ADAPTIVE METHODS AND RATE-DISTORTION OPTIMIZATION TECHNIQUES FOR EFFICIENT SOURCE CODING

by

Youngjun Yoo

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)

May 1998

Dedication

To My Parents.

Acknowledgments

First of all I would like to acknowledge the support, encouragement, and friendship from my thesis advisor Professor Antonio Ortega. Antonio has always inspired me with his insights and ideas, furthering my research to the right direction. It has been a privilege to work under his guidance, with all his extensive knowledge and great sense of humor. I would also like to thank Professors Chung-Chie Jay Kuo, Chunming Wang, Bart Kosko, and Zhen Zhang for serving on my disseration committee and for providing helpful advice and comments.

It has been my pleasure to collaborate with Professors Kannan Ramchandran of UIUC and Bin Yu of UC Berkeley. I am particularly grateful to Kannan—throughout several short yet intensive conversations, I could share his sparkling ideas and extended level of enthusiasm. I also wish to thank my M.S. thesis advisor Professor J. H. Seo of Seoul National University for showing me continued concern even during my study at Southern California.

I want to thank all my friends, colleagues, and officemates at the Signal & Image Processing Institute, especially, Mr. Sanya Mitaim, Mr. Woontack Woo, Mr. Christos Chrysafis, Mr. Chiyuan Hsu, Mr. Krisda Lengwehasatit, and Mr. Younggap Kwon for making it an enjoyable work environment. I cannot miss out the help from the staffs of Signal & Image Processing Institute and EE-Systems Department. In particular, I thank Ms. Linda Varilla and Mr. Tim Boston, who always provided instant help whenever I needed it desparately.

I also thank my fabulous friends in L.A. for letting me have so many wonderful memories. I will not forget the days we spent together at the Hollywood Bowl, the Dodger Stadium, and Venice Beach as well as those famous places in Korea Town.

The moments I shared with friends at KESO and *the Basketball Club* used to be the biggest fun on the USC campus. Special thanks goes to Mr. Il-Jun Jeong and Mr. Dong-Joon Shin who showed me true friendship.

I sincerely thank my parents for showing endless care and total support during my years at USC. Without them, I couldn't have even started my Ph.D. study in the U.S. I am truly thankful to my parents-in-law for their concern and encouragement. I also thank my sisters and sisters-in-law, who always showed a great interest in my life and career.

Most importantly I want to express my deepest thanks to my wife Jiyun for her love, support, and for putting up with me. When I could always be with her, studying together for our doctoral degrees or hanging around together to relax ourselves, I don't think that I could be more blessed.

Finally I would like to acknowledge the financial support arranged through Integrated Media Systems Center of USC. Part of this research is also funded by Texas Instruments where I am heading for the next journey.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Abstract

In this thesis we study various techniques for efficient quantization and entropy coding in several source coding applications. In particular we discuss adaptive methods and rate-distortion (R-D) optimization techniques as our vehicles to carry out the goal of efficient source coding.

We use adaptive methods to help compression of inputs such as images and video that have little known or non-stationary source statistics. We investigate new R-D optimization techniques to seek fast and optimal solutions in lossy source coding. We also consider the error robustness issue associated with entropy coding and develop techniques for robust image coding.

The specific topics of interest in this thesis are:

- Adaptive vector quantization (VQ) without side information. We employ non-parametric pdf estimation for adaptation of the scalar-vector quantizer and the trellis coded quantizer. The resulting adaptive VQ algorithms are moderate in complexity while achieving significant adaptation gain.

- Adaptive quantization of image subband data. Context-based subband data classification and parametric pdf estimation are used for adaptive image subband quantization. We also employ a forward adaptive mode to obtain robust adaptation performance. The resulting subband image coder is competitive or superior to other state-of-the-art wavelet image coders.

- Image domain coding of simple images. We develop an image domain compression algorithm for images consisting of a few pixel intensity values. We

observe that bit-plane coding with an adaptive binary arithmetic coder is effective for these simple images. We introduce a new context modeling technique to enhance the adaptive arithmetic coder efficiency.

- Fast R-D optimization algorithms. We propose a novel hybrid R-D optimization technique by combining the two most popular techniques, Lagrangian optimization and dynamic programming. The proposed technique is endowed with both the speed of the Lagrangian approach and the optimality of dynamic programming.

- Error robust compression of still images. We develop a JPEG-based image coder that transmits the number of coded bits for each group of DCT blocks as resynchronization side information. We show this approach to be useful compared to the resynchronization scheme found in baseline JPEG.

Chapter 1

Introduction

Recent years have witnessed increasing exchange of data over communication networks. As data traffic outgrows the capacity of communication channels, compact representation of data is essential for efficient use of available bandwidth for data transmission or storage. The need for efficient data representation, a common interest in many practical digital communication systems, manifests the importance of *source coding* or *signal compression* in modern communication environments.

Signal compression is usually categorized into *quantization*, as a lossy compression technique, and its lossless counterpart often called *entropy coding*. Both compression techniques have found fruitful application for various data representation needs. Entropy coding compresses data without loss of information but, in many cases, its achievable compression—bounded by the entropy of source data—is insufficient by itself for the purpose of low rate coding. By contrast, quantization can provide flexible compression for a wide range of encoding rates at the cost of accordingly introduced quantization error or information loss.

In general source coding systems, quantization is followed by entropy coding so as to achieve the desired compression rate with minimum loss of information. Transform of input data can also be considered prior to quantization in order to improve the efficiency of compression. Fig. 1.1 depicts the components of a generic source coding system. There are many well-established techniques available for each of the coding components in various coding applications. For example, in image and

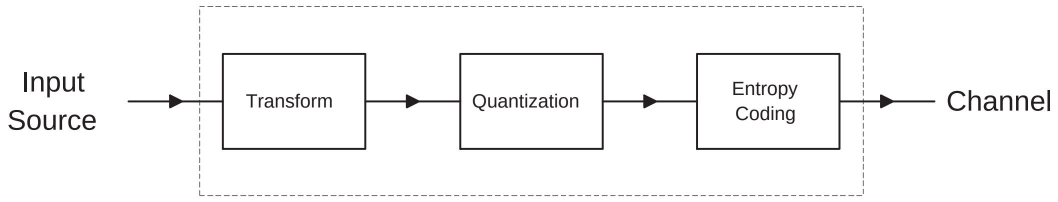


Figure 1.1: Block diagram of a generic encoding system.

video coding systems, the discrete cosine transform (DCT) and the wavelet transform are prevalently used for data transform while Huffman coding and arithmetic coding are the two most popular methods for entropy coding of quantized data [47, 34, 18, 19, 59].

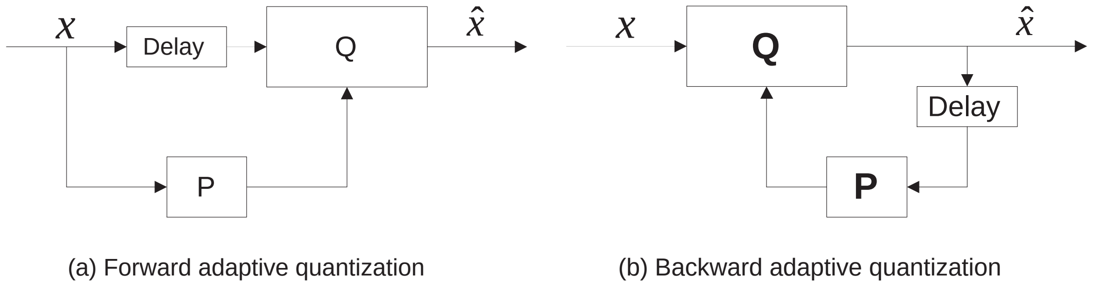
Meanwhile, it is interesting that uniform quantization, the oldest and simplest technique in the prolific literature of quantization, has been the quantization choice in many practical compression systems until recently. For example, JPEG (Joint Photographic Experts Group) [47] and MPEG (Motion Pictures Experts Group) [34], which emerged during the last decade, as, respectively, the image and video international standards, use uniform quantization to quantize the DCT block coefficients. Although the uniform quantizer does not offer the best available compression quality for these applications, it is preferred to other sophisticated quantizers mainly due to its simplicity. In other words, the high complexity of more advanced quantizers outweighs the potential performance advantage over the uniform quantizer and thus limits their use in practice.

However complicated source coding techniques become affordable thanks to the ceaseless improvement of the digital circuit speed, while the need for highly efficient digital signal compression is increasing as exemplified by high-fidelity multimedia data storage and transmission applications. Thus it is likely to see many practical compression systems equipped with sophisticated techniques in the near future. In fact, advanced quantization algorithms have been employed in standards for digital speech coding systems, e.g., Code Excited Linear Prediction (CELP) coder uses an adaptive vector quantization technique for a low-rate (4.8 kbits) speech coding standard [54]. In the case of still image compression, the emerging ISO/ITU standard JPEG 2000 [16] is considering a wavelet transform coder based on the trellis coded quantizer (W/TCQ) [41] as a possible baseline algorithm. Also, many adaptive arithmetic coders with sophisticated source modeling techniques have been proposed so as to enhance entropy coding in various image coding applications [20, 68, 65].

The interest of our study lies in the techniques to aid high quality source coding in demanding modern communications environments. We first focus on adaptive methods for quantization. Adaptive quantization is particularly effective when the input source statistics are little known or non-stationary. Thus audio, image, and video coding systems all can be improved by using adaptive quantization as they compress inputs having non-stationary characteristics. Other adaptation problems arise in entropy coding for lossless compression of a particular type of images.

Then we discuss fast algorithms for discrete rate-distortion (R-D) optimization, which can benefit various quantization and general source coding tasks where traditional optimal allocation solutions are computationally costly. Finally we investigate the problem of error robust source coding. Efficient quantization often involves entropy coding to generate variable bit rate (VBR) data being vulnerable to bit error propagation. We consider a technique to generate an inherently robust bit stream without losing the efficiency of VBR quantization.

In this chapter, we overview the basic concepts and techniques for high quality source coding. In Section 1.1 we discuss adaptive source coding. We first describe different types of adaptation algorithms, i.e., forward adaptation and backward adaptation. Then we present particular adaptation techniques used to develop our work in later chapters, e.g., adaptation via input pdf estimation and context-based adaptation. Section 1.2 introduces the budget-constrained R-D optimization formulation of source coding problems. We also introduce conventional solutions for R-D optimization along with their advantages and drawbacks. In Section 1.3 we bring up the issue of error robust source coding. This topic is relatively new in the source coding study but important as it deals with means to preserve the efficiency



x
Delay
Q
$\hat{x}$
P
(a) Forward adaptive quantization
x
Q
$\hat{x}$
Delay
P
(b) Backward adaptive quantization

Figure 1.2: Types of adaptive quantization systems. (a) Forward adaptive systems operates based on data to quantize and entails overhead. (b) Backward adaptive quantization operates based on quantized data for adaptation on the fly. No overhead is required but the decoder complexity is increased.

of VBR source coding schemes under imperfect data transmission environment. Finally, in Section 1.4, we give an overview of the work in this study and summarize the contribution.

1.1 Adaptive techniques for source coding

Most common source coding schemes rely on training sets and/or source models, which can represent the statistics of the actual input, to design the quantizer codebook or the entropy code. The performance of a source coding system using such *a priori* knowledge about the input is largely affected by the choice of the training set or the input source model, resulting in unexpected performance degradation if there is a mismatch between the actual input statistics and the design assumption. In practical situations of compressing complex data, it may be hard to have a good training set or sufficient knowledge for the input model. Thus there exists a motivation for adaptive source coding schemes which do not require any (or as little as possible) *a priori* information on the signal of interest.

We can categorize the various adaptation schemes into two broad classes: forward adaptation and backward adaptation. We will describe the difference between these two adaptation scenarios with a focus on the case of adaptive quantization. Fig. 1.2 depicts the block diagrams of the typical forward and backward adaptive quantization systems. In forward adaptive quantization, the encoder makes a decision on how to adapt the quantizer by probing the current and future inputs. Since the adaptation is based on information unavailable to the decoder, side information has to be sent to inform the decoder of the encoder's decision. As an example of forward adaptive quantization, JPEG encoders can selectively use a specific quantization table for each image. The optimized quantization table for a given image must be stored in the JPEG file header for correct dequantization while reconstructing the image.[1]

In the case of backward adaptation, quantizers are updated based on the "previously quantized" data which are available to both encoder and decoder. Although backward adaptation has the drawback of increasing the complexity not only in the encoder but also in the decoder, it has the clear advantage of avoiding the need for the adaptation overhead information in the decoder. When discussing adaptive techniques for quantization and entropy coding in our study, we are particularly interested in this *overhead-free* backward adaptive systems which we will simply call "adaptive," for convenience.

Early examples of adaptive quantization systems [14, 21] focus on the dynamic range adaptation of the uniform quantizer. The recent work of [46] has extended the adaptation to include updating of quantization thresholds and reconstruction levels, as well as dynamic range, for scalar quantizers. Assuming a quantizer design algorithm based on the pdf of input source, the basic idea to achieve adaptation is to estimate the input pdf from past quantized data. Figure 1.3 illustrates a method of input pdf estimation: the histogram of past quantized data is used to evaluate the probability mass function associated with the quantization bins, which

[1]For entropy coding, there is a similar forward adaptation option available in JPEG: the JPEG encoder can use a custom Huffman code table optimized for each input image, instead of the default Huffman table. The custom Huffman code also adds to the side information for correct entropy decoding.

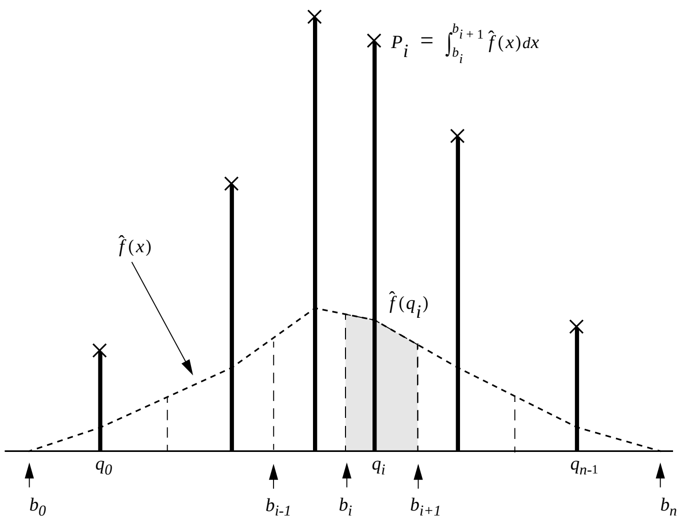


Figure 1.3: Input pdf estimation from histogram of past quantized data.

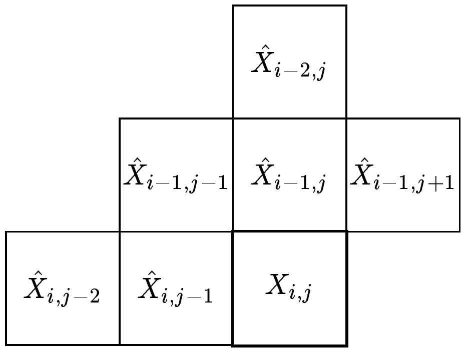
in turn provides a piecewise linear approximation of a *non-parametric* input pdf. The resulting adaptation scheme can "learn" the distribution of stationary memoryless sources and, more importantly, "detect" the short-term statistics changes of the input. Thus it performs well either for the input from an unknown stationary source or in the presence of switching modes in the source distribution.

[46] has demonstrated the usefulness of this quantizer adaptation technique by implementing adaptive scalar quantizers based on the Lloyd-Max minimum distortion design and based on the optimal entropy-constrained design. These adaptive scalar quantizers achieve good results for the input from a single Gaussian source and also for the mixed input obtained from switching Gaussian sources.

Now two questions can be raised regarding further application of this type of adaptive quantization idea:

1. Can we use this adaptation technique for more complicated quantizers, e.g., vector quantizers?

2. Can we utilize the pdf estimation idea in practical applications, e.g., adaptive quantization for compression of speech data, image data, etc.?



$$\hat{X}_{i,j} = Q(X_{i,j}|\hat{X}_{i-2,j}, \hat{X}_{i-1,j-1}, \ldots, \hat{X}_{i,j-1})$$

Figure 1.4: Context-based adaptive quantization. Quantization of the current data $X_{i,j}$ depends on the context given by the quantized values of neighboring data, $\hat{X}_{i-2,j}, \hat{X}_{i-1,j-1}, \ldots, \hat{X}_{i,j-1}$.

We will provide an answer to the first question by considering two specific vector quantizers in Chapter 2. Chapter 3, motivated by the second question, will introduce a high quality subband image compression scheme where quantizer adaptation is achieved by exploiting a different pdf estimation technique, i.e., *parametric pdf estimation* based on quantized data.

Another key tool for backward adaptive quantization in this study is the "context modeling" technique. See Fig. 1.4 for an example of context-based quantization of 2-D data (e.g., coefficients obtained by 2-D wavelet transform of an image). While a relatively large amount of previously quantized data are required for those pdf estimation techniques, only a few past quantized data would be enough for context-based adaptation. Thus context-based adaptation can be particularly useful when quantization has to deal with fast varying input statistics. In fact, context modeling has been a popular idea for efficient entropy coding, i.e., the arithmetic coder can use the context modeling technique to adaptively determine the conditional probability of the input symbol [63].

We will use a context-based method in Chapter 3 to classify the subband image data into the sets of different activity. Given multiple classes of data, we achieve quantization adaptation such that the dynamic range of the quantizer for each data class matches the estimated activity, or variance, of the class. Then, in Chapter 4, we will introduce a lossless algorithm for compression of “simple” images—images that consist of only a small number of pixel intensity levels. In this image domain algorithm, the context modeling technique is employed to enable efficient entropy coding of image bit-planes.

1.2 Rate-distortion optimization techniques

A fundamental problem in source coding is how to allocate available bits R (or the rate budget) over individual coding units to minimize the overall coding distortion D, measured by a given metric. As a tool to formulate and solve this problem, rate-distortion (R-D) optimization pervades all of source coding, both from an information-theoretic standpoint as well as for the design of practical coding systems (where the term *operational* rate-distortion is used).

While an optimal solution is always desirable in order to achieve the extreme performance of R-D formulated source coding problems, the ability to find the optimal solution is not the only concern in employing a particular optimization approach. The complexity of optimization algorithm—usually evaluated in terms of computation time and memory requirements—is often a bigger concern in many practical systems. For the particular example of Chapter 5, the codebook search for scalar-vector quantizer [30], conventional techniques for R-D optimization offer solutions which may be either optimal but too complex, or fast but far from optimal. This example suggests the need for a fast method to find optimal solutions which can improve the source coding applications in terms of the coding quality or the complexity, compared to the existing R-D optimization methods.

In what follows we discuss two popular techniques for R-D optimization in order to understand the problems of the conventional methods. We start with a mathematical formulation of our optimization problem.

1.2.1 Formulation: Budget-constrained R-D optimization

In this section we formulate budget-constrained R-D optimization for general lossy source coding problems. Suppose that we have a block of N coding units, with M different quantization choices being available for each unit. Assume that the rate and the distortion, $r_i(j)$ and $d_i(j)$, respectively, for each unit i and quantization option j are known. Our goal is to find, among all the possible quantization combinations $\mathbf{z} = (z(1), \ldots, x(N))$ where $z(i)$ denotes the quantization choice for unit i, the optimal solution $\mathbf{z}^*$ that minimizes the total distortion

$$\mathbf{z}^* = \arg\min_{\mathbf{z}} \sum_{i=1}^{N} d_i(z(i)) \tag{1.1}$$

while not exceeding the pre-determined rate budget R_T, i.e.,

$$\sum_{i=1}^{N} r_i(z^*(i)) \leq R_T. \tag{1.2}$$

This problem can be encountered for example in block-based DCT image coding where the input unit in the above formulation corresponds to a DCT block and the quantization choices correspond to different quantization scales as in MPEG. Also entropy-constrained quantization over a window of input samples can fit into this R-D optimization formulation, where each sample is the coding unit [79].

