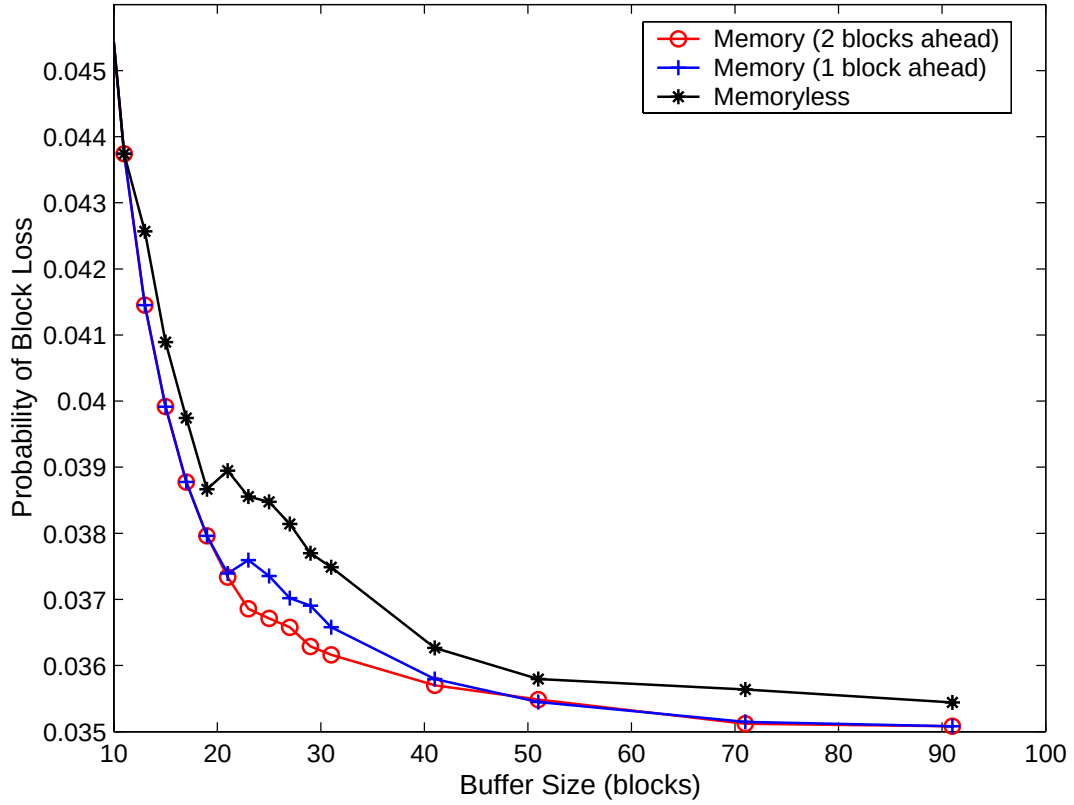
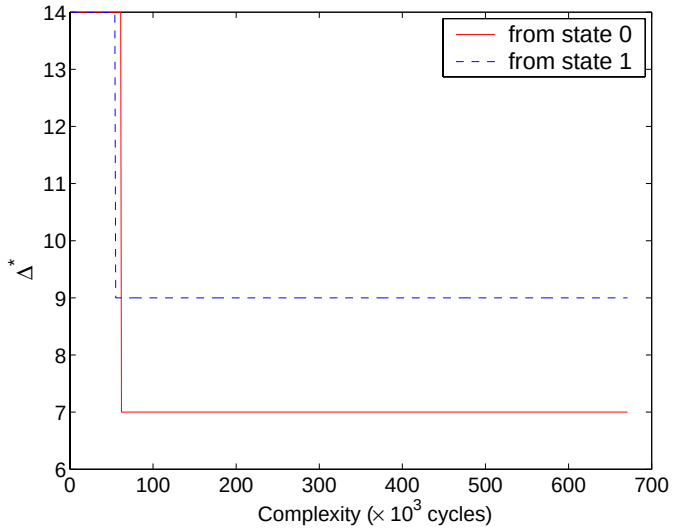
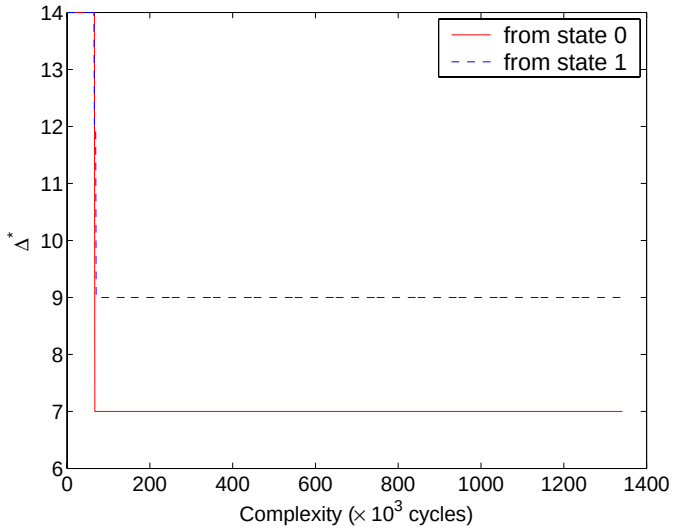


Figure 4.21: Improvement of PBL by prediction (average SNR = 14dB).The one-block and two-block ahead prediction are based on the tables (C_t, Δ_k^*), $k \in \{0, 1\}$ in Figure 4.22 (a) and (b), respectively. The $(*)$ is generated by treating the channel as being memoryless and use the table (C_t, Δ^*) in Figure 4.23.



(a)



(b)

Figure 4.22: Lookup Tables for average SNR = 14 dB. (a) one-block ahead prediction. (b) two-block ahead prediction. If the transition is from state 0 (solid line), then it is very likely to remain in the "good" state, therefore a small Δ can be used. The dashed line corresponds to the transition from state 1. A larger Δ should be taken to decode the "bad" block (large (b, c) values), partly in order to avoid buffer overflow.