1.2.2 Lagrangian optimization

The Lagrange multiplier method is a popular choice to solve the above problem [53]. The basic idea is to introduce the Lagrangian cost

$$J = \sum_{i=1}^{N} (d_i(j) + \lambda r_i(j)). \tag{1.3}$$

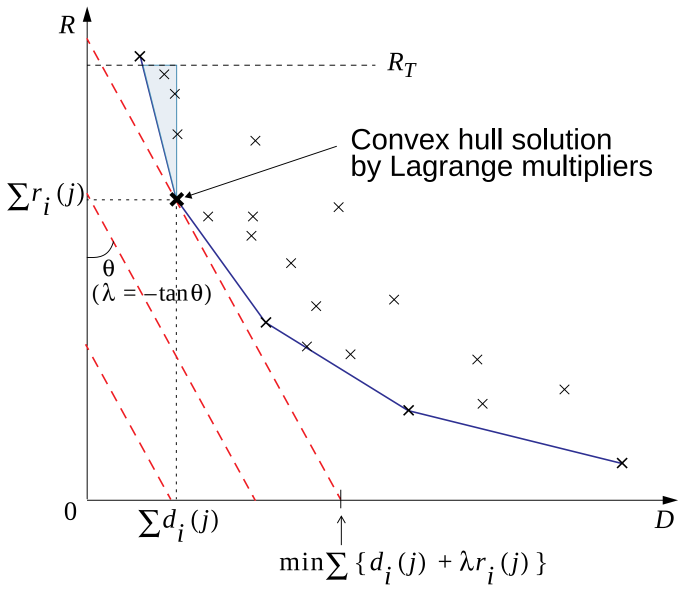

R
R_T
Convex hull solution by Lagrange multipliers
$\sum r_i(j)$
θ
$(\lambda = -\tan\theta)$
0
$\sum d_i(j)$
D
$\min \sum \{d_i(j) + \lambda r_i(j)\}$

Figure 1.5: Lagrangian solution in the discrete R-D space. Lagrangian solutions are limited to the only convex hull points in the space of operating points. Note that the shaded area contains better solutions, including the optimal solution, in the sense they have smaller distortion than the convex hull solution still within the budget.

Then a search is performed by choosing, for a given Lagrange multiplier λ, the optimal quantization choice $j(\lambda)$ that minimizes the cost $d_i(j) + \lambda r_i(j)$ of the i-th element in the coding block. The goal of the search is to find the Lagrangian multiplier λ^* for which the resulting total rate $\sum_{i=1}^{N} r_i(j(\lambda^*))$ is equal to or just below the desired budget R_T. If the budget is exactly met for λ^* then the solution is optimal.

We note that the Lagrangian optimization algorithm is very fast since each stage of the search for the optimal λ consists of greedy minimization of the cost terms on a term-by-term basis. But the Lagrange multiplier method has the limitation of reaching only the operating points on the convex hull of the R-D characteristics (see Fig. 1.5) and thus only approximates the performance of the optimal algorithms. This shortcoming could be crucial when the operating points on the convex hull are sparse so that the convex hull solution fails to approximate the optimal solution effectively.

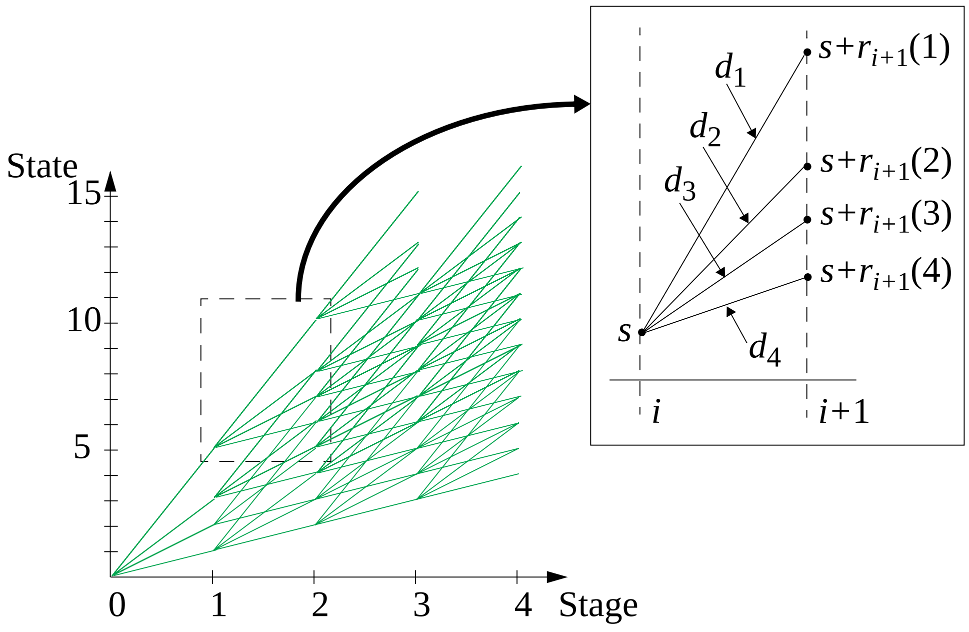


Figure 1.6: Components of the dynamic programming tree.

1.2.3 Dynamic programming

To optimally solve the R-D optimization problem, we can alternatively resort to dynamic programming (DP). The DP algorithm first builds a tree in a sequential manner to represent possible solutions. Fig. 1.6 depicts a DP tree in a 2-D plane where the x-axis corresponds to the stage (or input unit) and the y-axis represents the accumulated rate. Each stage in the tree corresponds to one of the N inputs and each tree node, or state, is associated with the accumulated distortion and rate. For example, consider the node (i,s) in the tree which represents a solution for which s bits have been used up to stage i at a total distortion cost Δ. If quantizer j is selected at stage $i+1$, it will generate a branch connecting (i,s) to $(i+1,s+r_{i+1}(j))$ and the accumulated distortion will be $\Delta + d_{i+1}(j)$. Along with the method to grow the tree, DP provides an optimal way to prune out the tree. Based on Bellman's optimality principle [1], of all paths arriving at a node, only the one with smaller distortion has any chance of being globally optimal. Thus we can eliminate, out of all paths with the same accumulated rate at a given stage, those with larger distortion.

After the growing-and-pruning procedure, the overall survival path arriving at the desired state at the final stage, which is determined by the budget constraint, yields the optimal solution of the problem.

Dynamic programming has the advantage of achieving solutions that are not reachable using the Lagrangian method. Thus it is particularly useful when there might be "gaps" in the convex hull, for example, it could achieve the top-left (i.e., optimal) point in the shaded area in Fig. 1.5. However the major drawback of DP is the computational complexity and memory requirement associated with its tree search operation, especially when there are too many quantization choices at each state of the DP tree or when the DP search block size is too large. For this reason, dynamic programming cannot be used in many practical situations. For instance, this method would be clearly impractical for real-time bit allocation over a large image or a long video sequence.

For various source coding applications formulated as the R-D optimization problem, we can employ either the dynamic programming method or the Lagrange multiplier method as standard optimization techniques depending on the main concern of the application, i.e., optimality or complexity. However, in Chapter 5, we propose a novel hybrid technique for efficient R-D optimization, which combines the speed of the Lagrangian approach with the optimality of the DP technique.

1.3 Error robust source coding

Entropy coding is an important tool to increase the efficiency of source coders. At the same time, it is an algorithm that is prone to channel or transmission errors by generating variable bit rate (VBR) output bit streams. While the entropy decoder relies on the prefix code's unique decodability for correct decoding of the VBR streams [7], channel errors result in loss of synchronization at the decoding end. Once synchronization loss has occurred, the effect of a bit error can propagate through the remaining part of the bit stream. Thus, in the worst case, all remaining bits become useless.

Forward error correction (FEC) is a conventional method which can prevent catastrophic error propagation of bit errors by adding redundant bits to the VBR streams in channel coding. Another popular approach is to insert a resynchronization marker (or codeword) at a pre-specified interval during the entropy coding. Then the decoder can recover synchronization by locating the following resynchronization maker in the bit stream after it experiences loss of synchronization. This type of approach can be found in the error propagation protection scheme of the JPEG standard.

We are interested in error protection techniques that can help entropy coding generate error robust bit streams where the redundancy is minimized. In Chapter 6 we will introduce an error robust image coding scheme based on the JPEG encoder. Our goal is to develop a JPEG-like coder that is more robust against channel errors and offers better image compression than the baseline JPEG. We note that R-D optimization techniques are used to compensate the overhead introduced for error propagation protection by optimally allocating the available bits over the DCT block.

1.4 Overview and contribution of study

To present and develop efficient source coding systems based on the techniques and problems introduced thus far, we organize this study as follows. We remark that, to keep each chapter self-contained, some important ideas and concepts will be revisited in different chapters. For an overview of the study, refer to Fig. 1.7.

In Chapter 2 a backward adaptive quantization technique is employed for two high performance vector quantizers: scalar-vector quantizer (SVQ) and trellis coded quantizer (TCQ). Adaptation algorithms for selected vector quantizers resort to non-parametric model estimation given quantized data, a proven technique for successful scalar quantizer adaptation. Through simulation, the adaptation technique is shown

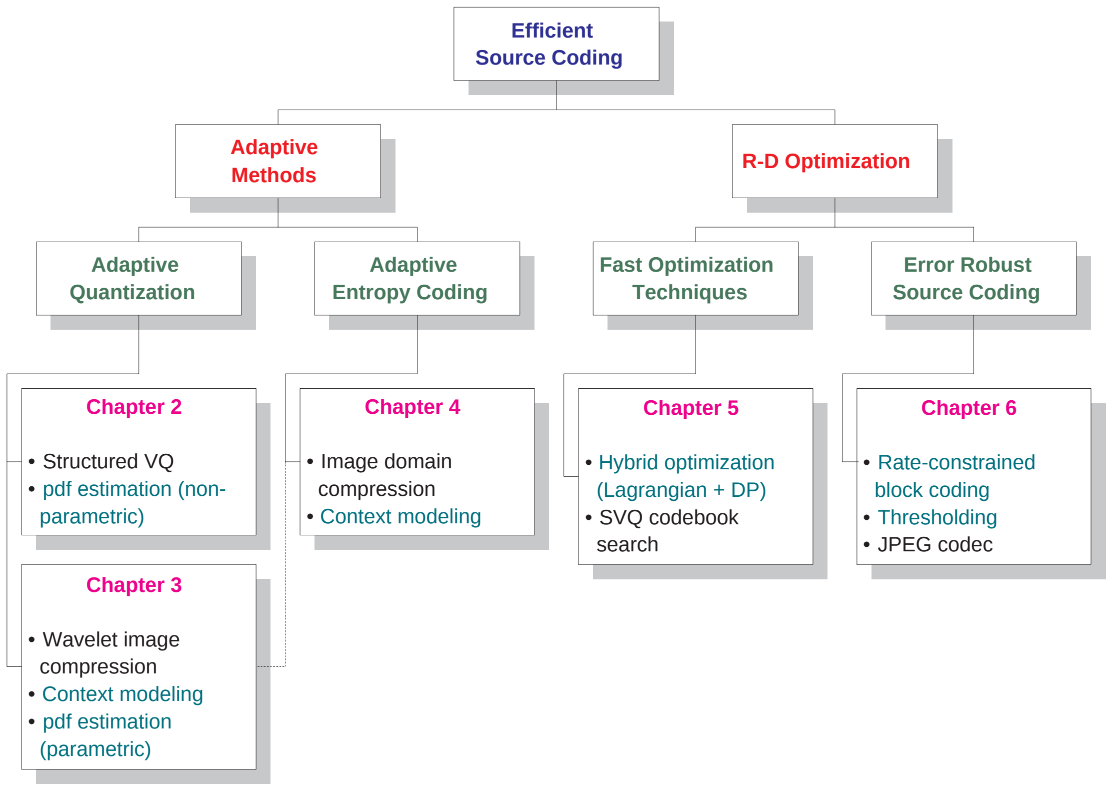


Figure 1.7: Overview of the study.

to be useful in improving robustness and stability of SVQ and TCQ without using side rate.

In this chapter we identify SVQ and TCQ as a special type of structured vector quantizers that are built on a underlying scalar quantizer (USQ). We demonstrate that adaptation of the USQ using the pdf estimation idea of [46] is useful for adaptive operation of SVQ and TCQ while we can avoid complication of treating the whole SVQ or TCQ as the adaptation unit.

Chapter 3 extends adaptive quantization to a real world application in subband image coding. Context-based classification of image subbands and quantizer adaptation using parametric pdf estimation are implemented in a backward adaptation framework. A forward adaptation technique is also exploited to help the performance of backward adaptation. The Laplacian distribution model assumed for image subband data yields efficient use of overhead necessary for forward adaptation as well as reduces the overall adaptation complexity. The resulting image coder outperforms most state-of-the-art image coders in the literature, especially at low rates, as its experimental results show.

In Chapter 4 we present another adaptive image coder that operates in the image domain rather than in the wavelet transform domain. In this chapter we are interested exclusively in a special type of images which we will call *simple*. Simple images are different from natural images in that they consist of only a limited number of pixel intensity levels. Since wavelet transform coders are not successful in compressing simple images such as scanned documents, computer generated graphics, and cartoons, a new image coding scheme is proposed to take the advantage of the characteristics of simple images.

Again the backward adaptation of the image domain compression algorithm resorts to context-based operation. The main tool used in the algorithm is bit-plane coding using an adaptive binary arithmetic coder. By developing a specialized compression algorithm for simple images, we provide a potential to establish a more generic image coding system that can handle even compound images with natural and simple regions through adaptive application of appropriate coding algorithms depending on the type of region.

Chapter 5 introduces efficient techniques for budget-constrained R-D optimization. We describe how to incorporate the Lagrange multiplier method with dynamic programming to improve the complexity of optimization. Excellent performance of the resulting optimization technique is demonstrated for SVQ codebook search, an R-D optimization problem.

The main contribution of this chapter is that we provide a clear framework for discrete R-D optimization based on a novel hybrid approach. The proposed framework can also be employed in more general class of finite resource allocation optimization problems, offering a practical means to find optimal solutions for complex large-scale optimization problems.

In Chapter 6 we turn our attention to the robustness issue that is often encountered in practical source coding systems. We develop a scheme to improve robustness of variable-rate coding systems with the baseline JPEG codec as our testbed.

As the contribution of this chapter a new approach is proposed to attack the error propagation problem associated with the JPEG bit stream, for which the standard uses explicit resynchronization approach. Our alternative approach is based on the idea of constraining the data length for the group of DCT blocks in the JPEG encoded data. The advantage of the proposed approach stems from its intelligent overhead use to endow the JPEG bit stream with error robustness.

Chapter 2

Adaptive Quantization without Side Information Using Scalar-Vector Quantizer and Trellis Coded Quantizer[1]

2.1 Introduction

Lossless compression methods such as arithmetic coding [63] and dynamic Huffman coding [13] are well known for successful use of adaptive techniques in source coding. These adaptive coders can learn the source statistics at both encoding and decoding ends so as to adjust their encoding and decoding behavior on the fly in a synchronized manner. It is this "self-adjusting" feature that makes the adaptive methods distinct from non-adaptive systems for which encoding and decoding is fixed throughout the process.

Aiming to develop adaptive quantization algorithms similar in their characteristics to those adaptive lossless compression schemes, [46] has introduced a framework of quantizer adaptation through on-line estimation of the statistics of the input. In particular, a piecewise linear approximation of the input pdf based on past quantized data is presented in [46]. The estimated pdf is used for the encoder and the decoder

[1]For related publication to this chapter see [71].

to periodically update quantization parameters (e.g., bin sizes and reconstruction levels). A theoretical analysis of this adaptation scenario is provided in [78].

Although the framework can provide virtually any quantizers with adaptivity as long as a quantizer design based on input pdf is available, the difficulty arises from the fact that the pdf estimation should depend only on quantized data. Since quantized data represent only partial information about the input statistics, estimation based on quantized data cannot resort to conventional pdf estimation techniques. In [46], an adaptation algorithm is developed taking into account this problem and applied to two simple scalar quantizers based on the Lloyd-Max design [38, 43] and the entropy-constrained design [11, 4]. The resulting adaptive quantizers are shown to be more effective for non-stationary input as well as competitive for stationary input in comparison with their non-adaptive versions, i.e., the Lloyd-Max quantizer and the optimal entropy-constrained scalar quantizer.

To achieve further gain over adaptive scalar quantization, we may consider advanced quantization schemes, e.g., vector quantizers, in the adaptation scheme. However estimating the joint pdf of a vector source is not a straightforward generalization of the scalar case. Moreover most of the practical vector quantizer design algorithms rely on a training set rather than the joint pdf of the vector source.

The goal of this chapter is to use the adaptive quantization technique of [46] as a building block in more sophisticated quantization schemes—in particular, the scalar-vector quantizer (SVQ) [30] and the trellis coded quantizer (TCQ) [42], which are constructed based on an underlying scalar quantizer (USQ). Our motivation is to demonstrate how adaptivity *can* be added and provide good results for popular quantization techniques such as TCQ and SVQ both of which have useful properties. We will introduce the adaptive scalar-vector quantizer (ASVQ) and the adaptive trellis coded quantizer (ATCQ) where we will use the previously quantized data to update their USQ.

The SVQ introduced in [30] approximates the performance of the optimal entropy-constrained scalar quantizer (ECSQ) while quantizing the input vectors at a fixed

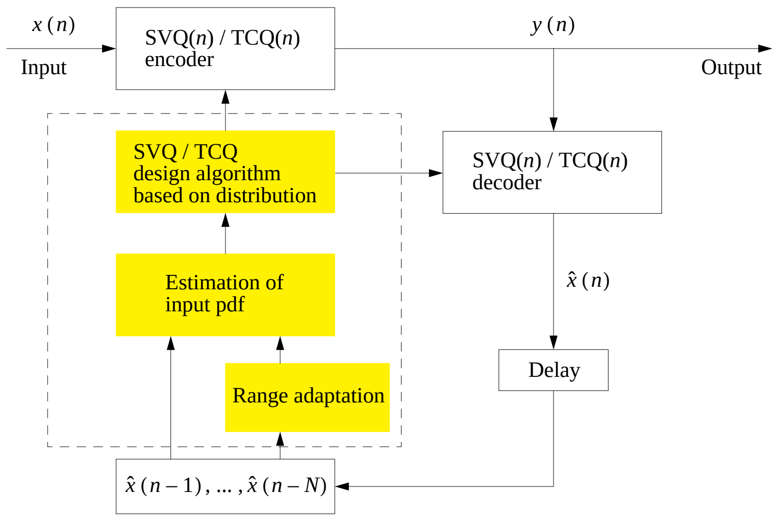

$x(n)$
Input
SVQ(n) / TCQ(n) encoder
$y(n)$
Output
SVQ / TCQ design algorithm based on distribution
SVQ(n) / TCQ(n) decoder
Estimation of input pdf
$\hat{x}(n)$
Delay
Range adaptation
$\hat{x}(n-1), \ldots, \hat{x}(n-N)$

Figure 2.1: Block diagram of backward adaptation. In the diagram, we assume SVQ or TCQ to quantize the input although, in general, we can employ any quantizers for the purpose of actual quantization of input as long as the quantized data can be used to estimate the input distribution.

rate and retaining structural and computational simplicity. The fixed rate approach is attractive to avoid the potential problems associated with transmitting variable-rate quantizer output over channels with error. The popularity of the TCQ [42] stems from its excellent SNR performance with encoding complexity which is far less than that of other vector quantizers. In fact, TCQ has demonstrated its performance in the practical application of wavelet image data quantization [41] and has been selected as part of the verification model for the emerging JPEG 2000 standard [17].

The chapter is organized as follows: In Section 2.2, we introduce the adaptive scalar quantization scheme. In Section 2.3, we explain how the adaptation scheme can combine the SVQ and the TCQ, leading to the ASVQ and the ATCQ, respectively. Experimental results with the ASVQ and the ATCQ are given in Section 2.4. We conclude this chapter in Section 2.5 with remarks on future research issues.

2.2 Backward adaptive quantization

Early works on adaptive quantization mainly concentrated on the problem of adapting the dynamic range of a uniform quantizer according to changes in the input [14, 21]. Clearly, there should be room for performance improvement if the reconstruction levels and quantization thresholds can also be adapted on the fly. This was the motivation of the algorithm of [46] and will contribute in this chapter to establish the basic adaptation structure for the selected VQ's.

Fig. 2.1 shows the block diagram of the proposed adaptive quantizer based on the SVQ or the TCQ. The adaptation algorithm is composed of two basic building blocks, namely, *model estimation* and *quantizer design* as in [46]. The key idea is to use past quantized data to estimate the input probability density function (pdf). Each time a new estimate of the pdf is obtained the parameters of the quantizer are accordingly updated. For that purpose, simplified versions of the quantizer design techniques for the SVQ and the TCQ are used at the encoder. The next input is then quantized using the updated quantizer. Depending on the required adaptation speed, we keep N past quantized outputs in the memory where N is called the adaptation window size. Note that we can either use a fixed value for N or determine N on the fly by monitoring the speed of change in the estimated input distribution. An extensive treatment on the pdf estimation based on the quantized past can be found in [46]. Here we briefly sketch the main ideas.

Given M reconstruction levels q_i, $i = 0, \ldots, M-1$, with $M-1$ decision thresholds b_i, $i = 1, \ldots, M-1$, and the N most recent quantized sample occurrences, let n_i be the number of samples which fell into the i-th bin $[b_i, b_{i+1})$ for $i = 0, \ldots, M-1$, where $b_0 = -\infty$ and $b_M = \infty$. Then we use the normalized histogram

$$\left\{ \frac{n_0}{N}, \ldots, \frac{n_{M-1}}{N} \right\}$$

to deduce the probability mass function P_i of the i-th bin such that

$$P_i = \int_{b_i}^{b_{i+1}} f(x)\, dx = \frac{n_i}{N}, \quad i = 0, \dots, M-1 \ , \tag{2.1}$$

where $f(x)$ is assumed to be the pdf of the input source. P_i's are used to determine $\hat{f}(x)$, the estimate of $f(x)$.

To simplify the problem, we assume a piecewise linear function for $\hat{f}(x)$. See also Fig. 1.3. We can determine $\hat{f}(x)$ which also needs to satisfy

$$\int_{b_i}^{b_{i+1}} \hat{f}(x)\, dx = P_i, \quad i = 0, \dots, M-1 \ , \tag{2.2}$$

by evaluating $\hat{f}(x)$ at $x_0, \dots, x_{M-1}$, where we choose

$$x_i = \frac{b_i + b_{i+1}}{2}, \quad i = 0, \dots, M-1 \ . \tag{2.3}$$

Here we need to further assume that $\hat{f}(\hat{b}_0) = \hat{f}(\hat{b}_M) = 0$ to completely specify $\hat{f}(x)$ in $[\hat{b}_0, \hat{b}_M]$, the support of $\hat{f}(x)$. $\hat{b}_0$ and $\hat{b}_M$ can be determined from the dynamic range adaptation algorithm. $\hat{f}(x)$ at $x \neq x_i, i = 0, \dots, M-1$, is then determined as the linear interpolation of the data points

$$\{\hat{f}(\hat{b}_0) = 0, \hat{f}(x_0), \hat{f}(x_1), \dots, \hat{f}(x_{M-1}), \hat{f}(\hat{b}_M) = 0\}.$$

For estimation of the dynamic range of the input distribution, we use a method different from that of [46]. We first detect a new statistical trend in the most recent quantized data by observing the empirical entropy (or sample entropy) determined from the histogram of the latest data. If the change in the empirical entropy is significant, i.e., if there is more change than a pre-specified threshold, we turn on the range adaptation algorithm of [14].

The range adaptation algorithm in [14] uses a simple decision rule to adjust the dynamic range of the uniform scalar quantizer. With appropriately defined notions of the *inner* and *outer* bins, the dynamic range is expanded if the last quantizer output belongs to one of the outer bins; otherwise, the algorithm reduces the range. The quantization levels are also adjusted proportionally to the change in the dynamic range.

Finally, note that we need a smoothness assumption on the input distribution for this input pdf estimation idea to produce a satisfactory result unless we have sufficiently many bins to detail the shape of the pdf in the histogram.

2.3 Adaptive scalar-vector quantizer and adaptive trellis coded quantizer

The adaptive quantizer of [46] achieved better performance than both the fixed Lloyd-Max quantizer and the fixed optimal ECSQ for non-stationary input sources while showing minimal performance degradation for stationary inputs. Our motivation is to apply the adaptive quantization technique to a vector quantizer (or a block-based quantizer) built on an underlying scalar quantizer (USQ) so that we redesign the quantizer using the estimate of the marginal input pdf. Hence we consider two quantizers whose structures are based on the constituent USQ: the scalar-vector quantizer (SVQ) and the trellis coded quantizer (TCQ).

For these quantizers, the input pdf estimate can be used to update the USQ. Note that both the SVQ and the TCQ quantize the input such that the vector-wise (or block-based) distortion is minimized rather than quantize in the component-wise manner according to the nearest-neighbor rule. Hence some input components can be quantized to a reconstruction level which is not the closest level in order for the overall performance. As a result, we may have a histogram of quantized data that is not exactly matched to the actual distribution of the input. This mismatch can result in a slight degradation in estimating the input pdf but we do not deal with this problem in this chapter.

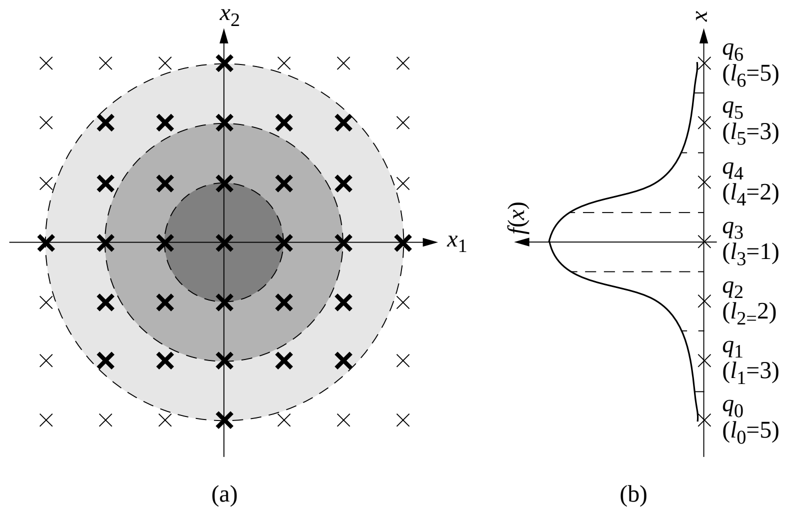


Figure 2.2: A 2-dimensional SVQ codebook derived from USQ for an i.i.d. Gaussian source. (a) SVQ codevectors are chosen from the 2-dimensional grid such that the codeword length is no greater than a pre-determined threshold L. (b) Quantization levels and the associated lengths of the USQ which determines each component of the vector grid.

2.3.1 Scalar-vector quantizer (SVQ)

The SVQ was first proposed in [30]. The motivation of the SVQ is to design a fixed-rate vector quantizer, allowing entropy-constrained quantization of the vector components, so as to be robust against transmission errors in noisy environments. While reducing the error propagation problem of the variable-rate VQ's, the SVQ can outperform fixed-rate Linde-Buzo-Gray VQ [37] with less design/encoding complexity, mainly thanks to its special codebook structure.

Fig. 2.2 illustrates a 2-dimensional SVQ codebook derived from a USQ for a memoryless Gaussian source. Each quantization level q_j is associated with a metric ℓ_j that amounts to the rounded self-information $[-\log p_j]$ of q_j, where p_j is the probability of an input component being quantized to q_j. For convenience we call ℓ_j the *length* associated with the level q_j. The SVQ codebook consists of the grid points in the shaded area of Fig. 2.2—the darker the region, the more likely the contained codevectors are used in quantization. We describe how to choose the SVQ codebook in the following.

If we use m times an n-level USQ, we have n^m vector points in an m-dimensional grid. By restricting the quantization budget to r bits per sample on average, we have chosen 2^{rm} codevectors among n^m grid points as the reproduction vectors in the SVQ codebook which can best represent the input source. Let the *codevector length* be defined as the sum of lengths of component quantization levels. Then the codevectors with smaller lengths are highly likely to be used in quantization. Hence we arrange n^m grid points in increasing order of codevector length and form the SVQ codebook with the first 2^{rm} points. This procedure gives a threshold L on the codevector length such that a grid point z is a SVQ codevector if and only if its length is no greater than L.

Therefore an SVQ is completely defined in terms of a triplet $(\mathcal{Q}, \mathcal{L}, L)$ where $\mathcal{Q} = \{q_1, \ldots, q_n\}$ is the set of quantization levels of the USQ, $\mathcal{L} = \{\ell_1, \ldots, \ell_n\}$ is the corresponding set of lengths, and L is the threshold on the codevector length for the SVQ codebook. For a detailed description and the design algorithms of SVQ, we refer to [30]. In our adaptation scheme, we need only to update and estimate $\mathcal{Q}$ and $\mathcal{L}$ using the estimated pdf.

2.3.2 Trellis coded quantizer (TCQ)

The TCQ proposed in [42] is also derived from a scalar quantizer. Motivated by the set partitioning ideas from trellis coded modulation of [58], the TCQ is designed on a USQ having twice as many levels as the quantization rate (r) allows. The quantization levels are then partitioned into 2^{m+1} subsets where $m \leq r$. We consider here a particular case of rate-2 TCQ with 4 subsets, i.e., $m = 1$.

We depict an 8-level USQ and its partition into 4 subsets in Fig. 2.3 (b) to show the structure of this particular TCQ. Note that each subset is assigned two quantization levels such that the average distance between the levels within a subset is maximized.

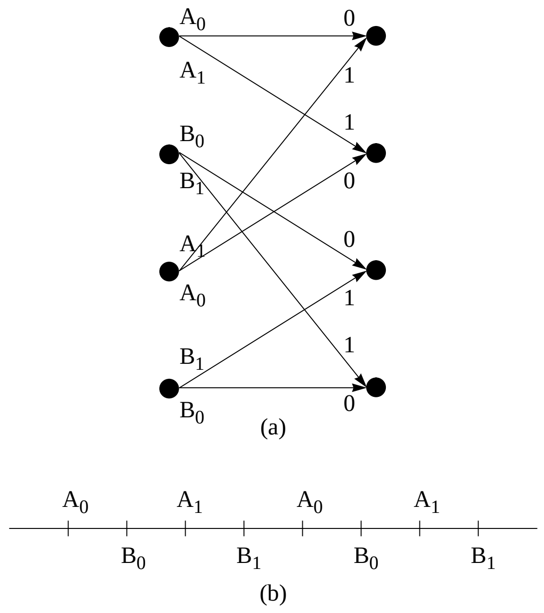


Figure 2.3: 4-state 8-level uniform trellis code. (a) The state transition diagram restricts the quantization of the current input depending on the quantization of the previous input. (b) Quantization levels of the underlying scalar quantizer are partitioned into two sets of $A = \{A_0, A_1\}$ and $B = \{B_0, B_1\}$ where each of the subsets A_i's and B_i's has two corresponding levels.

Since the quantizer rate is $r = 2$, we are in general allowed to use at most $2^r = 4$ distinct levels to quantize an input sample. For the TCQ, a finite state machine (or a state transition rule of the 4-state trellis in Fig. 2.3(a)) enables the quantizer to use one of two sets $\{A_0, A_1\}$ and $\{B_0, B_1\}$ at each trellis state, where each of two states has two subsets containing a pair of levels. Since the available set of levels depends on the state of the finite state machine, the best encoding for a given source corresponds to the optimal path through the trellis. As a result, for a given encoding rate, TCQ can have more degrees of freedom in choosing quantization levels than a fixed-rate scalar quantizer, due to the various combinations of quantization level subsets along the trellis path.

The best choice of the subsets for quantization of each input sample is made such that it globally minimizes the overall distortion of a given block of input samples using only the allowable transitions in the state transition diagram. The Viterbi algorithm is used to find the best trellis path which is a concatenation of state transitions through the block of samples.

Note that, since the SVQ and the TCQ have additional constraints on top of the USQ, inputs may not be always quantized to their nearest neighbor in the USQ, as mentioned earlier in the chapter. This will affect our pdf estimate slightly (the bin counts will not give an exact estimate of the probability of the source) but we have observed that the effect is negligible.

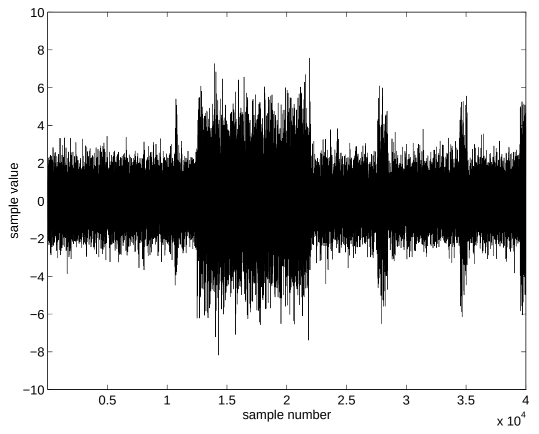
2.3.3 Complexity considerations

For both the ASVQ and the ATCQ, we use the empirical entropy of the output to detect the change in source statistics. We can calculate the empirical entropy at little extra cost since we keep generating the output data histogram.

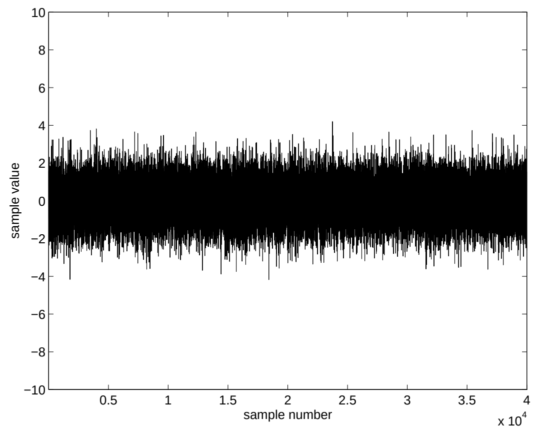
The input pdf estimation and the range adaptation take only a small fraction of the overall complexity because we have an estimation strategy which uses a linear interpolation for the pdf estimate and a simple decision rule for the range adaptation. Hence, from Fig. 2.1, most of the additional complexity of the ASVQ and the ATCQ compared to their respective non-adaptive quantizers arises from the quantizer redesign block.

In general, the SVQ and the TCQ are designed by iterative algorithms to find a set of reproduction levels for a given training sequence. Note however that, for the ASVQ and the ATCQ, the USQ levels are found based on the piecewise linear pdf estimate. By taking advantage of this, it is possible to implement the quantizer design algorithms with much reduced complexity as compared to their TCQ and SVQ counterparts based on a training set. As an example in the ATCQ case, all that is needed in each iteration is to adapt the USQ since the finite state machine to govern the state transition is fixed on top of the USQ.

We include a brief sketch of adaptation algorithm for SVQ and TCQ in the following.



(a)



(b)

Figure 2.4: (a) Samples generated by a bimodal Gaussian source switching between two Gaussian sources, N(0,1^2) and N(0,2^2) with the transition probability 0.001. (b) Samples from N(0,1^2).

Algorithm 1 *Sketch of Adaptive Quantization Using SVQ/TCQ*

For every N quantized samples from the SVQ or the TCQ,

1. Estimate the input pdf $\hat{f}(x)$ as described in Section 2.2.
2. Update the USQ using the Lloyd-Max design [38, 43] or the ECSQ design [11, 4] based on $\hat{f}(x)$.
3. For SVQ, calculate the length set $\mathcal{L}$ and the threshold L using $\hat{f}(x)$ and $\mathcal{Q}$ from the previous steps; No additional update is required for TCQ.
4. Quantize new inputs by SVQ or TCQ with the updated quantizer parameters.

2.4 Experiments and results

We compare our adaptive quantizers to the fixed SVQ and TCQ designed by training. We consider two types of input sources: (i) A bimodal source obtained by switching between two i.i.d. Gaussian sources with the same mean but different variances.

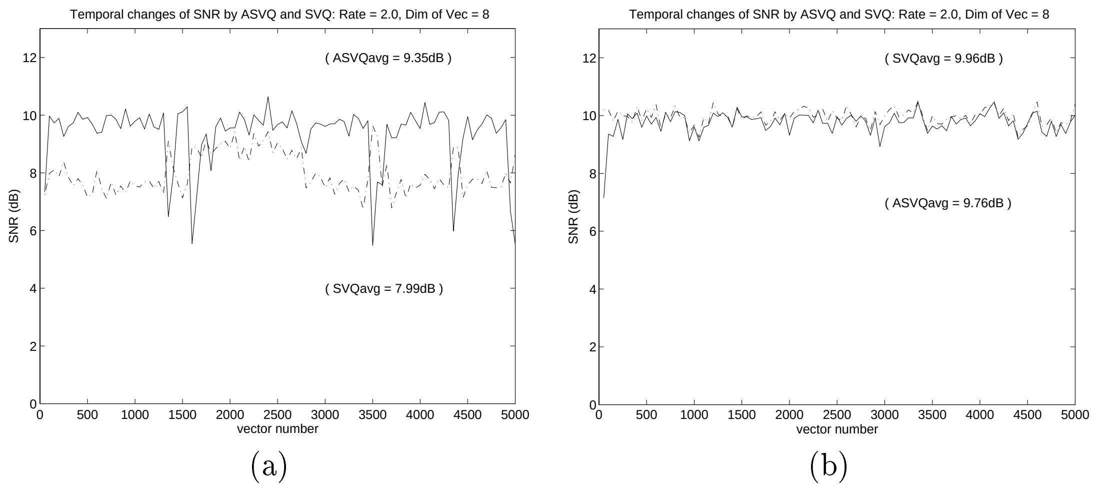
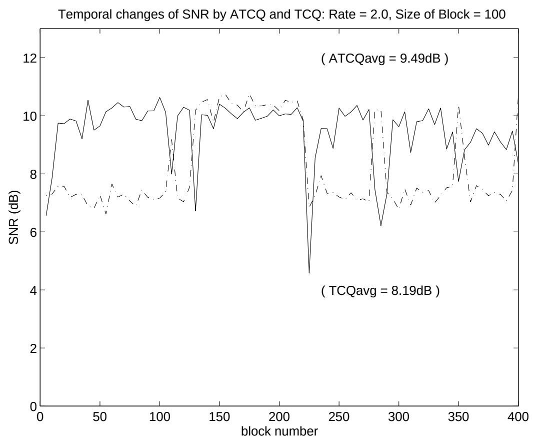


Figure 2.5: ASVQ vs. SVQ (a) When the input is bimodal with each of the modes being Gaussian of the same mean and different variances, the performance of the ASVQ remains nearly constant throughout the sequence except at the point of mode change while SVQ performance is better for one of the two modes. (b) When the input is stationary memoryless Gaussian, there is a slight degradation in performance due to the adaptation.

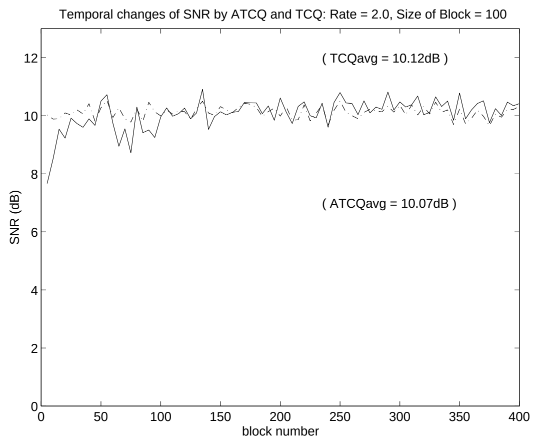
The transition probability between modes is 0.001; and (ii) an i.i.d. Gaussian source with parameters N(0,1^2). Samples from the two input sources are plotted in Fig. 2.4.