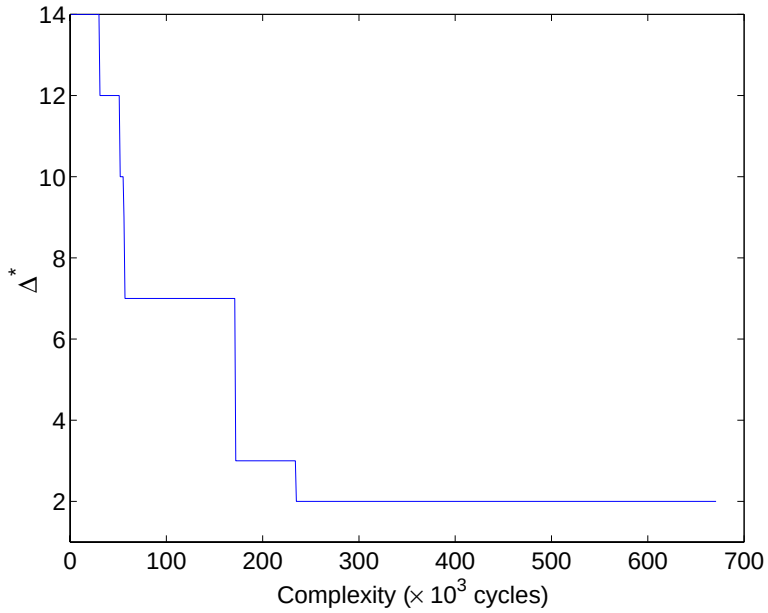


Figure 4.23: Lookup Tables for memoryless channel (average SNR = 14 dB).

4.7 Conclusions

We showed that Fano decoding algorithm is a variable complexity algorithm and the threshold increment Δ is a suitable control parameter for the useful trade-off between decoding complexity and the BER of a decoder block. It was shown that faster decoding means higher bit error rate and vice versa. We proposed a general framework of the Fano decoders with buffer to deal with the non-deterministic processing delay inherent to Fano decoding algorithms. We then formulated the computation control problem as one that seeks to minimize the overall probability of block loss, subject to the constraint of a finite buffer size. The overall probability of block loss consists of two terms. The first term describes loss due to excessive bit errors, and the second term characterizes the decoder buffer overflow. We propose a reverse mapping technique that

maps the buffer fullness status to a family of Fano decoder speed thresholds, which in turn, are used to define the probability mass on which the optimization operates. For the AWGN, memoryless channel, we obtained a Δ control policy that always selects the optimal Fano decoder parameter (Δ), at each decoding stage, in order to minimize block loss. The optimization process reduces to a simple real-time table lookup algorithm. Comparison was made against some popular adaptive rate control techniques and simulation results demonstrated the superior performance of our proposed algorithm. For the slow, flat Rayleigh fading channels with memory, we use two-state Markov models to characterize the memory of the channel. We show that the Fano metrics are good for the estimation of channel states. We develop one-block ahead and two-block ahead prediction algorithms based on the Markov models and the joint conditional of decoding complexity and BER. We demonstrate that block loss can be further lowered by taking into account the memory corresponding to low / medium mobility than when we assume that the channel is memoryless.

Bibliography

[1] Y. Arai, T. Agui, and M. Nakajima. A fast DCT-SQ scheme for images. *Trans. IEICE*, 71:1095–1097, 1988.

[2] P. A. A. Assuncao and M. Ghanbari. Buffer analysis and control in CBR video transcoding. *IEEE Trans. Circuits Syst. Video Technol.*, 10:83–92, Feb 2000.

[3] F. Babich, O. E. Kellly, and G. Lombardi. Generalized Markov modeling for flat fading. *IEEE Trans. on Communications*, 48(4):547–551, April 2000.

[4] F. Babich and G. Lombardi. A measurement based Markov model for the indoor propagation channel. In *IEEE 47th Vehicular Technology Conference*, volume 1, pages 77–81, Phoenix, AZ, May 1997.

[5] A. Bellaouar and M. I. Elmasry. *Low-power digital VLSI design: Circuits and Systems*. Kluwer Academic Publishers, 1995.

[6] I. Bharadvaj, A. Joshi, and S. Auephanwiriyakul. An active transcoding proxy to support mobile web access. In *17th IEEE Symposium on Reliable Distributed Systems (SRDS'98)*, West Lafayette, Indiana, Oct. 1998.

[7] E. Biglieri, J. Proakis, and S. Shamai (Shitz). Fading channels: information-theoretic and communications aspects. *IEEE Trans. on Information Theory*, 44(6):2619–2692, Oct 1998.

[8] H. Bischl and E. Luts. Packet error rate in the non-interleaved Rayleigh channel. *IEEE Trans. on Communications*, 43(2/3/4):1375–1382, Feb-April 1995.

[9] C. C. Chao and Y. L. Yao. Hidden Markov models for the burst error statistics of Viterbi decoding. *IEEE Transactions on Communications*, 44(12):1620–1622, Dec 1996.

[10] N. I. Cho and S. U. Lee. Fast algorithm and implementation of 2-D discrete cosine transform. *IEEE Trans. Circuits and Systems*, 38:297–305, March 1991.

[11] N. I. Cho, I. D. Yun, and S. U. Lee. A fast algorithm for 2-d DCT. In *Proc. of IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP'91)*, pages 2197–2200, April 1994.

[12] J. Choi and D. Park. A stable feedback control of the buffer state using the controlled Lagrange multiplier method. *IEEE Trans. Image Processing*, 3:546–558, Sep 1994.