For each source, a sequence of 40,000 samples is used as an input to both the ASVQ and the SVQ. The SVQ training sequence to obtain $\mathcal{Q}$, $\mathcal{L}$, and L consists of 100,000 samples and has the same characteristics as the simulation input. We use an adaptation window of 50 vectors, as an adaptation parameter for the ASVQ. For both quantizers, the rate is 2.0 bits per sample and the vector dimension is 8.

Fig. 2.5 contains the plots of the SNR changes by the ASVQ and the SVQ through the entire simulation sequences. Fig. 2.5(a) shows that the proposed ASVQ performs better than the fixed SVQ when a non-stationary input source is applied. The average SNRs included in the figure are obtained by averaging the SNRs evaluated for sets of 50 vectors. The overall SNRs of the ASVQ and the SVQ for the entire input sequence are 8.89dB and 8.00dB, respectively. The performance loss of the ASVQ compared to the SVQ is minimal when the input is stationary, as we can see



(a)



(b)

Figure 2.6: ATCQ vs. TCQ (a)When the input is bimodal, the performance of the ATCQ remains nearly constant throughout the sequence except at the point of mode change and outperforms the TCQ while the TCQ performance is better for one of the two modes. (b) For a stationary memoryless Gaussian input source, the ATCQ experiences a slight degradation in performance due to the adaptation.

from Fig. 2.5(b). And, in this case, the overall SNRs of the ASVQ and of the SVQ are 9.73dB and 9.96dB, respectively.

We also experimented with an ATCQ and a TCQ under the same setting as for the ASVQ/SVQ experiment. Here we use a block length of 100 samples to generate a trellis and the adaptation window of 5 blocks. Again we use the encoding rate of 2.0. Fig. 2.6 includes the resulting plots of the SNR changes for the bimodal and the stationary Gaussian input sequences. The overall SNRs for the ATCQ and the TCQ are 9.32dB and 8.77dB for the bimodal source, and 10.03dB and 10.13dB for the stationary Gaussian source, respectively.

Finally, we summarize the experimental data in Table 2.1. We include, in parentheses, the data from the ASVQ and the ATCQ which use only the range adaptation algorithm to update quantizers provided each of the quantizers has a uniform USQ. Hence, we conclude that the ASVQ and the ATCQ with the input pdf estimation can achieve performance gain over those with the range adaptation only.

Quantizer	**Nonstationary bimodal**		**Stationary Gaussian**	
	$\mathrm{SNR}_{\mathrm{avg}}$	SNR	$\mathrm{SNR}_{\mathrm{avg}}$	SNR
ASVQ	9.35 (9.24)	8.89 (8.75)	9.76	9.73
SVQ	7.99	8.00	9.96	9.96
ATCQ	9.49 (8.87)	9.32 (7.61)	10.07	10.03
TCQ	8.19	8.77	10.12	10.13

Table 2.1: Performance comparison of rate-2.0 quantizers for non-stationary and stationary sources. $\mathrm{SNR}_{\mathrm{avg}}$ denotes the average of SNRs evaluated for blocks of input samples while SNR is the overall SNR for the whole sequence. In the parentheses are the SNRs from the adaptive quantizers using only the range adaptation on *uniform* USQ's. All results are in dB.

2.5 Conclusion and future work

In this chapter we have presented backward adaptive quantization using the scalar-vector quantizer (SVQ) and the trellis coded quantizer (TCQ). Both VQ's have an underlying scalar quantizer (USQ) in their structure, which makes it convenient to exploit the adaptation technique for scalar quantizers in [46]. The resulting adaptive scalar-vector quantizer (ASVQ) and adaptive trellis coded quantizer (ATCQ) redesign the USQ based on the past quantized outputs.

The adaptive quantizers require no side information while outperforming the SVQ and the TCQ, respectively, when the input signal is non-stationary. For an input sequence from a bimodal source switching infrequently between two Gaussian distributions with the same mean and different variances, both adaptive quantizers achieve performance gains of more than 1.3dB over the non-adaptive quantizers designed for a training set from the same bimodal source. Also the adaptive quantizers demonstrate minimal performance degradation due to adaptation when stationary inputs are considered.

As future work extended from this chapter, we can consider adaptation of extended quantizers from the SVQ and the TCQ, e.g., *trellis-based scalar-vector quantizer (TB-SVQ)* [32] and *arithmetic coded trellis coded quantizer (AC-TCQ)* [25, 26], which are among the best quantizers in the literature in terms of SNR quality.

It is also interesting to investigate the performance of the proposed adaptive quantization schemes in noisy environments where channel error can hamper synchronized adaptation in the encoder and decoder. Our adaptation algorithm can be contaminated by transmission error due to its backward adaptive nature, i.e., its dependency on the past encoded data. It is meaningful to evaluate the effect of adaptation in the presence of channel error, especially for the SVQ, since it has been originally proposed as a robust quantizer against the error propagation problem [30, 31]. A related work is to develop an error-robust adaptive quantizer based on ASVQ.

Chapter 3

Image Subband Coding Using Progressive Classification and Adaptive Quantization[1]

3.1 Introduction

Subband coding has been a popular and promising framework for image coding since Woods and O'Neil's early work using quadrature mirror filters (QMF's) [64]. The interest in subband coding techniques was renewed thanks in part to the recent work by Shapiro [52] which demonstrated that these techniques had the potential to outperform the more popular Discrete Cosine Transform (DCT) based systems (for example the JPEG standard [47]), both in terms of objective and subjective measures. In fact, subband or wavelet image coders are very likely to play a significant role in the new JPEG 2000 standard [16] for which work is just getting underway. Subjectively, images reconstructed after subband coding at low rates do not present the same blocking artifacts as those obtained with DCT-based methods. In addition, subband decompositions provide excellent space/frequency energy compaction so that (i) there are significant differences in total energy among different subbands and (ii) within each subband energy tends to be clustered spatially. The energy

[1]Part of this chapter represents work done jointly with Bin Yu. For related publications see [75, 76].

compaction property can be exploited through efficient quantization and entropy coding, explaining to a large extent the success of subband coding.

The space/frequency localization of subband image data has been exploited using two sets of techniques which were both utilized in Shapiro's breakthrough work [52]. First, subband coders take advantage of *efficient data structures*, e.g., zerotrees in [52, 70], hierarchical trees in [51], or inter-subband conditioning in [57]. These techniques make use of the correlation across subbands and the correlation within subbands. By comparison, JPEG only uses energy compaction properties and ignores any potential correlation between frequency contents in successive blocks (other than the correlation between DC coefficients which is exploited using DPCM coding).

The second component is the use of *adaptive techniques*, i.e., schemes which allow any of the three basic building blocks of typical image coding—transform (or linear decomposition), quantization, and entropy coding—to change from one image to another, or even locally within an image. Adaptivity can be achieved by using an adaptive arithmetic coder for entropy coding as in [52] or [57]. Other methods include adaptive selection of the subband decomposition for the given image [49] or adaptively pruned zerotrees in different regions of the image [70].

In this chapter we concentrate on incorporating *adaptive quantization techniques* in the subband image coding framework. While many adaptive quantization ideas already have been presented in the literature, most of those schemes were *forward adaptive*. That is, the encoder adaptively assigns the best quantizer from a finite collection of available quantizers to each of the subband coefficient sets by examining in advance the "current" data set to quantize. Note that forward adaptive quantizers, including the ones in [70] and [24], require overhead information to be sent to the decoder, thus increasing the required overall rate.

Alternatively, *backward adaptation* operates by adjusting the system through prediction of the current data characteristics, where the prediction is based on "previously processed" data. A good example can be found in arithmetic coding which achieves adaptation by having encoder/decoder keep track of already transmitted

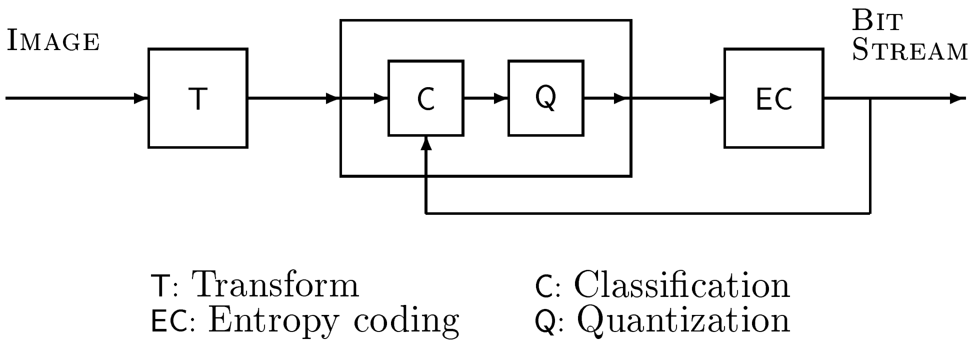


Figure 3.1: Building blocks in the conventional encoder for transform image compression. In our work the quantization block combines backward adaptive quantization and classification.

symbols and updating their probability models accordingly [63]. While a backward adaptive system cannot be as accurate in describing the upcoming information as a forward adaptive system, it has the obvious advantage of operating with *little or no overhead*, since the encoder adapts its quantizer based only on information available also at the decoder. The main drawback of the backward approach is the relatively higher complexity of the decoder as compared to a forward adaptive scheme.

In this chapter we develop novel techniques for backward adaptive quantization and demonstrate their effectiveness for subband image coding, especially at low rates. We consider a general quantization scheme which includes classification of subband coefficients (see Fig. 3.1). For image subbands it is useful to separate the coefficients in a given subband into different classes so that different quantization—*determined by bit allocation*—can be applied to each class. This fact is well known and a good summary on the previous classification efforts for image compression can be found in [24] and [27], which also demonstrate very successful applications of classification in subband image coding. We note that most previous classification attempts, including [24] and [27], can be categorized under forward adaptation: as a preprocessing stage to quantization, classification is performed over the subband data at the encoding end, generating explicit classification data (e.g., a classification map) which is delivered to the decoder as overhead. Forward adaptive classification is completed by assigning the optimal available quantizer to each class.

The contributions of the work in this chapter are summarized in the following. First, we propose a novel technique, *progressive classification*, to assign a class to each subband coefficient based on the previously quantized data. Thus we can reproduce the same classification at the decoder without explicit side information. However we still selectively use overhead to improve the performance. In particular, we transmit the *classification thresholds* which allow us to select a class given the causal past. We will present a technique to compute them. *The source statistics* of the classified coefficients are also included in the overhead as each class is modeled using a parametric distribution. Specifically, we approximate the distribution of each class using a mean-zero Laplacian model so that we only need to specify the Laplacian parameter. We will see that the Laplacian assumption allows efficient design of the classification thresholds as well as use of simple parametric quantizers. Note that the concept of progressive classification is analogous to the context-based adaptation methods (for example [61, 66, 3, 67, 39, 6]). The main difference is that in our case we use this adaptation mechanism for lossy rather than lossless compression (as in [61, 66]) and adapt *both* quantizer *and* entropy coder, while [3, 67, 6] only adapt the entropy coding.

The second contribution is the introduction of a method that allows the updating of the Laplacian parameter of a given class, *based on past quantized data*, thus allowing us to further take advantage of localized behavior. The on-the-fly update of the Laplacian parameter facilitates backward adaptation of the parametric quantizer in a similar way to the ideas introduced in Chapter 2 (see also [46] and [71]) where non-parametric piecewise linear distributions were used.

Finally, we demonstrate how these two techniques can be integrated in a complete subband image coding system having uniform threshold quantizer (UTQ) [11] as the baseline quantizer. The resulting image coder shows excellent performance, competitive with other state-of-the-art image compression algorithms. In particular, at low rates, the new scheme exhibits superior PSNRs comparing to other classification-based coding results. Our system has the potential for further improvements if the proposed adaptive scheme is combined with adaptive subband decompositions and/or more sophisticated quantizers instead of simple UTQ, e.g., the parametric optimal entropy-constrained scalar quantizer (ECSQ) [55].

The chapter is organized as follows. In Section 3.2 we introduce our progressive classification technique. An efficient use of forward adaptation to help the progressive classification is also discussed in the same section. In Section 3.3 we present techniques to allow the on-line model parameter estimation using the quantized data, and show how these can be applied using UTQ as the baseline quantizer. Implementation issues and experimental results are included in Section 3.4 and concluding remarks follow in Section 3.5.

3.2 Progressive classification

We can easily see the impact of data classification in general quantization applications: for an input source modeled as a mixture of sources with different distributions, classification allows us to use a set of quantizers customized to the individual distribution components. Then a sequence of quantizers can be used rather than a single average quantizer fitted to the overall input statistics. Thus we can localize quantization with respect to the classification information and improve performance through what is described as *classification gain* [22].

A typical example of image subbands obtained by 3-layer dyadic decomposition is depicted in Fig. 3.2 and serves to motivate that classification can provide potential benefits in subband image coding. For natural images, high activity regions such as edges are clustered and separated from surrounding low activity regions. We can identify those regions using distributions with large and small variances, respectively,

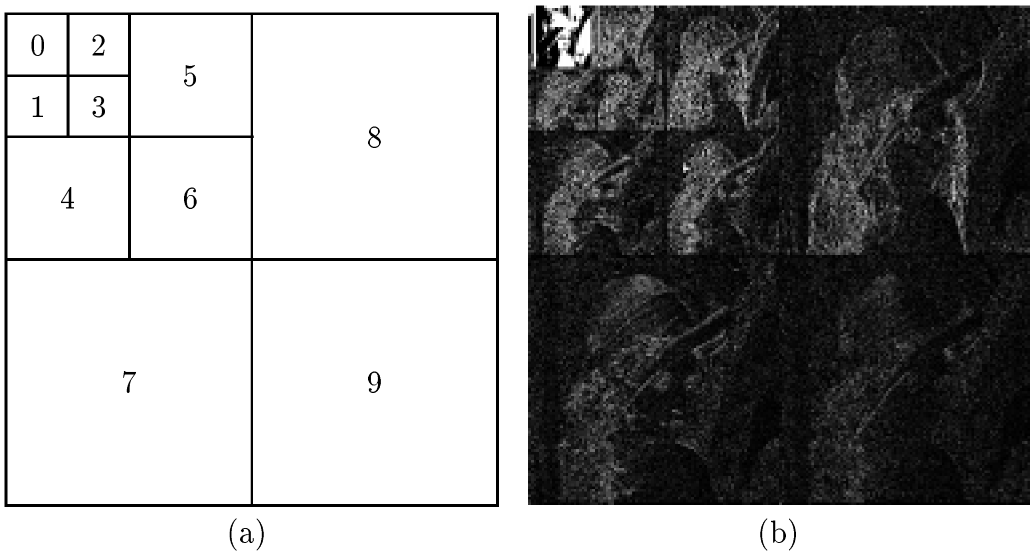
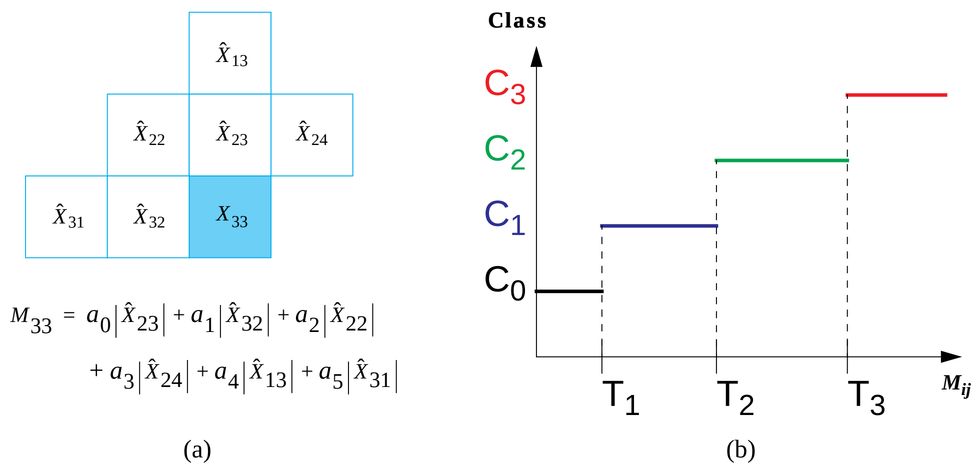


Figure 3.2: (a) Layout of the image subbands, with subband indices, from 3-level dyadic decomposition. (b) Subband coefficients obtained by 3-level dyadic decomposition of the Lena image using a wavelet filter set. We can observe frequency compaction of energy towards low frequency subbands and, more importantly, spatial clustering of the transform coefficients at similar energy level within each subband.

so as to assign appropriate quantizers. An extensive study, by Joshi *et al.* [27], on various classification-based image compression algorithms demonstrates substantial gains due to classification in subband image coding applications. The methods proposed in [27] are all based on block-wise classification: each block of coefficients within a subband is assigned to one of a finite number of classes and each class is characterized as generalized Gaussian source with different parameters. A different quantizer is then used for each class.

In the framework of [27] the assigned class (or class index) for each block of coefficients is explicitly stated as side information. To maintain a relatively low overhead, the size of blocks cannot be too small and only a limited number of classes can be used. Even under these constraints and even with efficient management of the overhead, the overhead rate may become excessive for low target bit rates. We can observe, by comparing the image coders proposed in [27] and [70], that the



$\hat{x}_{13}$
$\hat{x}_{22}$ $\hat{x}_{23}$ $\hat{x}_{24}$
$\hat{x}_{31}$ $\hat{x}_{32}$ x_{33}
$M_{33} = a_0|\hat{x}_{23}| + a_1|\hat{x}_{32}| + a_2|\hat{x}_{22}| + a_3|\hat{x}_{24}| + a_4|\hat{x}_{13}| + a_5|\hat{x}_{31}|$
(a)
Class
C_3
C_2
C_1
C_0
T_1 T_2 T_3
M_{ij}
(b)

Figure 3.3: (a) A template of 6 quantized coefficients forms the classification context. The activity of the current coefficient is predicted as the weighted average magnitude of the data from the template. (b) Thresholds on the predicted activity to define the classification mapping for each coefficient.

PSNR gain margin of a block-based forward classification coder over a good non-classification coder narrows as the encoding rate becomes low.

To avoid this penalty of block-based classification at low rates we introduce an image subband classification technique which aims at achieving both *finer grain classification*, down to the coefficient level rather than a block level, and *an efficient use of the classification overhead.*

3.2.1 Context-based classification

We define *context* as a set of previously transmitted data which is used to characterize (through prediction) the current data. Well known examples of context-based prediction include DPCM coding and adaptive entropy coding as employed for both lossless ([61, 66]) and lossy ([57, 67, 6]) image/video coding. For example in the case of entropy coding, the probability model selection is based on the context.

If we form a classification context with previously quantized coefficients, we can overcome the typical shortcomings of block-based classification. The context can be

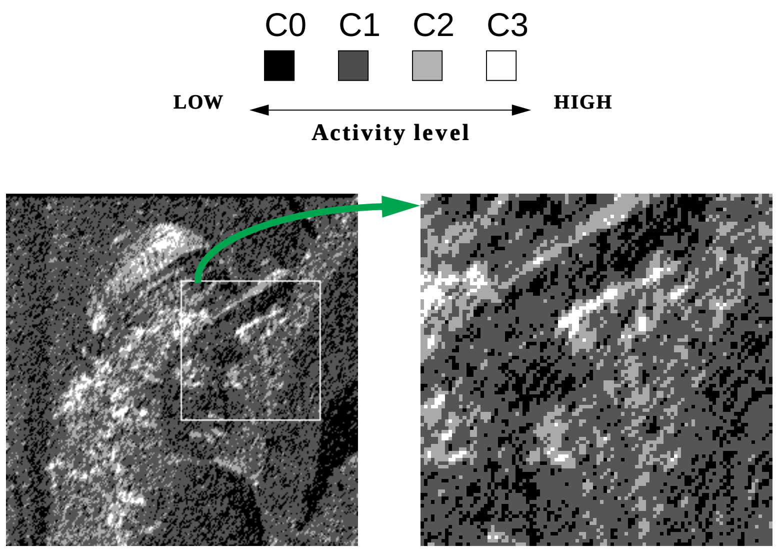


Figure 3.4: 4-level classification map of Subband 8. A causal 6-coefficient context from the same subband is used to obtain the map. Black indicates coefficients from the least active class and white depicts the most active. The right figure, the zoomed detail of the boxed region in the left, shows that classification is performed at the coefficient level.

used to classify image subbands on a coefficient-by-coefficient basis and, if needed, permit an increased number of classes without heavy use of overhead.

We assume a row-wise raster scan to illustrate the idea of context-based subband coefficient classification. Obviously the same idea can be applied with other scan options, e.g., column-wise, zigzag, Peano scan, etc. Let X_{ij} be the current coefficient located at the array coordinates (i, j) on a raster-scanned subband. Define $\mathcal{T}_{ij}$ as a template to derive the context, consisting of quantized coefficients in the causal neighborhood of X_{ij}. For example, we consider a 6-coefficient template (see Fig. 3.3(a)):

$$\mathcal{T}_{ij} = \{\hat{X}_{i-1j}, \hat{X}_{ij-1}, \hat{X}_{i-1j-1}, \hat{X}_{i-1j+1}, \hat{X}_{i-2j}, \hat{X}_{ij-2}\}. \tag{3.1}$$

At the subband boundaries only the available part of the template is considered.

Now we can establish a rule to classify each coefficient X_{ij} as a function of $\mathcal{T}_{ij}$. The classification rule is based on the predicted activity of X_{ij}. We predict X_{ij}'s

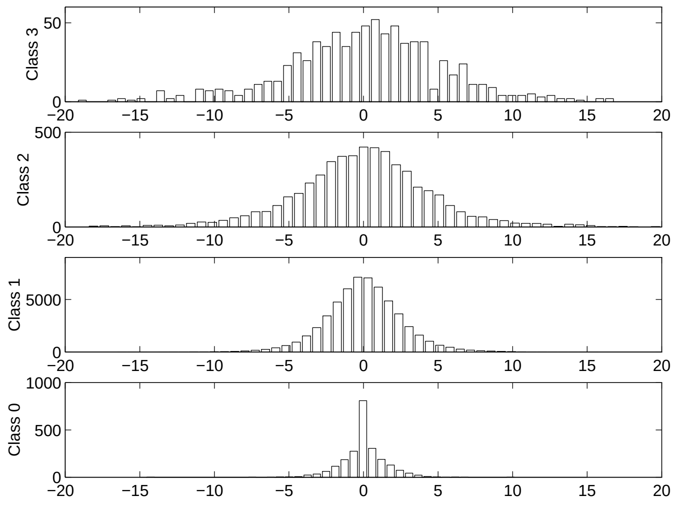


Figure 3.5: Histograms of the coefficients classified into 4 different sets, from high activity (top) to low activity (bottom) sets.

	Class 0	Class 1	Class 2	Class 3
m_X	0.0075	0.0083	-0.0272	0.0659
σ_X	1.4618	2.7450	4.6767	5.2891

Table 3.1: Means (m_X) and standard deviations (σ_X) of the class histograms in Fig. 3.5.

activity using the weighted average magnitude of the quantized coefficients in $\mathcal{T}_{ij}$, i.e.,

$$M_{ij} = a_0|\hat{X}_{i-1j}| + a_1|\hat{X}_{ij-1}| + a_2|\hat{X}_{i-1j-1}| + a_3|\hat{X}_{i-1j+1}| + a_4|\hat{X}_{i-2j}| + a_5|\hat{X}_{ij-2}| \quad (3.2)$$

where a_k, $0 \leq k \leq 5$, are the relative weights satisfying $\sum_k a_k = 1$. The predicted activity M_{ij} is then compared with the fixed classification thresholds (see Fig. 3.3(b)) to determine the class assignment for X_{ij}.

Consider Figs. 3.4 and 3.5 to motivate the performance of the proposed classification scheme. Fig. 3.4 depicts an example where 4 classes are used in Subband 8 of the Lena image (Fig. 3.2(a)). It clearly indicates that the areas with different energy levels are effectively classified. The zoomed region of the classification map shows that classification is performed at the coefficient level. Fig. 3.5 contains the resulting histograms of the data in the 4 classes and verifies that the classification has separated the coefficients into 4 classes with different degrees of activity, i.e., higher activity results in larger variance as given in Table 3.1.

The goal of classification is to allow assigning to each coefficient class a quantizer from a set of available quantizers[2]. In the following sections we consider the issue of selecting the set of available quantizers and, more importantly, finding the best quantizer for the given class. We will then discuss the design of the classification thresholds (e.g., T_1, T_2, and T_3 in Fig. 3.3(b)). Taking into account the difficulties identified from the threshold design, we will modify the baseline classification scheme presented in this section.

3.2.2 Parametric distribution estimation given unquantized data

To avoid the excessive overhead of describing the selected quantizers for different classes as well as to simplify the quantizer design, it is convenient to assume a parametric distribution model for each class. In this way, each quantizer can be uniquely identified from the model parameter(s) and, for a given class, the decoder needs to know only the model parameter estimate(s) from the encoder in order to dequantize correctly. The classification-based image coders in [27], for example, use a set of Arithmetic Coded Trellis Coded Quantizers (ACTCQ's) designed under the generalized Gaussian distribution (GGD) assumption. For each class, the selected quantizer—specified by the estimates of the shape parameter and the variance of GGD—is sent to the decoder as overhead (along with the class assignment information), as is typical of forward adaptive operation.

[2]This idea will be later extended in Section 3.3 where we show how the quantizer within a particular class can be adapted on the fly.

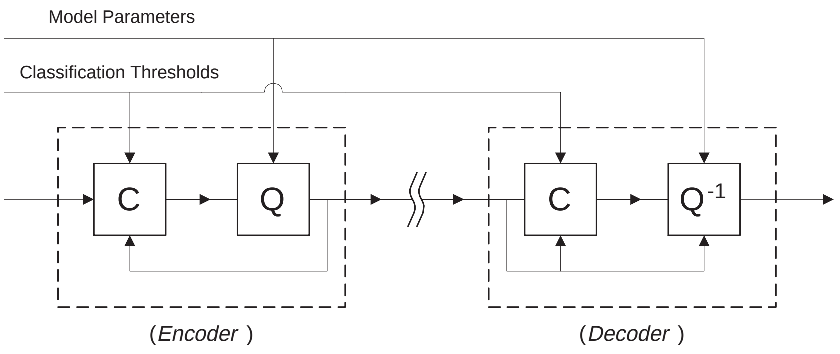


Figure 3.6: Overhead usage to help the backward adaptive classification scheme. The resulting overhead is negligible compared to those of forward adaptive block-based classification/quantization schemes.

In our scheme we also adopt a parametric modeling approach for class-adaptive quantization. Note that, since context-based classification does not require overhead, the side information to identify the model parameters for each class, together with the classification thresholds, is the only overhead required. See Fig. 3.6 for an illustration of the overhead usage in our image codec.

Many works in subband image compression (e.g., [56] and [39]) have favored zero-mean GGD to characterize the subband coefficient distribution. GGD modeling for image subband is justified in [62]. We first discuss GGD parameter estimation given a class of subband coefficients, assuming that the GGD is also good for each class in subbands. We then focus on Laplacian distribution, a special case of GGD, which makes our quantizer design and classification threshold design algorithms mathematically tractable. Note that the distribution for each class assumes zero mean as the actual average is indeed close to zero in general (see Table 3.1).

3.2.2.1 Generalized Gaussian modeling

The pdf of a GGD with *shape parameter* α and *scale parameter* β is defined as:

$$f_{\alpha,\beta}(y) = \left[\frac{\alpha\eta(\alpha,\beta)}{2\Gamma(1/\alpha)}\right] \exp(-[\eta(\alpha,\beta)|y|]^{\alpha}) \quad \text{for} \quad -\infty < y < \infty \tag{3.3}$$

where $\eta(\alpha, \beta) = \beta^{-1}\Gamma(3/\alpha)^{1/2}\Gamma(1/\alpha)^{-1/2}$ [11]. As a special case of GGD, for $\alpha = 1$, (3.3) reduces to the Laplacian pdf, with $\lambda = \sqrt{2}\beta^{-1}$,

$$f_\lambda(y) = \frac{1}{2}\lambda e^{-\lambda|y|} \quad \text{for} \quad -\infty < y < \infty. \tag{3.4}$$

The GGD modeling needs to estimate α and β for a given class of coefficients. While we can derive a shape parameter estimator as in [40], we can alternatively approximate α through the Kolmogorov-Smirnov test over a wide class of images [56]. Once α is fixed, β can be estimated from the subband coefficients as described in what follows.

Let $Y_1, \ldots, Y_n$ be i.i.d. observations from a GGD source with pdf of (3.3). Then $|Y_1|, \ldots, |Y_n|$ can be regarded as i.i.d. observations from the *positive generalized Gaussian pdf*

$$g_{\alpha,\beta}(y) = \left[\frac{\alpha\eta(\alpha,\beta)}{\Gamma(1/\alpha)}\right] \exp(-[\eta(\alpha,\beta)y]^\alpha) \quad \text{for} \quad y \geq 0. \tag{3.5}$$

Fix $\alpha = \alpha^*$ and define $\theta = \beta^{-1}$. (3.5) becomes

$$g_\theta(y) = \theta A_1 e^{-(\theta A_0 y)^{\alpha^*}} \quad \text{for} \quad y \geq 0, \tag{3.6}$$

where $A_0 = \Gamma(3/\alpha^*)^{1/2}\Gamma(1/\alpha^*)^{-1/2}$ and $A_1 = \alpha^* A_0 \Gamma(1/\alpha^*)^{-1}$. Thus the log-likelihood of θ given n independent observations Y_i, $i = 1, \ldots, n$, is

$$\ell(\theta) = n \log \theta + n \log A_1 - \sum_i (\theta A_0 |Y_i|)^{\alpha^*}. \tag{3.7}$$

By maximizing (3.7) with respect to θ, we can obtain the maximum likelihood estimator (MLE) of θ as

$$\hat{\theta} = \frac{1}{A_0}\left[\frac{n}{\alpha^* \sum_i |Y_i|^{\alpha^*}}\right]^{1/\alpha^*}. \tag{3.8}$$

By the Invariance Property of MLE [44], (3.8) yields the MLE of β as $\hat{\beta} = \hat{\theta}^{-1}$.

3.2.2.2 Laplacian modeling

Though the MLE of the GGD scale parameter is available in a handy form it still depends on the shape parameter which must be determined independently. For the general image subbands, a good choice of the shape parameter is $\alpha = 1$ that reduces the GGD to the Laplacian distribution. It has been pointed out [2] that the increased quantization error due to the model mismatch under the Laplacian assumption can be well compensated by its small modeling cost. Moreover, when we need a parameter estimation procedure based on quantized data, for example for the quantizer update in [75], we can avoid exhaustive numerical evaluation of integrals involved in the GGD modeling by using the Laplacian model.

The Laplacian assumption simplifies the positive generalized Gaussian pdf in (3.5) to the well-known "exponential" pdf

$$g_\lambda(y) = \lambda e^{-\lambda y} \quad \text{for} \quad y \geq 0 \tag{3.9}$$

where λ is identical to the Laplacian parameter of (3.4). Then the MLE of λ is given as

$$\hat{\lambda} = \frac{n}{\sum_i |Y_i|} = \frac{1}{\frac{1}{n}\sum_i |Y_i|} \tag{3.10}$$

by reducing (3.8) with $\alpha^* = 1$.

3.2.3 Classification threshold design

The 4-class example from Section 3.2.1 has demonstrated how context-based classification can split a set of coefficients into classes having different variances. As compared to the block-based methods it has the advantage of achieving classification regions with arbitrary shapes but its performance will largely depend on the threshold choice for M_{ij} of (3.2). We now discuss a method to define those thresholds for a given subband. Our algorithm consists of procedures to find $N-1$ thresholds from a set of candidate thresholds with the goal of maximizing the classification gain.

Note that there is inter-dependency between the classification thresholds and the quantizers, i.e., the actual classification will be based on the quantized past and the selected quantizers will depend on the classification thresholds. Hence it is difficult to estimate *a priori* performance of a particular threshold choice, and in general a particular choice can be shown to be effective only after it has been implemented. Instead of an iterative joint design to handle this inter-dependency, which may be optimal but is computationally costly, we will introduce a simple sequential design approach in the following. For the time being let us assume that we know the context for every coefficient, i.e., M_{ij} for each X_{ij}, and concentrate on describing the overall threshold design algorithm (this is equivalent to assuming that we have already selected a quantizer). Then we will present ideas to approximate M_{ij} in the design stage without performing actual quantization.

3.2.3.1 Threshold design algorithm

Given all the context information available, we want to determine the class to which each of the (unquantized) coefficients belongs. We start by considering $N_0 > N$ initial classes $\{C_0, \ldots, C_{N_0-1}\}$ defined by a set of $N_0 - 1$ monotonically increasing thresholds

$$\{T_k, 1 \leq k \leq N_0 - 1 : 0 < T_1 < \cdots < T_{N_0-1} < \infty\}. \tag{3.11}$$

The classification rule is such that a coefficient X_{ij} is assigned to C_k if

$$T_k \leq M_{ij} < T_{k+1} \tag{3.12}$$

for $k = 0, \ldots, N_0 - 1$, with $T_0 = 0$, $T_{N_0} = \infty$, where M_{ij} is the activity prediction of X_{ij} in (3.2). For a given class C_k, n_k is the number of coefficients in the class, i.e.,

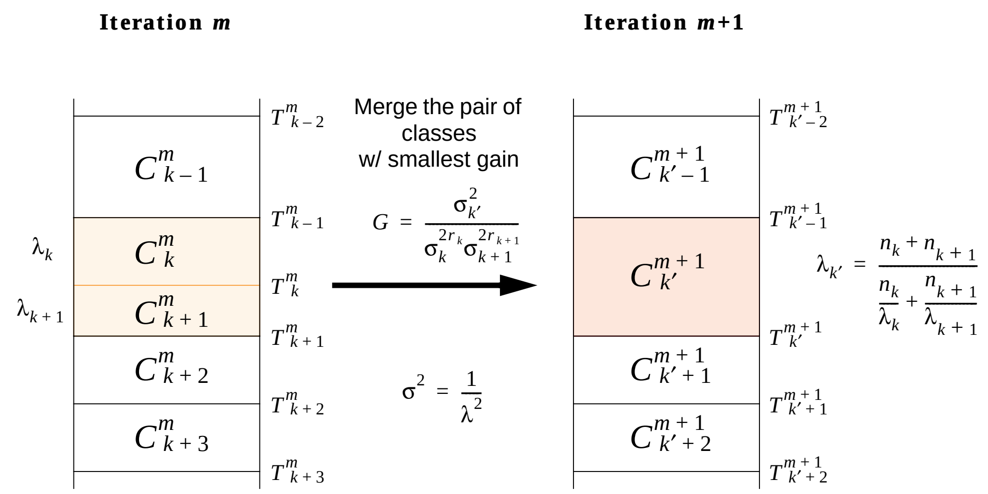


Figure 3.7: Illustration of the merging process in the classification threshold design. At the m-th iteration, the pair of the k-th and the $(k+1)$-st classes has the smallest classification gain so as to merge into one class.

the total number of X_{ij} such that $T_k \leq M_{ij} < T_{k+1}$. See Fig. 3.3(b) for the mapping associated with this classification rule.

After collecting the coefficients into the N_0 initial classes according to (3.12), the MLE of the Laplacian parameter for each class, $\hat{\lambda}_k$, $k = 0, \ldots, N_0 - 1$, is determined as described in Section 3.2.2. Since the final thresholds will be chosen from the initial set it is desirable to initialize the design algorithm with a large N_0 so that the final $N-1$ thresholds can be at almost any arbitrary point within the support of M_{ij}. However N_0 cannot be arbitrarily large because each of the initial classes should contain enough coefficients to ensure a reliable estimate of $\hat{\lambda}_k$. Furthermore the initial thresholds should be chosen carefully such that each class has approximately the same number of coefficients, which is important to guarantee equally reliable estimates for different classes. We notice that the trivial choice of uniformly spaced thresholds is in general not appropriate in this sense.

We apply a greedy algorithm which removes one class at a time until the desired number of classes is reached. The algorithm operates iteratively by merging at each stage the pair of classes having the smallest classification gain among all pairs of adjacent (in the sense of corresponding to consecutive thresholds) classes. As in [24] we estimate the classification gain by comparing the variances of two classes, say σ_k^2 and σ_{k+1}^2, with the variance $\sigma_{k'}^2$ of the union of the two classes. See Fig. 3.7 for one iteration of the threshold design algorithm.

Let n_k and n_{k+1} be the respective numbers of coefficients in two adjacent classes C_k and C_{k+1} for which the parameter estimates are $\hat{\lambda}_k$ and $\hat{\lambda}_{k+1}$ respectively. Assume that the Laplacian model is still valid for the new class obtained after merging C_k and C_{k+1}. Then we can use the Laplacian parameter estimator in (3.10) to find the Laplacian parameter $\hat{\lambda}_{k'}$ of the new class

$$\hat{\lambda}_{k'} = \frac{n_k + n_{k+1}}{n_k/\hat{\lambda}_k + n_{k+1}/\hat{\lambda}_{k+1}}. \tag{3.13}$$

Under the Laplacian assumption the variance is equal to $1/\lambda^2$ and thus the classification gain [22]

$$G = \frac{\sigma_{k'}^2}{\sigma_k^{2r_k}\sigma_{k+1}^{2r_{k+1}}}, \tag{3.14}$$

can be estimated by

$$G = G(\lambda_k, \lambda_{k+1}; n_k, n_{k+1}) = \frac{\hat{\lambda}_k^{2r_k}\hat{\lambda}_{k+1}^{2r_{k+1}}}{\hat{\lambda}_{k'}^2} \tag{3.15}$$

where $r_k = n_k/(n_k + n_{k+1})$ and $r_{k+1} = 1 - r_k = n_{k+1}/(n_k + n_{k+1})$.

We notice that the computational complexity is kept very low when assuming the Laplacian model. Once we calculate the Laplacian parameter estimates for the initial classes we do not have to recalculate a new set of $\hat{\lambda}_k$'s using (3.10) at every iteration. Instead we use (3.13) to find the new parameter estimate, without affecting the other classes not involved in merging, and (3.15) to evaluate the classification gain for a new set of classes.

We summarize the threshold design procedure under the Laplacian model assumption in the following. Given a subband, the desired number of classes N, and N_0 initial classes ($N_0 > N$) corresponding to the initial threshold set in (3.11),

Algorithm 2 *Classification Threshold Design*

1. Find the estimates of $\lambda_0, \ldots, \lambda_{N_0-1}$ by (3.10). Set $K = N_0$.

2. Find k^* s.t.
$$k^* = \arg \min_{0 \le k < K} G(\lambda_k, \lambda_{k+1}; n_k, n_{k+1})$$
where $G(\cdot)$ is given in (3.15).

3. Update the thresholds, the class sizes, and the Laplacian parameters by merging classes k^* and $k^* + 1$ such that the merged class has index $k = k^*$ with $n_k = n_k + n_{k+1}$. Update λ_k according to (3.13) for $k = k^*$. T_k, n_k, and λ_k remain unchanged for the other values of k.

4. Set $K = K - 1$. STOP if $K = N$. Otherwise go to Step 2.

The issue of selecting an appropriate number of classes will be discussed in Section 3.4.