[13] T. M. Cover and J. A. Thomas. *Elements of Information Theory.* John Wiley & Sons, 1991.

[14] M. Crouse and K. Ramchandran. Joint thresholding and quantizer selection for transform coding: Entropy-constrained analysis and applications to baseline JPEG. *IEEE Transactions on Image Proc.*, 6(2):285–298, Feb. 1997.

[15] E. O. Elliot. Estimates of error rates for codes on burst-noise channel. *Bell System Technical Journal*, 42:1977–1997, Sept 1963.

[16] R. M. Fano. A heuristic discussion of probabilistic decoding. *IEEE Trans. Information Theory*, IT-9:64–74, April 1963.

[17] P. Fernandez and A. Ortega. An input dependent algorithm for the inverse discrete wavelet transform. In *32nd Asilomar Conference on Signals, Systems & Computers.*, volume 1, pages 472–476, Pacific Grove, CA, Nov 1998.

[18] Flarion Technologies Inc. Vector low-density parity-check coding solution data sheet, http://www.flarion.com/ldp.

[19] A. Fox, S. D. Gribble, Y. Chawathe, and E. A. Brewer. Adapting to network and client variation using infrastructural proxies: lessons and perspectives. *IEEE Personal Communications*, pages 10–19, 1998.

[20] B. D. Fritchman. A binary channel characterization using partitioned Markov chains. *IEEE Trans. Information Theory*, IT-13:221–227, April 1967.

[21] K. Froitzheim and H. Wolf. Knowledge-based approach to JPEG acceleration. In *Proc. of IS&T/SPIE Symposium on Electronic Imaging Science and Technology*, San Jose, Feb. 1995.

[22] R. G. Gallager. *Information Theory and Reliable Communication.* Wiley, New York, 1968.

[23] J. Garcia-Frias and P. M. Crespo. Hidden Markov models for burst error characterization in indoor radio channels. *IEEE Trans. on Vehicular Technology*, 46(4):1006–1020, Nov 1997.

[24] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression.* Kluwer Academic Publishers, 1992.

[25] E. N. Gilbert. Capacity of a burst-noise channel. *Bell System Technical Journal*, 39:1253–1265, Sept 1960.

[26] B. Girod and K. Stuhlmller. A content-dependent fast DCT for low bit-rate video coding. In *Proc. of Intl' Conf. of Image Proc., ICIP'98*, Chicago, Oct. 1998.

[27] V. K. Goyal and M. Vetterli. Computation-distortion characteristics of block transform coding. In *ICASSP-97*, Munich, 1997.

[28] K. Ham, S. Jung, S. Yang, K. Lee, and K. Chung. Wireless-adaptation of WWW content over CDMA. In *IEEE International Workshop on Mobile Multimedia Communications (MoMuC'99)*, pages 368–372, 1998.

[29] R. Han, P. Bhagwat, R. LaMaire, T. Mummert, V. Perret, and J. Rubas. Dynamic adaptation in an image transcoding proxy for mobile web browsing. *IEEE Personal Communications*, pages 8–17, Dec. 1998.

[30] M. A. Haque. Two-dimensional fast cosine transform. *IEEE Trans. Acoustics, Speech, and Signal Processing*, 33(6):1532–1539, Dec 1985.

[31] B. G. Haskell, A. Puri, and A. N. Netravali. *Digital Video: An Introduction to MPEG-2*. Chapman & Hall, 1997.

[32] H. S. Hou. A fast recursive algorithm for computing the discrete cosine transform. *IEEE Trans. Acoustics, Speech, and Signal Processing*, 35:1455–1461, Oct 1987.

[33] C. Y. Hsu and A. Ortega. A Lagrangian optimization approach to rate control for delay-constrained video transmission over burst-error channels. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP'98)*, volume 5, pages 2989–2992, 1998.

[34] C. Y. Hsu, A. Ortega, and M. Khansari. Rate control for robust video transmission over wireless channels. *IEEE Journal on Sel. Areas in Comm.*, 17:756–773, May 1999.

[35] M. R. Hueda. A Markov-based model for performance evaluation in multimedia CDMA wireless transmission. In *IEEE Vehicular Technology Conference (VTC'00)*, pages 668–673, Boston, MA, Sept 2000.

[36] A. C. Hung and T. H. Y. Meng. A comparison of a fast inverse discrete cosine transform algorithms. *ACM & Springer International Journal on Multimedia Systems*, 2:204–217, 1994.

[37] A. C. Hung and T. H. Y. Meng. Statistical inverse discrete cosine transform for image compression. *Proc. of SPIE*, 2187:196–207, 1994.

[38] ICAP forum. ICAP. http://www.i-cap.org/index.cfm.

[39] I. Ismaeil, A. Docef, F. Kossentini, and R. Ward. Computation-performance control for DCT-based video coding. In *Proc. of IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP'00)*, volume 4, pages 1911 –1914, Istanbul, Turkey, June 2000.

[40] ISO/IEC JTC1/SC29/WG11 DIS. 13818-2: Information technology - generic coding of moving pictures and associated audio information: video, 1994.

[41] ITU-T (CCITT). Recommendation H.261: Video codec for audiovisual services at $p \times 64$ kbits, March 1993.

[42] ITU-T (CCITT). Recommendation H.263: Video coding for low bit rate communication, Feb 1998.

[43] ITU-T SG XV. Recommendation H.261: Line transmission of non-telephone signals video codec for audiovisual services at $p \times 64$ kbps, 1993.

[44] Independent JPEG's group. JPEG software version 6b. http://www.ijg.org.

[45] Y. Kim and H. Kim. A new buffer control strategy for image data compression. *IEEE Transactions on Consumer Electronics*, 40(4):932–937, 1994.

[46] A. Konrad, A. D. Joseph, R. Ludwig, and B. Y. Zhao. A Markov-based channel model algorithm for wireless networks. In *4th ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWiM 2001)*, pages 28–36, Rome, Italy, July 2001.

[47] K. Lengwehasatit. *Complexity-distortion tradeoffs in image and video compression.* PhD thesis, University of Southern California, April 2000.

[48] K. Lengwehasatit and A. Ortega. Distortion/decoding time tradeoffs in software DCT-based image coding. In *Proc. of ICASSP'97*, Munich, Germany, 1997.

[49] K. Lengwehasatit and A. Ortega. DCT computation based on variable complexity fast approximations. In *Proc. of Intl' Conf. on Image Proc., ICIP'98*, Chicago, Oct. 1998.

[50] K. Lengwehasatit and A. Ortega. Rate-complexity-distortion optimization for quadtree-based DCT coding. In *Prof. of Intl' Conf. on Image Processing (ICIP'00)*, Sep. 2000.

[51] S. Lin and D. J. Costello (Jr). *Error Control Coding : Fundamentals and Applications.* Prentice-Hall, 1983.

[52] C. Loeffler, A. Ligtenberg, and G. S. Moschytz. Practical fast 1-D DCT algorithms with 11 multiplications. In *Proc. of IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP'89)*, pages 988–991, May 1989.

[53] J. Lu, K. B. Letaief, and M. L. Liou. Robust video transmission over correlated mobile fading channels. *IEEE Trans. on Circuits and Systems for Video Technology*, 9(5):737–751, August 1998.

[54] L. McMillan and L. Westover. A forward-mapping realization of the inverse discrete cosine transform. In *Data Compression Conference (DCC'92)*, pages 219–228, 1992.

[55] J. L. Mitchell, W. B. Pennebaker, C. E. Fogg, and D. J. LeGall. *MPEG Video Compression Standard.* Chapman & Hall, 1996.

[56] R. Mohan, J. Smith, and C. S. Li. Adapting multimedia internet content for universal access. *IEEE Transactions on Multimedia*, pages 104–114, March 1999.

[57] E. Murata, M. Ikekawa, and I. Kuroda. Fast 2D IDCT implementation with multimedia instructions for a software MPEG2 decoder. In *Proc. of IEEE ICASSP'98*, May 1998.

[58] A. Ortega, K. Ramchandran, and M. Vetterli. Optimal trellis-based buffered compression and fast approximations. *IEEE Trans. on Image Processing*, 3:26–40, Jan. 1994.

[59] P. Orten and A. Svensson. Sequential decoding of convolutional codes for Rayleigh fading channels. *submitted to Kluwer, Wireless Personal Communications*, July 1999.

[60] W. Pan and A. Ortega. Complexity-scalable transform coding using variable complexity algorithms. In *Proceedings of Data Compression Conference (DCC 2000)*, pages 263–272, Snowbird, Utah, March 2000.

[61] W. Pan and A. Ortega. Buffer control for variable complexity fano decoders. In *IEEE Global Telecommunications Conference (Globecom'01)*, pages 176–180, San Antonio, Texas, November 2001.

[62] W. Pan, A. Ortega, I. Hajj-Ahmad, and R. Sannino. Proxy-based approaches for idct acceleration. In *IS&T and SPIE Conf. on Visual Communications and Image Processing (VCIP 2001), Proceedings of SPIE*, volume 4310, pages 625–636, San Jose, CA, January 2001.

[63] I. M. Pao and M. T. Sun. Modeling DCT coefficients for fast video encoding. *IEEE Trans. Circuits Syst. Video Technol.*, 9:608–616, June 1999.

[64] A. Papoulis. *Probability, Random Variables, and Stochastic Processes.* McGraw-Hill, 1991.

[65] W. B. Pennebaker and J. L. Mitchell. *JPEG: Still Image Data Compression Standard.* Van Nostrand Reinhold, 1992.

[66] J. G. Proakis. *Digital Communications, 4th Ed.* McGraw-Hill, 2000.

[67] J. Rabaey and M. Pedram (editors). *Low Power Design Methodologies.* Kluwer Academic Publishers, 1996.

[68] L. R. Rabiner. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, Feb 1989.

[69] K. Ramchandran and M. Vetterli. Rate-distortion optimal fast thresholding with complete JPEG/MPEG decoder compatibility. *IEEE Transactions on Image Proc.*, 3(5):700–704, Sept 1994.

[70] K. R. Rao and P. Yip. *Discrete Cosine Transform– Algorithms, Advantage, Applications.* Academic Press, 1990.

[71] T. S. Rappaport. *Wireless Communications.* Prentice Hall, 1996.

[72] J. Ribas-Corbera and S. Lei. Rate control in DCT video coding for low-delay communications. *IEEE Trans. Circuits Syst. Video Technol.*, pages 172–185, Feb. 1999.

[73] M. Rice and S. B. Wicker. Adaptive error control for slowly varying channels. *IEEE Trans. on Communications*, 42:917–926, February-April 1994.

[74] J. I. Ronda, M. Eckert, F. Jaureguizar, and N. Garcia. Rate control and bit allocation for MPEG-4. *IEEE Trans. Circuits Syst. Video Technol.*, pages 1243–1258, Dec. 1999.

[75] M. Sajadieh, F. R. Kschischang, and A. Leon-Garcia. A block memory model for correlated Rayleigh fading channels. In *IEEE International Conference on Communications (ICC '96)*, volume 1, pages 282–286, Dallas, TX, June 1996.

[76] Y. Saw, P. M. Grant, and J. M. Hannah. Reduced storage transmission buffer designs for MPEG video coder. In *Proc. of IEE Int. Conf. on Image Processing and its Applications*, pages 608–612, July 1995.

[77] Y. S. Saw. *Rate-Quality Optimized Video Coding.* Kluwer Academic Publishers, 1999.