3.2.3.2 Approximation of quantized context M_{ij}

As we mentioned earlier, we do not know the quantized coefficients $\{\hat{X}\}$ *a priori*. Thus M_{ij} given by (3.2) cannot be used to set up the thresholds. To resolve this difficulty, [75] uses pre-quantization before applying the threshold design algorithm. In this idea, the subband coefficients are initially quantized by a uniform quantizer of stepsize Δ_0 as an approximation to the actual UTQ. Then, with pre-quantized coefficients $\{\hat{X}^0\}$, M_{ij} is approximated by

$$M^0_{ij} = a_0|\hat{X}^0_{i-1j}| + a_1|\hat{X}^0_{ij-1}| + a_2|\hat{X}^0_{i-1j-1}| + a_3|\hat{X}^0_{i-1j+1}| + a_4|\hat{X}^0_{i-2j}| + a_5|\hat{X}^0_{ij-2}|. \quad (3.16)$$

However the pre-quantization technique introduced in [75] has a drawback of using a value of Δ_0 chosen *ad hoc*. Moreover, while the eventual UTQ stepsize Δ depends on the coding rate, Δ_0 is fixed for all rates so that the threshold design of Algorithm

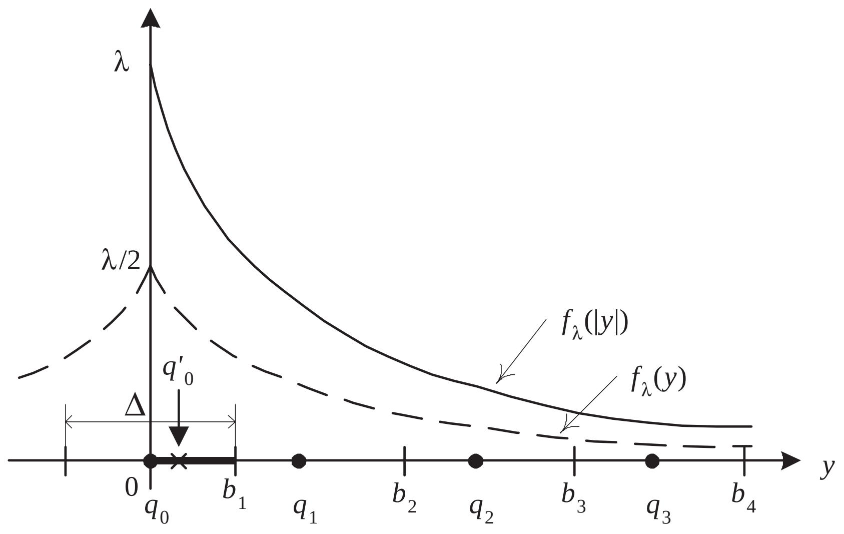


Figure 3.8: The positive half of the UTQ with stepsize Δ. q_j's and b_j's are quantization levels and bin boundaries, respectively. q'_0 is the conditional mean of $|y|$, i.e., the centroid, in the interval of $[0, b_1)$ where $|y|$ follows the exponential distribution $f_\lambda(|y|)$.

2 generates a fixed threshold set regardless of the rate. Thus, in this work, we consider an approach to approximate M_{ij} using pre-quantization with a rate-dependent stepsize. This modified approach is based on the techniques introduced in [39] for context-based estimation of the wavelet coefficient magnitude.

First, suppose that we approximate M_{ij}^U using only the unquantized coefficients, i.e.,

$$M_{ij}^U = a_0|X_{i-1j}| + a_1|X_{ij-1}| + a_2|X_{i-1j-1}| + a_3|X_{i-1j+1}| + a_4|X_{i-2j}| + a_5|X_{ij-2}|. \quad (3.17)$$

For (3.17) to be a good approximation to M_{ij} in (3.2), $|X|$ must be a good approximation to $|\hat{X}|$. This is in general true since all the inputs X are quantized to the closest $\hat{X}$ and thus, if we have no information about the quantizer, X may well approximate $\hat{X}$. However, if X is in the center bin of the quantizer, we will have $|X| > 0$ in most cases while $|\hat{X}| = 0$ and thus this particular case has to be treated separately to avoid biased estimation.

Fig. 3.8 depicts the quantization levels and the bin boundaries of a UTQ of stepsize Δ obtained under the Laplacian model assumption. It shows why $|X|$ is not a good approximation to $|\hat{X}|$ in the center bin, i.e., $|X|$ approximates q_0', the centroid of the exponential pdf $f_\lambda(|y|)$ in the interval $[0, b_1)$, rather than $|\hat{X}| = q_0 = 0$.

The mismatch between $|X|$ and $|\hat{X}|$ in the center bin results in M_{ij}^U overestimating M_{ij}. This is significant especially when the context region contains many zeros, which is frequently observed in high frequency subbands. In addition, the error when using the M_{ij}^U of (3.17) to approximate M_{ij} tends to increase at low rates, as more coefficients are quantized to zero.

To reduce the effect of mismatch between $|X|$ and $|\hat{X}|$ in the center bin, [39] proposed to treat the coefficients having "small" magnitude differently from the other coefficients. A specific procedure was used to 1) choose a rate-dependent threshold δ_0 to identify the coefficients of small magnitude; 2) pre-quantize the small-magnitude coefficients to zero while preserving the other coefficients; and 3) find and characterize the *set of unpredictable coefficients* for which the context consists of all zero pre-quantized values. Note that in the last point we are taking care of a particular situation (all-zero context) which will be treated as a separate context.

As the rate-dependent threshold, [39] suggests

$$\delta_0(\xi) = \sqrt{\xi / 0.264} \tag{3.18}$$

where ξ is the Lagrange multiplier to control the rate-distortion trade-off in the entropy-constrained quantization and 0.264 is the value of the Lagrange multiplier which yields the UTQ of stepsize $\Delta = 2.0$, i.e., a center bin of $[-1.0, +1.0)$, for the unit-variance Laplacian distribution. While we refer to [39] for the details in deriving (3.18), an intuition can be given to verify that $\delta_0(\xi)$ appropriately conforms to the change of the rate. For example, a large ξ forces a low-rate UTQ yielding increased number of all-zero contexts. This aspect can be captured by a large $\delta_0(\xi)$, proportional to $\sqrt{\xi}$, which produces a large number of unpredictable (i.e., zero context) coefficients.

We identify the unpredictable coefficient set by pre-quantizing a given subband with the threshold δ_0 of (3.18) and then collecting a set of coefficients for which the context is made of only the coefficients thresholded to zero. For the other coefficients, we find $N-1$ thresholds, corresponding to N *ordinary* classes, by applying Algorithm 2. The Laplacian model parameter for the zero-context class is estimated also using (3.10) and then included in the overhead.

Note that the new classification threshold design generates a total of $N+1$ classes, i.e., N ordinary classes and 1 zero-context class. Thus we need a slight modification to the original N-level context-based classification procedure of Section 3.2.1, where we simply need to assign X_{ij} to the zero-context class if M_{ij} from Equation (3.2) is equal to zero. If M_{ij} is not equal to zero, we need to recalculate M_{ij} by substituting zero quantized coefficient(s) with q_0' before comparing M_{ij} with the thresholds.

3.3 Adaptation of uniform threshold quantizer

We employ uniform threshold quantizer (UTQ) to quantize the classified subband coefficients. First, we briefly review UTQ and its efficient design under the Laplacian model assumption, where a quantizer can be matched to $\hat{\lambda}$ of each class. We then consider on-line adaptation of UTQ within each class which can be achieved by update of the Laplacian parameter estimate of each class using the past quantized coefficient.

3.3.1 Model-based UTQ

UTQ is a popular choice for image subband quantization [56, 39] as it effectively approximates the optimum entropy-constrained scalar quantizer (ECSQ) for GGD without needing the complication of the optimum ECSQ [11]. The structure of UTQ, completely defined by its fixed stepsize[3] Δ, reduces its design and description complexity while the reconstruction levels are optimized by the centroid condition. UTQ outperforms the optimal fixed-rate scalar quantizer, designed under the minimum distortion criterion, as well as the simple uniform quantizer.

The entropy-constrained design algorithm of UTQ in [11] can be simplified by assuming a parametric model for the input distribution as shown in the following. Given a fixed Lagrange multiplier ξ which controls bit allocation, the design objective is to minimize the joint rate-distortion (R-D) cost function $J = D + \xi R$ where $D = D(\Delta)$ and $R = R(\Delta)$ are respectively the distortion and the rate associated with a UTQ with stepsize Δ. The design goal can be achieved by finding the minimizing stepsize Δ^* since Δ is the only independent variable of the cost function. A simple bisection search [48], for example, can be used to iteratively find Δ^*.

At each iteration of the algorithm, we need to evaluate q_j and p_j, the reconstruction level and its probability in the j-th bin of UTQ, in order to calculate $D(\Delta)$ and $R(\Delta)$. We will show that our Laplacian model assumption gives closed form expressions for q_j and p_j. Consider a UTQ with $(2L+1)$ levels. By the zero-mean assumption, p_j and q_j are symmetric and antisymmetric, respectively, i.e., $q_{-j} = -q_j$ and $p_{-j} = p_j$ for $j = 1, \ldots, L$. For the center bin and the bins from the positive half of the distribution,

$$p_0 = 2\int_0^{b_1} f_\lambda(y)dy = 1 - e^{-\lambda b_1} \tag{3.19}$$

$$q_0 = 0 \tag{3.20}$$

and

$$p_j = \int_{b_j}^{b_{j+1}} f_\lambda(y)dy = \frac{1}{2}\left[e^{-\lambda b_j} - e^{-\lambda b_{j+1}}\right] \tag{3.21}$$

[3]A more general definition of UTQ allows a different stepsize for the center bin from the fixed Δ for the outer bins. In this case we often refer to the center bin as *deadzone*.

$$q_j = \frac{1}{p_j}\int_{b_j}^{b_{j+1}} y f_\lambda(y)dy = \frac{1}{2p_j}\left[b_j e^{-\lambda b_j} - b_{j+1}e^{-\lambda b_{j+1}}\right] + \frac{1}{\lambda}, \qquad (3.22)$$

for $j = 1, \ldots, L$, where $b_j = j - 1 + \Delta/2$ denotes the j-th (positive) bin boundary and $f_\lambda(y)$ is the Laplacian pdf in (3.4). As a result, once we know the Laplacian parameter λ—in practice, its estimate $\hat{\lambda}$—we have a much simpler UTQ design compared to that based on other distribution models which may involve numerical integrations. Note that a non-iterative design algorithm of UTQ for exponential and Laplacian distributions is also available in [55].

3.3.2 Adaptive UTQ: Parameter estimation given quantized data

In the baseline quantization system the Laplacian parameter estimate $\hat{\lambda}_k$ for Class C_k is obtained in the classification threshold design, based on the unquantized coefficients. We now show how the quantizer can be further refined by estimating $\hat{\lambda}_k$ on the fly based on the previously quantized data. Note that the basic idea is extended from the non-parametric pdf estimation paradigm introduced in [46] to a parametric case.

With the same assumptions and notations from Section 3.2.2, consider an L-level quantizer with decision levels

$$0 = b_0 < b_1 < b_2 < \ldots < b_L < \infty$$

for which n_j, $1 \leq j \leq L$, is the number of data quantized to the j-th reproduction level. Note that the quantizer is designed for the exponential distribution associated with the magnitude data $|Y_1|, \ldots, |Y_n|$ where $Y_1, \ldots, Y_n$ are samples from a Laplacian distribution—see also Fig. 3.8. Thus

$$n_j = \sum_{i=1}^{n} I_{[b_{j-1}, b_j)}(|Y_i|) \quad \text{for} \quad j = 1, 2, \ldots, L \qquad (3.23)$$

$$\sum_{j=1}^{L} n_j = n \tag{3.24}$$

where

$$I_{[a,b)}(y) = \begin{cases} 1 & \text{if} \quad a \le y < b \\ 0 & \text{otherwise} \end{cases} . \tag{3.25}$$

Let $\hat{P}_l$ be the cumulative normalized frequency of occurrences of data quantized to levels $1, 2, \ldots, l$, i.e.,

$$\hat{P}_l = \sum_{j=1}^{l} \hat{p}_j \tag{3.26}$$

where $\hat{p}_j = n_j/n$ is the normalized count of the data quantized to the j-th level.

For any given l, we can find an estimator of λ as follows. Denote by $F_\lambda(y)$ the cumulative distribution function (cdf) of the exponential random variable with parameter λ. That is, for $y > 0$,

$$F_\lambda(y) = \int_0^y \lambda e^{-\lambda t} dt = 1 - e^{-\lambda y}. \tag{3.27}$$

$\hat{P}_l$ is a good empirical estimate, based on a total of n data, of $F_\lambda(b_l)$ which is the proportion of data falling between b_0 and b_l assuming infinite data. To be more precise, by the Law of Large Numbers,

$$\hat{P}_l \xrightarrow{\text{i.p.}} F_\lambda(b_l) \tag{3.28}$$

as $n \to \infty$.

By solving (3.27) for λ, we get

$$\lambda = -\frac{\log(1 - F_\lambda(y))}{y}. \tag{3.29}$$

Hence a reasonable estimate of λ based on $F_\lambda(b_l)$ can be obtained as

$$\hat{\lambda}_l = -\frac{\log(1 - \hat{P}_l)}{b_l}. \tag{3.30}$$

The question remains how to choose l. Let $P_l = F_\lambda(b_l) = 1 - e^{-\lambda b_l}$ then

$$\mathrm{Var}(\hat{P}_l) = \frac{n}{P_l(1-P_l)} \tag{3.31}$$

since $n\hat{P}_l$ follows a binomial distribution $\mathcal{B}(n, P_l)$. We can approximate the variance of $\hat{\lambda}_l$ as

$$\frac{1}{nb_l^2}\frac{P_l(1-P_l)}{(1-P_l)^2} = \frac{P_l}{nb_l^2(1-P_l)} = \frac{e^{\lambda b_l}-1}{nb_l^2}.$$

It follows that for $\hat{\lambda}_l$ to have a small variance, $(e^{\lambda b_l}-1)/b_l^2$ should be small. Since

$$(e^{\lambda b_l}-1)/b_l^2 \to \infty \tag{3.32}$$

as $b_l \to 0$ or ∞, there exists a finite value of b_l somewhere for the approximate variance of the estimate to be at its minimum. To know the exact value of this minimum, we need to know the true value of λ which is unavailable. An alternative approach is to pick an initial estimate λ_0 which corresponds to some b_{l_0} in the *middle* of the range of the quantizer decision levels and then find l^* such that

$$l^* = \arg\min_l (e^{\lambda_0 b_l}-1)/b_l. \tag{3.33}$$

Or we can just use the initial guess l_0 to make the procedure simpler. It seems reasonable to pick l_0 such that $\hat{P}_{l_0}$ is around 1/2.

For each of the classes the encoder transmits to the decoder the Laplacian parameter estimate —based on unquantized coefficients—that fits the statistics of the whole data in the class. This parameter serves to initialize the adaptive quantizer. The adaptation algorithm estimates an updated Laplacian parameter based on the past quantized data in each class. Since the initial Laplacian parameter is a fairly good representation of the data in the class we in fact use a weighted average between the initial quantizer and the newly obtained (local) one. The system could be

	`Lena.512`		`Goldhill.512`	
	0.5 bpp	1.0 bpp	0.5 bpp	1.0 bpp
UTQ	35.37	38.89	32.15	35.57
PC-UTQ	36.23	39.57	32.61	36.05
PC-AUTQ	36.31	39.68	32.71	36.18

Table 3.2: PSNR (dB) comparison of different quantization configurations. Subband data are generated by 3-layer dyadic decomposition with the Daubechies D4 filters. PC-UTQ and PC-AUTQ use 4 classes. Adaptive arithmetic coding is considered to obtain the bit rates. For simplicity, the classification algorithm does not include the all-zero context class in this example.

further improved by using overhead to give more degrees of freedom in determining the centroids (cf. [77]), deciding on a block-by-block basis whether to adapt the quantizer, etc.

To see the effect of the quantizer adaptation, we experiment with the subband quantization system in three different configurations: baseline UTQ, UTQ with 4-level progressive classification (PC-UTQ), and PC-UTQ with quantizer adaptation (PC-AUTQ). We summarize the results for the Lena and Goldhill images (512×512) in Table 3.2. We consider the adaptive arithmetic coder output bit stream size to compare the peak signal-to-noise ratio (PSNR) defined as

$$\mathrm{PSNR} = 10 \log_{10} \frac{255^2}{\mathrm{MSE}} \tag{3.34}$$

where MSE is the mean square error between the original image and the compressed image. The Daubechies D4 filters [9] are used in the 3-layer dyadic decomposition for all three configurations. To keep the experiment simple, the context-based classification uses only the ordinary classes with the thresholds obtained by the fixed stepsize pre-quantization and the approximated activity prediction of 3.16.

This preliminary result shows that the quantizer adaptation is quite useful while a significant gain is achieved via progressive classification of subbands. We note however that the quantizer adaptation gain depends on the choice of the filters for subband decomposition. In the next section we will consider a different filter choice to compare PC-AUTQ with other high performance wavelet image coders based on the same filter choice. We will see that, for the new filter choice, the PSNR performance is improved for all three configurations while the gain from adaptation becomes smaller.

3.4 Experiments and results

We have implemented an image coder using the proposed classification/quantization ideas and experimented with popular 512×512 gray-scale images. Throughout the experiments, we consider a 3-layer dyadic decomposition. The subband layout from the decomposition is depicted in Fig. 3.2(a). We mainly use the 9/7-tap biorthogonal wavelet filters [60] to compare our results with other state-of-the-art results obtained by using the same filter choice. We also consider the 32-tap quadrature mirror filters (QMF's) [23]. While both filter sets are popular for their good performance in subband image compression, the 32-tap QMF's tend to be better than the other—especially for the images containing a large amount of high frequency components—but the 9/7-tap filters enable faster subband transform. In processing the coefficients in each subband, we scan Subbands 1, 4, and 7 column-wise and the other subbands line by line in order to utilize the spatial orientation of each subband.

We apply our progressive classification and adaptive uniform threshold quantization (PC-AUTQ) to the coefficients in the subbands except the lowest frequency subbands, i.e., Subband 0 in Fig. 3.2(a), where we use PC-AUTQ as part of 4-th order DPCM coding. As the predicted coefficient value at the current position to calculate differential data, a weighted mean of the 4 nearest causal neighboring coefficients is used. The weights are experimentally determined and fixed for all input images.

The differential data in Subband 0 can be also modeled as samples from Laplacian sources. Hence we can use the same modeling assumption for Subband 0 and the other high frequency subbands (Subband 1 to 9) so that we use a single quantization algorithm for all subbands.

Bit allocation is performed over all classes of coefficients from 10 subbands simultaneously. An iterative bit allocation based on [53] is used to determine the Lagrange multiplier ξ in the cost function $J = D + \xi R$ where D and R are the overall distortion and rate from each attempt to find the stepsizes of UTQ's which minimize J. We stop the iteration if we find $\xi = \xi^*$ for which the corresponding R falls within a certain range from the target rate. Since ξ^* specifies the resulting bit allocation or, equivalently, the UTQ stepsize Δ^* for each class, ξ^* is quantized using 16 bits and sent to the decoder as overhead.

The rate used to compare our results consists of the actual output rate from the entropy coder and the overhead rate. In our algorithm the overhead rate remains the same for different compression ratios. The bits to identify the classification thresholds (T_k) in each band and the Laplacian parameter estimate ($\hat{\lambda}_k$) in each class take up most of the overhead. For instance suppose that each of the S subbands has $N+1$ classes: N ordinary classes and 1 special class to handle the all-zero context. The overhead for the thresholds and the Laplacian parameter estimates is

$$B \cdot S \cdot (N-1) + B \cdot S \cdot (N+1) = 2N \cdot B \cdot S \tag{3.35}$$

where B bits are assumed for each of the $N-1$ classification thresholds and $N+1$ Laplacian parameters in each subband. Specifically we use 8 bits per parameter in the experiment. Then the 10-band decomposition with 4 ordinary classes and 1 zero-context class needs overhead of 640 bits per image or, equivalently, 0.0024 bpp for a 512×512 image[4].