[78] L. Shue, S. Dey, B. D. O. Anderson, and F. De Bruyne. On state-estimation of a two-state hidden Markov model with quantization. *IEEE Transactions on Signal Processing*, 49(1):202–208, Jan 2001.

[79] S. Singh, P. Thienniviboon, R. Ozdag, S. Tugsinavisute, R. Chokkalingam, P. A. Beerel, and K. M. Chugg. Algorithm and circuit co-design for a low-power sequential decoder. In *Proc. of Asilomar Conf. on Signal, Systems and Comp.*, pages 389–394, Oct. 1999.

[80] G. L. Stuber. *Principles of Mobile Communication.* Kluwer Academic Press, 1996.

[81] C. C. Tan and N. C. Beaulieu. On first-order Markov modeling for the Rayleigh fading channel. *IEEE Trans. on Communications*, 48(12):2032–2040, Dec 2000.

[82] W. Turin and R. van Nobelen. Hidden Markov modeling of flat fading channels. *IEEE Journal on Selected Areas in Communications*, 16(9):1809–1817, Dec 1998.

[83] J. Vass, S. Zhuang, J. Yao, and X. Zhuang. Efficient mobile video access in wireless environments. In *IEEE Wireless Communications and Networking Conference (WCNC'99)*, pages 354–358, 1999.

[84] A. J. Viterbi and J. K. Omura. *Principles of digital communication and coding.* McGraw-Hill, 1979.

[85] H. S. Wang and P. C. Chang. On verifying the first-order Markovian assumption for a Rayleigh fading channel model. *IEEE Trans. on Vehicular Technology*, 45(2):353–357, May 1996.

[86] H. S. Wang and N. Moayeri. Finite-state Markov channel – a useful model for radio communication channels. *IEEE Trans. on Vehicular Technology*, 44(1):163–171, Feb 1995.

[87] Z. Wang. Pruning the fast discrete cosine transform. *IEEE Trans. Communications*, 39(5):640–643, May 1991.

[88] WAP forum. WAP. http://www.wapforum.org/.

[89] L. L. Winger. Source adaptive software 2D IDCT with SIMD. In *Proc. of IEEE ICASSP'2000*, Istanbul, Turkey, June 2000.

[90] J. M. Wozencraft and I. M. Jacobs. *Principles of Communication Engineering*. Wiley, 1965.

[91] J. Zdepski, D. Raychaudhuri, and K. Joseph. Statistically based buffer control policies for constant rate transmission of compressed digital video. *IEEE Trans. Comm.*, pages 947–957, June 1991.

[92] Q. Zhang and S. A. Kassam. Finite-state Markov model for Rayleigh fading channels. *IEEE Trans. on Communications*, 47(11):1688–1692, Nov 1999.

[93] M. Zorzi, R. R. Racd, and L. B. Milstein. A Markov model for block errors on fading channels. In *7th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC'96)*, volume 3, pages 1074–1078, Taipei, Taiwan, Oct 1996.

Appendix A

2-point VCA for Gaussian Sources

A.1 Class Probabilities

This section presents the derivation of the probability of class 1, i.e., the case where the first coefficients contains sufficient energy to terminate the computation:

Consider the sample space as shown in Figure 2.3, class 1 corresponds to the shaded area with $\frac{Y_1^2}{Y_1^2+Y_2^2} \geq T$, or equivalently,

$$\left|\frac{Y_2}{Y_1}\right| \leq \sqrt{\frac{1-T}{T}} = \tan\Phi = K, \tag{A.1}$$

where K and $-K$ are the slopes of the two straight lines that partition the sample space into two classes (Figure 2.3).

Assume that $Y_1(u)$ and $Y_2(u)$ are zero-mean/independent Gaussian random variables with distributions:

$$f_{Y_1}(y_1) = \frac{1}{\sqrt{2\pi}\sigma_1} e^{-\frac{y_1^2}{2\sigma_1^2}}, \quad \text{and} \quad f_{Y_2}(y_2) = \frac{1}{\sqrt{2\pi}\sigma_2} e^{-\frac{y_2^2}{2\sigma_2^2}}. \tag{A.2}$$

Due to symmetry and independence, the probability of class 1 (shaded area in Figure 2.3) can be expressed as

$$P = \int_{-\infty}^{\infty} \int_{-Ky_1}^{Ky_1} f_{Y_1Y_2}(y_1, y_2) dy_1 dy_2 = 4 \cdot \int_0^{\infty} f_{Y_1}(y_1) dy_1 \left(\int_0^{Ky_1} f_{Y_2}(y_2) dy_2 \right). \tag{A.3}$$

To seek a close-form solution of the integration in Eq.(A.3), we introduce the polar coordinate system as

$$\begin{cases} y_1 = r\cos\theta \\ y_2 = r\sin\theta \end{cases}, \quad \text{and} \quad dx_1 dx_2 = r dr d\theta. \tag{A.4}$$

Then Eq.(A.3) can be rewritten as

$$P = \frac{2}{\sigma_1\sigma_2\pi} \int_0^{\Phi} d\theta \int_0^{\infty} r \cdot e^{-\frac{r^2}{2}\left(\frac{\cos^2\theta}{\sigma_1^2} + \frac{\sin^2\theta}{\sigma_2^2}\right)} dr \tag{A.5}$$

$$
\begin{aligned}
&= \frac{2}{\pi}\int_0^{\Phi} \frac{\sigma_1\sigma_2}{\sigma_1^2\sin^2\theta+\sigma_2^2\cos^2\theta}d\theta \\
&= \frac{2}{\pi}\arctan\left(\frac{\sigma_1}{\sigma_2}\tan\Phi\right),
\end{aligned}
$$

which is equivalent to Eq.(2.10).

A.2 Class Variances

In a two-point VCA, the energy test classifies the output Y_1 and Y_2 into two classes. Let Y_{11} and Y_{12} denote first and second coefficients that fall into the first class, similarly, Y_{21} and Y_{22} first and second coefficients categorized into the second class. Y_{ij} $(i=1,2; j=1,2.)$ are random variables, our objective is to establish the relationship between the class variance σ_{ij}^2 and the original variance σ_i^2.

From the pdf of Y_{11} in Eq.2.11 and Figure 2.7, we know that $f_{Y_{11}}(y)$ is a bi-modal, even function and $f_{Y_{11}}(0)=0$. It follows that

$$
\begin{aligned}
\sigma_{Y_{11}}^2 &= 2\int_0^{\infty} y^2 f_{Y_{11}}(y)dy \qquad (A.6)\\
&= \frac{2}{\pi\sigma_1\sigma_2 P}\int_0^{\infty} y^2 e^{-\frac{y^2}{2\sigma_1^2}}dy\left(\int_0^{Ky} e^{-\frac{x^2}{2\sigma_2^2}}dx\right)\\
&= \frac{2}{\pi\sigma_1\sigma_2 P}\int_0^{\infty} y^2\cdot R(y)\cdot e^{-\frac{y^2}{2\sigma_1^2}}dy\\
&= \frac{2}{\pi\sigma_1\sigma_2 P}\cdot I,
\end{aligned}
$$

where $R(y)=\int_0^{Ky} e^{-\frac{x^2}{2\sigma_2^2}}dx$, and $I=\int_0^{\infty} y^2\cdot R(y)\cdot e^{-\frac{y^2}{2\sigma_1^2}}dy$.

Then the integration I in Eq.(A.7) can be evaluated as

$$
\begin{aligned}
I &= \int_0^{\infty} -y\cdot R(y)\cdot\sigma_1^2\cdot d\left(e^{-\frac{y^2}{2\sigma_1^2}}\right) \qquad (A.7)\\
&= \left[-y\cdot R(y)\cdot\sigma_1^2\cdot e^{-\frac{y^2}{2\sigma_1^2}}\right]|_0^{\infty} + \int_0^{\infty} e^{-\frac{y^2}{2\sigma_1^2}}\cdot d\left[\sigma_1^2\cdot y\cdot R(y)\right].
\end{aligned}
$$

The first term in (A.8) turns out to be 0 , hence

$$
\begin{aligned}
I &= \sigma_1^2\left(\int_0^{\infty} y\cdot e^{-\frac{y^2}{2\sigma_1^2}}\frac{dR(y)}{dy}dy + \int_0^{\infty} e^{-\frac{y^2}{2\sigma_1^2}}\cdot R(y)dy\right) \qquad (A.8)\\
&= \sigma_1^2\left[\int_0^{\infty} K\cdot y\cdot e^{-\frac{K^2\sigma_1^2+\sigma_2^2}{2\sigma_1^2\sigma_2^2}y^2}dy + \int_0^{\infty} e^{-\frac{y^2}{2\sigma_1^2}}dy\left(\int_0^{Ky} e^{-\frac{x^2}{2\sigma_2^2}}dx\right)\right]
\end{aligned}
$$

$$= \sigma_1^2 \left[\frac{K\sigma_1^2\sigma_2^2}{K^2\sigma_1^2 + \sigma_2^2} + \frac{\pi\sigma_1\sigma_2}{2} P \right].$$

Substituting (A.9) into (A.7), we have

$$\sigma_{Y_{11}}^2 = \left[1 + \frac{2Kr}{P\left(K^2r^2+1\right)\pi} \right] \sigma_1^2 = (1+G)\sigma_1^2, \tag{A.9}$$

which is Eq.(2.14), where $r = \frac{\sigma_1}{\sigma_2}$, and G is given in (A.10).

$$G = \frac{K\sigma_1\sigma_2}{\left(K^2\sigma_1^2 + \sigma_2^2\right) \arctan\left(K\frac{\sigma_1}{\sigma_2}\right)}. \tag{A.10}$$

In a similar manner, we can obtain the remaining closed-form formula that describes how class variances are related to the original sample variances, as given in Eq.(2.14).

$$\begin{aligned} \sigma_{12}^2 &= (1-G)\sigma_2^2, \\ \sigma_{21}^2 &= (1-H)\sigma_1^2, \\ \sigma_{22}^2 &= (1+H)\sigma_2^2. \end{aligned} \tag{A.11}$$

Note that G is termed as the variance adjustment factor in Section 2.2.3. It can be verified in Figure A.1 that G always lies within the range of $[0, 1]$, which is the same case for the other factor H given by Eq.(2.15).

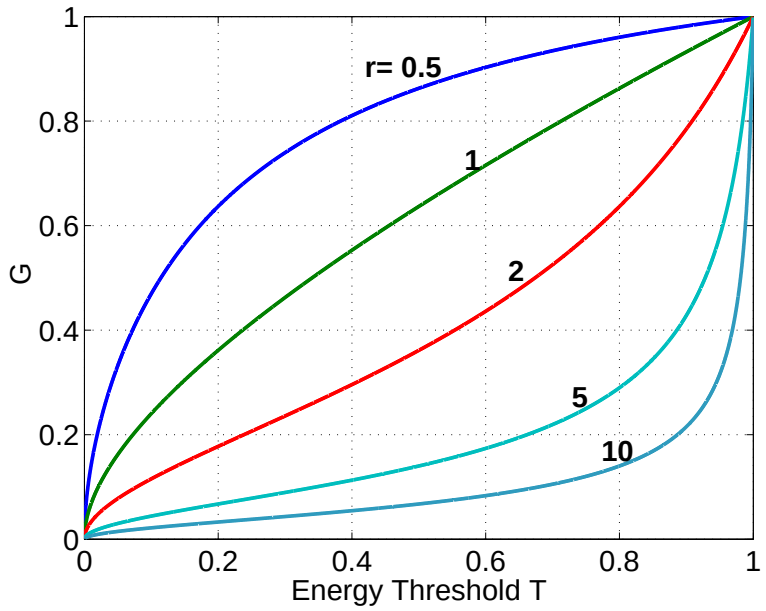


Figure A.1: The class variance adjustment factor G as a function of threshold T and class std ratio $r = \frac{\sigma_1}{\sigma_2}$.