We use an adaptive arithmetic coder for entropy coding of the index stream corresponding to the quantized coefficients. The probability model of the index

[4]The total overhead including the other miscellaneous side information is 0.00247 bpp for the actual compression of 512×512 images

Rate (bpp)	EZW	SPIHT	SFQ	EQ	PC-AUTQ
0.20	-	33.16	33.32	33.46	33.41
0.25	33.17	34.13	34.33	34.45	34.46
0.50	36.28	37.24	37.36	37.58	37.56
1.00	39.55	40.45	40.52	40.85	40.75

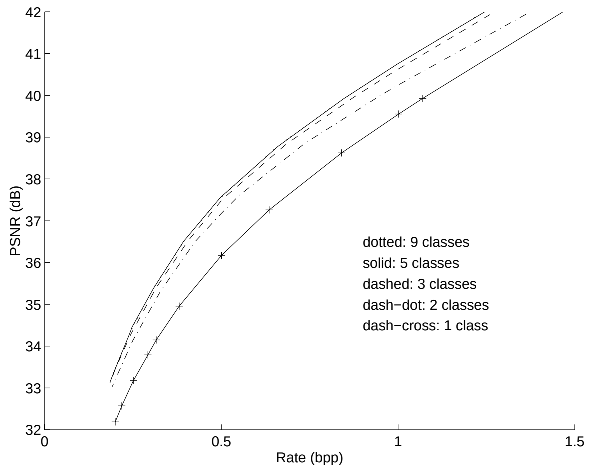
Table 3.3: Performance comparison in PSNR (dB) for Lena.

Rate (bpp)	EZW	SPIHT	SFQ	EQ	PC-AUTQ
0.20	-	29.84	29.98	30.05	30.04
0.25	-	30.55	30.71	30.77	30.78
0.50	-	33.12	33.37	33.42	33.46
1.00	-	36.54	36.70	36.89	36.99

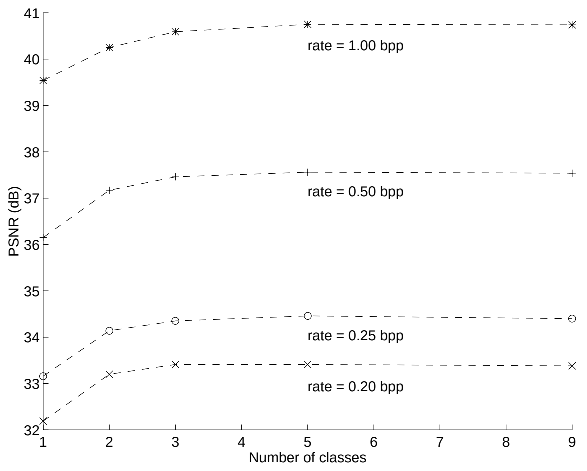
Table 3.4: Performance comparison in PSNR (dB) for Goldhill.

stream used in the arithmetic coder is initialized for each class with aid of the Laplacian parameter estimate for the class. Thus no additional overhead is required for the initialization of the arithmetic coder while the compression performance is improved over an adaptive arithmetic coder without any particular assumption on the initial probability models.

We might expect a better performance for our classification-based encoder by increasing the number of classes. Fig. 3.9 shows the plots of the PSNR performance as a function of (a) the rate and (b) the number of classes for Lena decomposed with the 9/7-tap filters. From both plots, we can see little improvement of PSNR by using more than 5 classes. Moreover, from (3.35), the overhead rate increases linearly as more classes are used. Thus we limit the number of classes in our experiment to 5, including the all-zero context class, and this is found to be a good number also for the other test images. We have similar results for the 32-tap filters. We note that the modified method for approximation of M_{ij}, as described in Section 3.2.3.2, is used to design the classification thresholds.



(a)



(b)

Figure 3.9: Effect of the number of the classes on the PSNR performance of PC-AUTQ in compressing the Lena image. In the cases of 2 or more classes, the special class for the all-zero context is included. (a) PSNR vs. rate performance for different number of classes. Note that the dotted line for 9 classes is overlapped by the solid line for 5 classes. (b) PSNR as a function of number of classes for various output rates.

We summarize the PSNR results of PC-AUTQ in Tables 3.3 and 3.4, for Lena and Goldhill, respectively, and compare with those of popular and state-of-the-art methods from the literature. In the table EZW stands for the Embedded Zerotree Wavelet coder [52], SPIHT for the Set Partitioning in Hierarchical Trees algorithm [51], and SFQ for the Space-Frequency Quantization coder using joint optimization of zerotree quantization and non-zero coefficient quantization [70]. Also EQ denotes the Estimation-Quantization scheme featuring backward adaptive R-D optimization of quantization [39]. PC-AUTQ consistently outperforms the other coders at all test rates for both test images, except EQ. Though comparison of PC-AUTQ with EQ depends on image and rate, PC-AUTQ is very competitive with EQ in general. The results in the table are obtained from each method using the 9/7-tap filters, with an exception for EZW which uses 9-tap QMF's. Note that the algorithms of EZW and SPIHT have embedded structures to allow easier rate control and bit stream scalability while none of the other algorithms in our comparison do.

To show the compression performance of PC-AUTQ, we include, in Fig. 3.10, the original and the reconstructed images of Lena after compression at various rates. Though the images are printed at a reduced resolution, we can see that the reconstructed images offer good quality even at a very low rate of 0.15 bpp. The images are also available on the Internet at `http://sipi.usc.edu/~yyoo/pcautq/`.

Fig. 3.11 shows an interesting result by comparing PC-AUTQ with another classification-based image coder, OC-ACTCQ [27] which uses block-based VQ classification method. While OC-ACTCQ shows superior performance at high rates for the Lena image, our progressive classification achieves better performance than OC-ACTCQ at near 0.25 bpp. The performance edge of PC-AUTQ at low rate can be attributed to its efficient use of overhead rate. The classification itself is better for the forward adaptive OC-ACTCQ at the cost of large overhead. But the progressive classification allows more bits for the actual data stream, which is critical at low rates. Note that both classification-based coders use the 32-tap QMF's in this comparison.

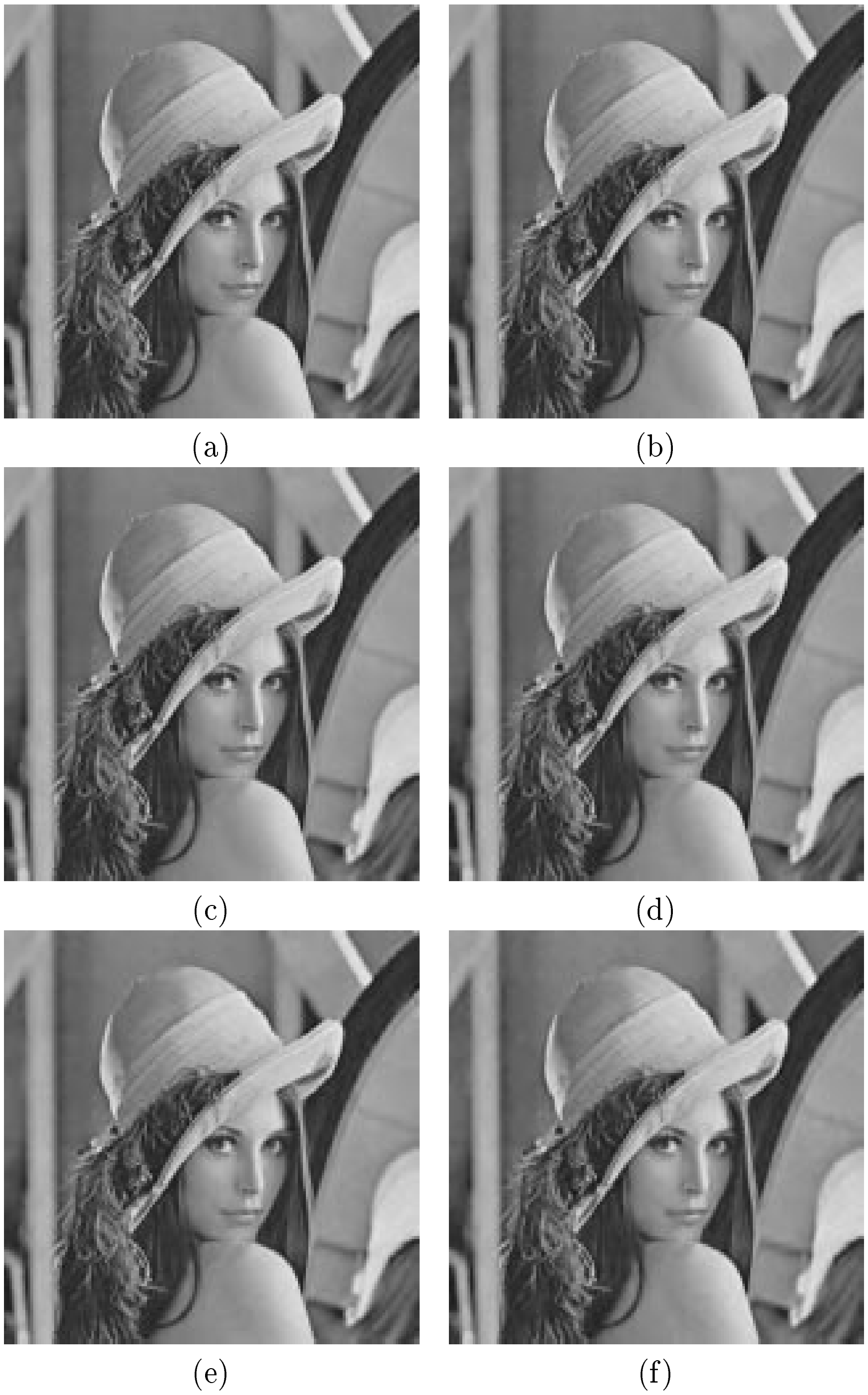


Figure 3.10: Original and reconstructed Lena images. (a) Original image (b) Rate = 1.00 bpp (PSNR = 40.75 dB) (c) 0.50 bpp (37.56 dB) (d) 0.25 bpp (34.46 dB) (e) 0.20 bpp (33.41 dB) (f) 0.15 bpp (32.08 dB)

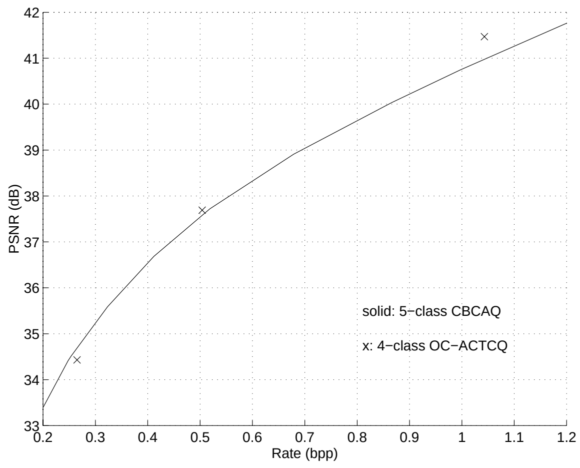


Figure 3.11: Comparison of two classification-based image coders, PC-AUTQ and OC-ACTCQ for Lena.

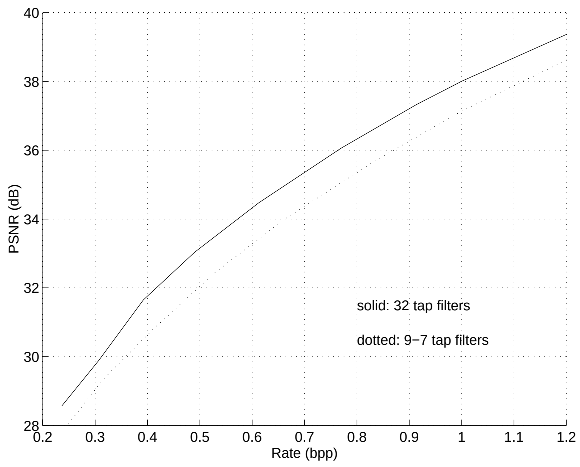


Figure 3.12: Comparison of two filters combined with PC-AUTQ for Barbara.

In Fig. 3.12 the 9/7-tap filters and the 32-tap filters are compared in our coding scheme. For the Barbara[5] image, the 32-tap filter gives far better performance, which has been reported for other subband image coders.

Finally we consider the effect of quantizer adaptation for the 9-7 filters in Fig. 3.13. We can see that quantizer adaptation offers more gain as compression is done at lower rates though the gain margin is small as compared to the D4 filter case of

[5]The Barbara image in the experiment is obtained from the UCLA Image Communications Lab's web site (`http://www.icsl.ucla.edu/~ipl/`).

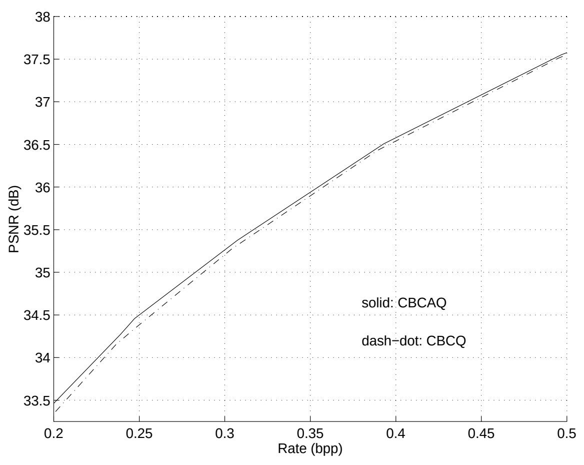


Figure 3.13: Comparison of the image coders with quantizer adaptation and without adaptation, for Lena.

Table 3.2. The encoder can selectively use quantizer adaptation depending on the quality-complexity tradeoff for the given image, filter choice, etc., since the on-the-fly quantizer adaptation technique requires no overhead but results in increased complexity.

3.5 Conclusion

We have proposed a new subband image coder with backward adaptive quantization. While developing the new coding scheme we have introduced a clean conceptual framework based on the parametric model for the distribution of image subband data, which is also simple to implement in practice.

Context-based progressive classification of subband data is the key component of our backward adaptive system. The novel classification scheme also combines a forward adaptive technique to improve the classification performance. In pursuit of efficient overhead transmission, which is essential in forward adaptation, we have assumed the Laplacian probability model for each of the classes in the image subbands.

The Laplacian parameter estimate for each class, with the thresholds for classification, reserves most of the total overhead and the resulting overhead is negligible compared to the rate allowed for the actual data bit stream.

As a part of the progressive classification scheme, we have introduced a classification threshold design algorithm. We addressed the problem of biased approximation to the quantized context by using unquantized context and thus employed a mixed context of unquantized coefficients and pre-quantized coefficients as a solution. As a result, the basic progressive classification method was modified to consider a special class for the context of all zero-quantized coefficients in addition to the ordinary classes.

We tried to further utilize the backward adaptation technique by considering on-the-fly update of UTQ. We have also derived a good heuristic estimator of the Laplacian parameter based on past quantized coefficient to make the adaptation of UTQ viable.

We have used only conventional methods for the subband decomposition and the entropy coding but still achieved a very good image coder with the proposed ideas in its quantization block. The experimental results from our PC-AUTQ show PSNR numbers competitive with those from the best image coders in the literature. Especially PC-AUTQ's performance at low rates is superior, which can be explained by an efficient use of overhead. To extend the proposed image coder, the performance improvement can be readily available by replacing the current baseline quantizer with a more versatile one, e.g., AC-TCQ [26]. More sophisticated subband decomposition can also be used to improve the current performance. Coding complexity reduction by using a table-based quantizer design/update would allow the coder to use more general distribution models (e.g., GGD) and is planned as part of our future work.

Chapter 4

Image Domain Compression of Simple Images[1]

4.1 Introduction

Wavelet transform coding methods provide excellent performance for lossy compression of natural images over all compression rates, as seen in Chapter 3. However wavelet-based methods fall short when the histogram of an image, or its subimage, has only a small number of active intensity values, i.e., a large portion of intensity values are never used within the image/subimage. We call such image objects "simple." A typical histogram of a simple image is depicted in Fig. 4.1. Examples of simple images of interest include: *bi-level images; gray-scale or color images scanned from bi-level images; computer generated graphics with simple textures, cartoons, screendumps, diagrams, etc.*

Although simple images are inherently easier to compress compared to natural images, the wavelet transform is not very effective for these images. For instance, wavelet transform coders still have to deal with a large number of wavelet coefficient values even for simple images. And strong Markov statistics in simple images, which can promote data source modeling and adaptive coding techniques, are not preserved through wavelet transform. Moreover, while the sharp edges in simple images are

[1]For related publications see [36, 73].

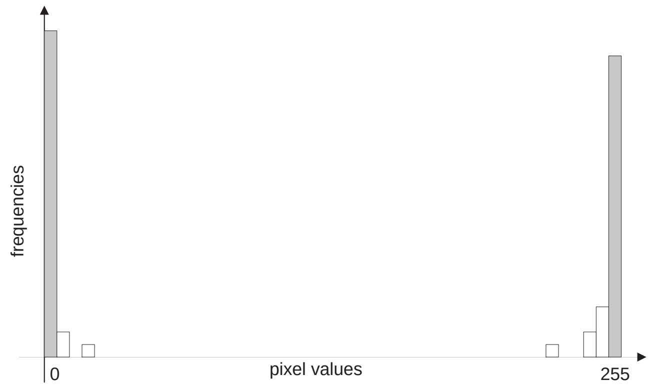


Figure 4.1: The histogram of a simple image is characterized by a few strong modes. The example shown here is a typical histogram of an 8-bit gray-scale image scanned from a bi-level image. While most pixel intensity values are either 0 or 255 (corresponding respectively to black or white pixels in the bi-level image), the scanning noise can cause a small number of pixels to have intensity values slightly different from 0 or 255.

usually important visual objects, wavelet transform coders can result in blurring sharp edges, especially when compression is lossy.

While the wavelet transform is not very effective in compressing simple images, an image domain compression algorithm can be employed as an alternative to achieve good compression. In this chapter we introduce an image domain algorithm that aims at compression of simple images. The algorithm is proposed in the context of lossless and near-lossless compression. However, even in a lossy compression environment, these types of approaches are useful. In particular, when compound images are considered, efficient performance can be achieved even with lossless or near-lossless compression of the simple image regions. We thus propose a simple, yet effective method to encode "simple" regions with lossless or near-lossless compression. Also we discuss a simple technique to separate a compound image into simple regions and natural regions so that we can apply the proposed compression algorithm to the simple regions in a compound image. Transform-based coding can be potentially used to compress the natural regions so as to optimize the overall performance, but we do not attempt to provide solutions to this subject in this chapter.

A previous work that treats the simple image/subimage with a dedicated compression algorithm is found in the course of RICOH's CREW algorithm development [81, 82]. An early implementation of CREW depends on wavelet transform to obtain good coding performance for general continuous-tone images [81]. Performance degradation of this wavelet-based coder, when applied to images having "unusual" first order Markov statistics, is considered in the improved version of CREW [82] which opts for a *binary mode.* CREW compression in the binary mode for coding is an image domain algorithm similar to JBIG [20]. This algorithm is applied to parts of the image which are deemed to be simple based on a suitable segmentation criterion. With this option, CREW has achieved significant gains in compression performance for simple images and compound images containing simple regions.

The chapter is organized as follows. In Section 4.2 we describe an image domain algorithm for efficient coding of simple images and discuss a generic image compression system in which our algorithm is used to handle simple regions. Other components of the generic hybrid image coder, *region segmentation* and *histogram compaction*, are also discussed in the same section. Experiments and results are presented in Section 4.3. We conclude this chapter in Section 4.4.