Appendix B

Optimal Bit Allocation for N-point VCA

In Section 2.2.4, we aim at allocating optimally bits to each of the classes, i.e., to find B_k for each class. This allocation has the objective of minimizing the total *average* distortion D under a constraint on total rate B, where

$$D = \sum_{k=1}^{N} (P_k \cdot D_k) = \sum_{k=1}^{N} \left[P_k \cdot \left(k H_k \rho_k^2 2^{-2\frac{B_k}{k}} + \sum_{i=k+1}^{N} \sigma_{ki}^2 \right) \right], \tag{B.1}$$

and

$$B = \sum_{k=1}^{N} (P_k \cdot B_k). \tag{B.2}$$

B.1 Rate Distortion Relations

We shall establish the relations between rate B and distortion D.

Lagrange multiplier techniques can be used by minimizing the cost function J:

$$J = B + \lambda \cdot D = \sum_{k=1}^{N} (P_k B_k) + \lambda \cdot \left[\sum_{k=1}^{N} P_k \cdot \left(k H_k \rho_k^2 2^{-2\frac{B_k}{k}} + \sum_{i=k+1}^{N} \sigma_{ki}^2 \right) \right]. \tag{B.3}$$

Taking the N derivatives: $\partial J / \partial B_k = 0$, with $k = 1, ..., N$, we have

$$P_k + \lambda \cdot P_k \cdot k \cdot H_k \cdot \rho_k^2 \cdot 2^{-2\frac{B_k}{k}} \cdot (\ln 2) \cdot \left(\frac{-2}{k} \right) = 0. \tag{B.4}$$

Consequently the slope,

$$\lambda = \frac{2^{2\frac{B_k}{k}}}{2(\ln 2) H_k \rho_k^2} = \frac{2^{2B_1}}{2(\ln 2) H_1 \rho_1^2}, \tag{B.5}$$

Thus B_k can be expressed in B_1 as

$$B_k = k \left[B_1 + \log_2 \left(\sqrt{\frac{H_k}{H_1}} \cdot \frac{\rho_k}{\rho_1} \right) \right]. \tag{B.6}$$

Substituting Eq.(B.6) into Eq.(B.1), we obtain

$$B_1 = \frac{B-U}{P_t}, \tag{B.7}$$

where

$$P_t = \sum_{k=1}^{N} k \cdot P_k, \tag{B.8}$$

is the average number of coefficients computed for all N classes, and

$$U = \sum_{k=1}^{N} \left[(P_k \cdot k) \cdot \log_2 \left(\sqrt{\frac{H_k}{H_1}} \cdot \frac{\rho_k}{\rho_1} \right) \right]. \tag{B.9}$$

Substituting Eq.(B.6) and (B.7) into Eq.(B.1), we have,

$$D = \sigma_t^2 + P_t \cdot 2^{-2\frac{B-U}{P_t}} \cdot H_1 \cdot \rho_1^2, \tag{B.10}$$

where

$$\sigma_t^2 = \sum_{k=1}^{N} P_k \left(\sum_{i=k+1}^{N} \sigma_{ki}^2 \right). \tag{B.11}$$

Substituting Eq.(B.9) into Eq.(B.10), we obtain the R/D relation:

$$D = \sigma_t^2 + P_t \cdot \left[2^{-B} \cdot \prod_{k=1}^{N} (H_k \rho_k^2)^{\frac{kP_k}{2}} \right]^{\frac{2}{P_t}}. \tag{B.12}$$

B.2 Optimal Bit Allocation

We shall further find the optimal bit allocation $B_{k,opt}$ for $k = 1, ..., N$.

Combining Eq.(B.5) and (B.6), we have another form of B_k given by

$$B_k = \frac{k}{2} \log_2 \left[2(\ln 2)\lambda \cdot H_k \cdot \rho_k^2 \right]. \tag{B.13}$$

Considering Eq.(B.2), we then have

$$B = \sum_{k=1}^{N} \left\{ P_k \cdot \frac{k}{2} \log_2 \left[2(\ln 2)\lambda \cdot H_k \cdot \rho_k^2 \right] \right\} = \frac{1}{2} P_t \cdot \log_2 \lambda + \sum_{k=1}^{N} \left(kP_k \cdot \epsilon_k \right), \tag{B.14}$$

where

$$\epsilon_k = \frac{1}{2} \log_2 \left[2(\ln 2) \cdot H_k \cdot \rho_k^2 \right]. \tag{B.15}$$

Combining Eq.(B.13) and (B.14) and removing λ, we have the relation between B_k and B:

$$B_k = k \cdot \left[\frac{B - \sum_{k=1}^{N} (kP_k \cdot \epsilon_k)}{P_t} + \epsilon_k \right] \tag{B.16}$$

Hence, the average number of bits assigned to class k is

$$B_{k,opt} = \frac{B_k}{k} = \frac{1}{P_t}\left[B + P_t \cdot \epsilon_k - \sum_{k=1}^{N}(kP_k \cdot \epsilon_k)\right]. \tag{B.17}$$

If we assume that $\bar{B} = B/P_t$, the average number of bits per coefficient, then Eq.(B.17) can be rewritten as

$$B_{k,opt} = \bar{B} + \frac{1}{P_t} \cdot \sum_{i=1}^{N}[i \cdot P_i \cdot (\epsilon_k - \epsilon_i)]. \tag{B.18}$$

From Eq.(B.15), we have

$$\begin{aligned}\sum_{i=1}^{N}[i \cdot P_i \cdot (\epsilon_k - \epsilon_i)] &= \sum_{i=1}^{N}\left[i \cdot P_i\left(\frac{1}{2}\log_2\frac{H_k\rho_k^2}{H_i\rho_i^2}\right)\right] \\ &= \frac{1}{2}\log_2\frac{\left(H_k\rho_k^2\right)^{P_t}}{\prod_{i=1}^{N}\left(H_i\rho_i^2\right)^{iP_i}},\end{aligned} \tag{B.19}$$

which, when substituted into Eq.(B.18), finally leads to the optimal bit allocation expression as given in Section 2.2.4:

$$B_{k,opt} = \frac{B_k}{k} = \bar{B} + \frac{1}{2}\log_2\frac{H_k\rho_k^2}{\left[\prod_{i=1}^{N}(H_i\rho_i^2)^{iP_i}\right]^{\frac{1}{P_t}}}. \tag{B.20}$$

Appendix C

PDF of Sum Random Variable

Let X and Y be two random variables being Rayleigh and Gaussian distributed, respectively. They have the following pdf's:

$$f_X(x) = \frac{2x}{\Omega} e^{\frac{-x^2}{\Omega}} U(x), \tag{C.1}$$

where $U(x)$ is the unit step function, and $\Omega = E(X^2)$ is the average power. Also,

$$f_Y(y) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}} \tag{C.2}$$

Let $Z = X + Y$. If we assume that X and Y are independent, then the PDF of the sum Z is the convolution of the Rayleigh and Gaussian density functions, i.e.,

$$f_Z(z) = f_X(x) * f_Y(y) = \int_{-\infty}^{\infty} f_X(z-y) \cdot f_Y(y) dy \tag{C.3}$$

It can be shown that the $f_Z(z)$ has a closed-form expression as

$$f_Z(z) = \begin{cases} \sqrt{\frac{2}{\pi}} \frac{\sigma}{G^2} e^{\frac{-z^2}{\Omega}} + \frac{z\sqrt{\Omega}}{G^3} e^{\frac{-z^2}{G^2}} \left[erf\left(\frac{z}{G}\sqrt{\frac{\Omega}{2\sigma^2}} \right) + 1 \right], & \text{if } z \geq 0; \\ \sqrt{\frac{2}{\pi}} \frac{\sigma}{G^2} e^{\frac{-z^2}{\Omega}} + \frac{z\sqrt{\Omega}}{G^3} e^{\frac{-z^2}{G^2}} \left[erf\left(1 - \frac{-z}{G}\sqrt{\frac{\Omega}{2\sigma^2}} \right) + 1 \right], & \text{if } z < 0, \end{cases} \tag{C.4}$$

where

$$erf(x) \equiv \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt, \tag{C.5}$$

and

$$G = \sqrt{\Omega + 2\sigma^2}. \tag{C.6}$$

Fig. C.1 illustrates an example of applying Eq.(C.4).

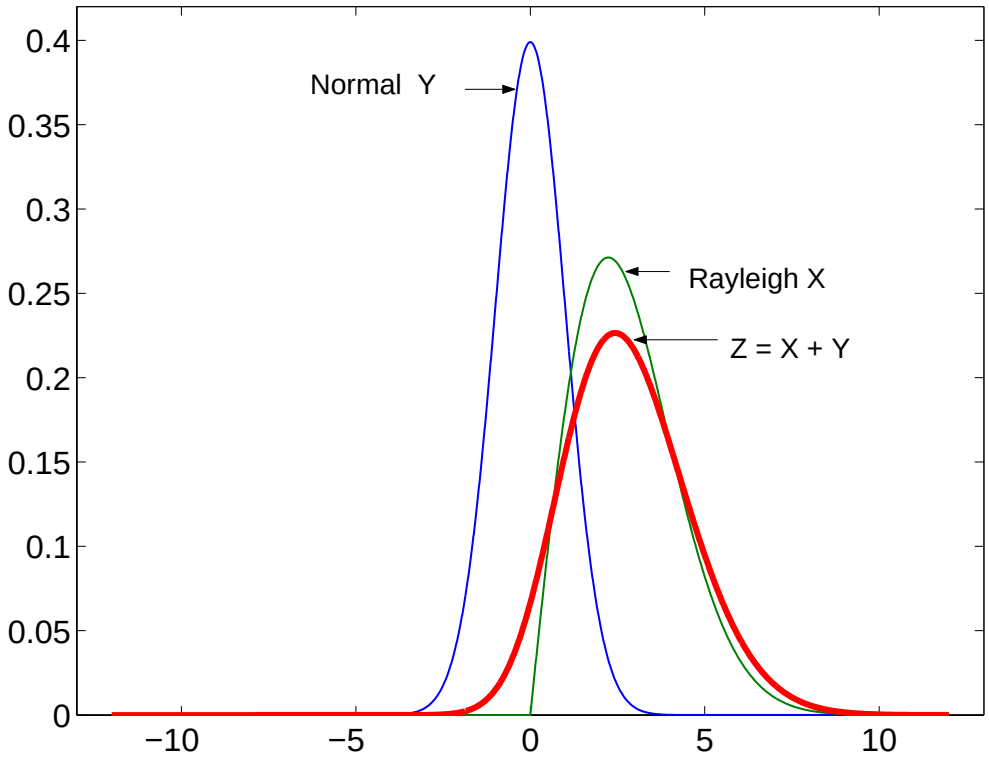


Figure C.1: PDF of the sum $Z = X + Y$. $f_X(x) = \frac{2x}{\Omega} e^{\frac{-x^2}{\Omega}}$, with $\Omega = 10$, and $f_Y(y) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}}$, with $\sigma = 1$.