4.2 Coding of simple image/subimage

The proposed image compression algorithm is developed exclusively for simple images. For a typical simple image, only a small number of symbols are needed to encode the pixel values and the pixels tend to exhibit a strong first-order Markov statistics. While transform-based coding is inappropriate to make use of these properties, direct coding in the original image domain is not only more convenient, but also more effective for simple images. As our simple-image coding algorithm operates in the image domain, it will be referred to as *Image Domain Processing (IDP).*

We note that IDP can also be used as a part of a more general image coding system to cope with various types of images. The block diagram in Fig. 4.2 shows an

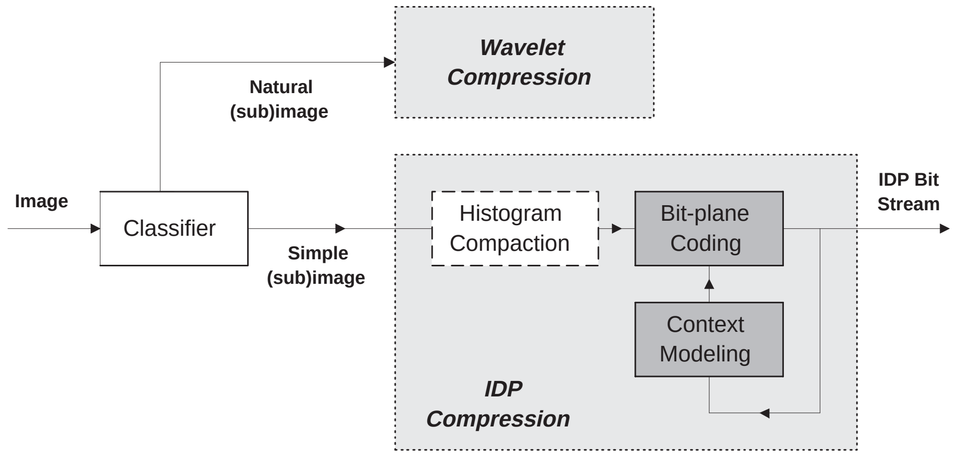


Figure 4.2: The block diagram of a generic coding system that can be consistently good for both natural and simple images. The Classifier block can be further generalized by assuming a segmentation function for the input image into natural and simple regions.

example of a generic system that uses IDP in conjunction with a wavelet coder. The coding system is intended to offer consistently good performance for natural and simple images by adaptively selecting the coding algorithm in the *Classifier* module, depending on the image type. Moreover, assuming a segmentation procedure within Classifier to divide the input image into simple and natural regions, this generic scheme can successfully compress even a compound image that consists of simple and natural regions. However, assuming wavelet transform coding applied to natural regions, it is difficult to deal with region segments of arbitrary shape. We will introduce a simple block-based segmentation procedure after describing the IDP algorithm in the following.

4.2.1 Description of algorithm

Suppose that we compress a simple image or a simple region. The compression algorithm processes each bit-plane successively, from the most significant bit (MSB) plane to the least significant bit (LSB) plane, thus achieving an embedded bit stream. See Fig. 4.3 for an example of the bit-planes obtained from an 8-bit image. A binary

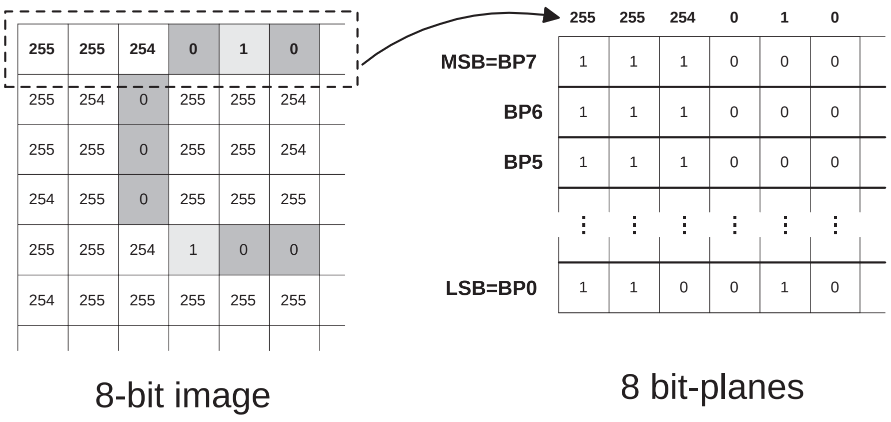


Figure 4.3: Bit-planes obtained from an 8-bit gray-scale image. In this figure one line of pixels in the image is converted into bit-planes.

adaptive arithmetic coder is used to compress each bit-plane. The conditional probability model of the adaptive arithmetic coder is instrumental for good compression performance and it is described in detail below.

Given an input bit to the binary arithmetic coder in the current bit-plane, the context model is composed of two sets of bits: one set containing neighboring bits in the same bit-plane, C_{intra}, and the other containing bits in the other bit-planes, C_{inter}. We do not want to increase overhead to estimate the conditional probability of the current symbol so that we use causal contexts, i.e., the bits from the causal past, or the bits already processed by the arithmetic coder thus known to the encoder and the decoder. Our causal context models are shown in Fig. 4.4.

The reason for considering two different sets of context bits is clear. First, the strong spatial correlation among image pixels also extends to some extent to its bit-planes and thus using C_{intra} will be beneficial. Second, C_{inter} is useful because we are running our algorithm on simple images. Consider as an example a bi-level image scanned as an 8-bit image. In this case the two more likely intensity levels are going to be $0000\,0000_2$ and $1111\,1111_2$ and, possibly, the levels that differ only in some of the LSB's. Thus, if a particular pixel's MSB is 1, the less significant bits would also be 1 with high likelihood. This motivates the fact that, for simple images,

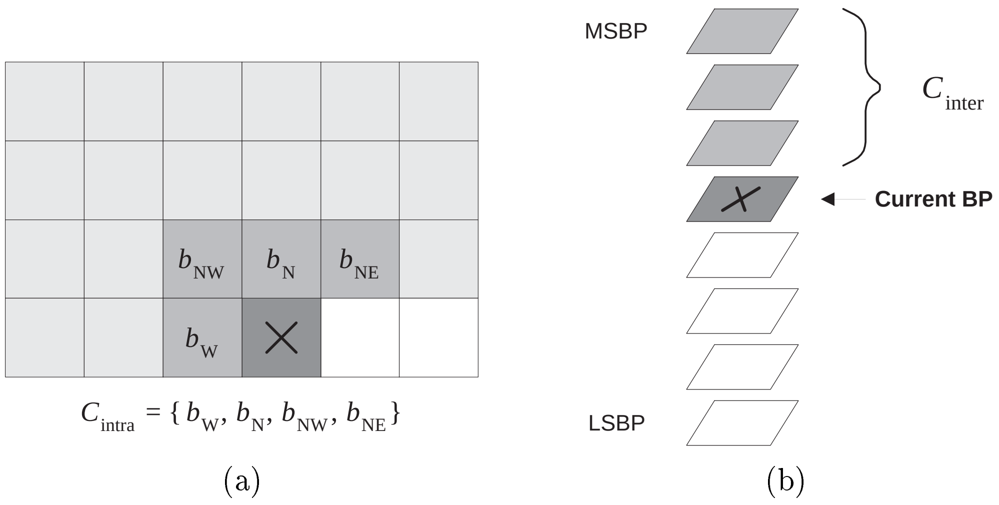


Figure 4.4: Causal contexts to generate the conditional probability model for an input bit to the binary arithmetic coder in IDP. Two sets of bits, (a) C_{intra} and (b) C_{inter}, are used to exploit the correlations among neighboring bits within the current bit-plane and across the bit-planes, respectively.

gain can be achieved by using information about the previously scanned bit-planes and thus justifies the potential benefits of C_{inter}.

Note that a similar intuition applies in the case where the original images scanned to 8-bit images are not bi-level. Since the number of gray-scale levels is much less than the maximum of 256 in those simple images, for each given pattern of, say, 4 MSB's, the number of possible combinations of 4 LSB's is likely to be less than 16 and may indeed be biased to one particular level. This aspect can be exploited through context modeling. Also note that further gains can be achieved for simple images by performing *histogram compaction*, e.g., if only 32 pixel values are used in an 8-bit image, we first map the 32 levels into a 5-bit code and then run bit-plane coding for each of the 5 bit-planes. We will discuss histogram compaction briefly later in this chapter.

Once the context model has been selected, the compression algorithm processes each bit plane in the usual raster scan order. The only necessary side information is the dimension of the input image and, when segmentation is applied, the classification map to describe the location of simple and natural regions.

4.2.2 Region segmentation

In order to separate simple regions from a compound input image, we use a segmentation procedure. The main issues in developing the segmentation algorithm are: (i) the criterion to decide whether a region is simple or natural, (ii) the amount of side information required to deliver the segmentation result, and (iii) applicability of the coding methods to the classified regions.

Although segmentation into an arbitrary region shape is optimal in extracting simple regions, the practical concerns given in (ii) and (iii) above make it difficult to use arbitrary shapes. Instead we apply segmentation over the blocks obtained by tiling the input image. Segmentation based on the image tiles can avoid the excessive side information required for shape coding. Furthermore, for the compression of the natural regions, rectangular tiles are more suitable for the popular transform coding techniques (e.g., wavelet coding or DCT coding).

Choosing an optimal segmentation criterion depends on the particular coding method for each of the simple and natural regions. While we do not specify the compression method for the natural tiles, we employ a simple segmentation rule which tests the number of active[2] levels in the image tile.

We summarize our segmentation procedure in the following. Given an M-by-N image,

Algorithm 3 *Image Tile Segmentation*

1. Partition the image into $\frac{MN}{mn}$, m-by-n tiles.

[2] *Active* levels are the pixel values that are used at least once to represent the given image tile.

2. For each m-by-n tile, generate a histogram of the pixel values to find the active levels having non-zero frequency of occurrences.

3. Compare the number of active levels L_a with a given threshold L_T. If $L_a < L_T$, then the given tile is compressed with IDP. Otherwise, apply a natural image coding method.

In the above algorithm we assume that we choose m and n such that M and N are multiples of M and N, respectively. Note that m, n, and L_T are all design parameters for the overall hybrid coding system. This scheme needs no more than mn bits as overhead for tile classification.

4.2.3 Histogram compaction

Histogram compaction can be optionally used to preprocess simple images to help the performance of our image domain compression algorithm (See Fig. 4.2). If the number of active intensity levels is small, then we use side information to specify what pixel values are in use and represent each pixel intensity with a reduced number of bits.

Suppose that only 29 active pixel values are used in a given image. Then we can represent image using only 5 bits per pixel after specifying those 29 levels explicitly in the side information. This makes our compression algorithm avoid compressing 3 bit-planes so as to increase compression efficiency. However, when to use histogram compaction must be decided carefully, taking into account the trade-off between bits added by overhead and bits saved by removing redundant bit-planes. For example, a convenient, yet efficient way of specifying the active pixel values is to use a 256-bit overhead (for 8-bit gray-scale images) where the i-th bit is set to 1 if the pixel value $i-1$ is engaged in image representation, or set to 0 otherwise. In this case, obviously, histogram compaction is not worth considering if it can save no more than 256 bits by bit-plane reduction.

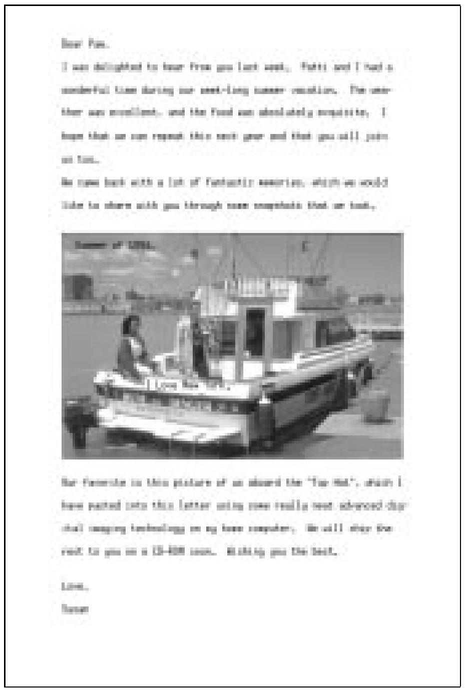
4.3 Experiments and results

For the experiment with the proposed coding scheme, we first generated simple images from some of the JPEG 2000 test images [16], `cmpnd1.raw` (512x768), `target.raw` (512x512), and `us.raw` (512x448). The selected test images are shown in Fig. 4.5. From these images, we obtained the corresponding simple images in Fig. 4.6 by using the segmentation procedure in the previous section. The tile size is fixed to $m = n = 64$. $L_T = 64$ is used for segmentation and those tiles classified as the natural regions are filled with $0000\,0000_2$'s and shown as the black tiles in the figure.

In the implementation of the IDP algorithm for experiments we choose 4 bits for C_{intra}: the adjacent bits to the current bit in W, N, NW, and NE directions. When compressing the k-th bit plane, k=0,...,7, we use k bits from Bit-Plane 0 to Bit-Plane $k-1$ in C_{inter}, where Bit-Plane 0 corresponds to the MSB's and Bit-Plane 7 to the LSB's. See also Fig. 4.4.

Lossless compression with IDP is compared to independent bit-plane by bit-plane compression using JBIG [20], the ISO/ITU bi-level image compression standard. Also we consider CALIC [68] which is known as one of the best lossless coding algorithms for the general gray-scale images. The black tile regions are also considered in compression as parts of a simple image. The results indicate that our method outperforms the other schemes for simpler images, i.e., image represented by a smaller number of pixel intensity levels. These results are summarized in Table 4.1. We also include, in the parentheses below the IDP result for each test image, the results of combining histogram compaction with IDP. Since histogram compaction on a tile-by-tile basis produces a large amount of side information, it is either used for all simple tiles or not used at all in the given image. Thus, while it increases compression efficiency for `cmpnd1.smp`, histogram compaction is automatically turned off for `cmpnd1.smp` and `us.smp` because it does not offer any reduction in bit-planes.

We then ran the same experiment on the images obtained by a different segmentation threshold: $L_T = 24$ is used instead of 64, yielding the results in Table 4.2. By



Dear Pam,

I was delighted to hear from you last week. Patti and I had a wonderful time during our week-long summer vacation. The weather was excellent, and the food was absolutely exquisite. I hope that we can repeat this next year and that you will join us too.

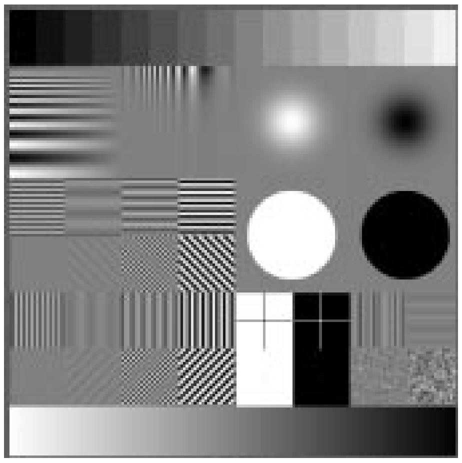
We came back with a lot of fantastic memories, which we would like to share with you through some snapshots that we took.

Our favorite is this picture of us aboard the "Top Hat", which I have pasted into this letter using some really neat advanced digital imaging technology on my home computer. We will ship the rest to you on a CD-ROM soon. Wishing you the best.

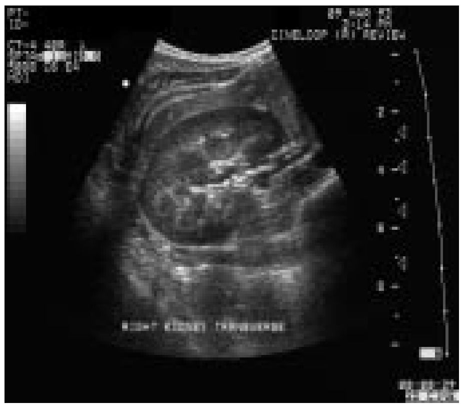
Love,

Susan

(a)

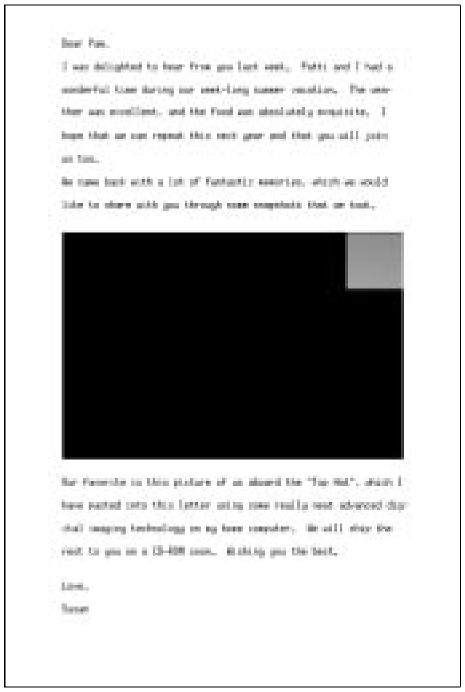


(b)



(c)

Figure 4.5: JPEG 2000 test images used in the experiment. (a) `cmpnd1.raw`, (b) `target.raw`, and (c) `us.raw`. All are 8-bit gray-scale images.



Dear Pam,

I was delighted to hear from you last week. Patti and I had a
wonderful time during our week-long summer vacation. The wea-
ther was excellent, and the food was absolutely exquisite. I
hope that we can repeat this next year and that you will join
us too.

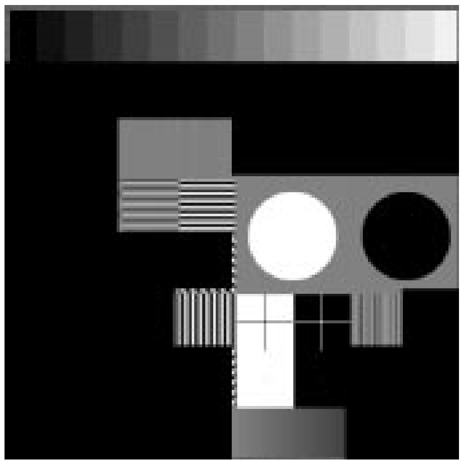
We came back with a lot of fantastic memories, which we would
like to share with you through some snapshots that we took.

Our favorite is this picture of us aboard the "Top Hat", which I
have pasted into this letter using some really neat advanced dig-
ital imaging technology on my home computer. We will ship the
rest to you on a CD-ROM soon. Wishing you the best.

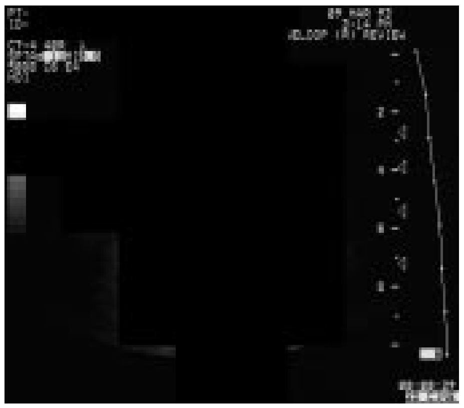
Love,

Susan

(a)



(b)



(c)

Figure 4.6: Simple images obtained by preprocessing JPEG 2000 test images in Fig. 4.5. (a) `cmpnd1.smp` is a bi-level image. (b) `target.smp` and (c) `us.smp` have no more than 64 intensity values in their 64x64 tiles.

Test Images	Original Size (bytes)	Compressed Size (bytes)		
		IDP	I-JBIG	CALIC
`cmpnd1.smp`	393,216	5,400 (**5,400**)	18,893	6,256
`target.smp`	262,144	11,539 (11,539)	**9,739**	12,686
`us.smp`	229,376	21,158 (21,158)	24,011	**19,365**

Table 4.1: Lossless compression comparison of IDP with I-JBIG (compression by independent application of JBIG to each of 8 bit-planes) and CALIC. Test images are obtained by segmentation with $L_T = 64$ and shown in Fig. 4.6. Note that `cmpnd1.smp` is a bi-level image except one 64×64 tile at the upper left corner of the original photo region. The file sizes in the parentheses in the IDP column are the results of applying histogram compaction before IDP compression.

Test Images	Original Size (bytes)	Compressed Size (bytes)		
		IDP	I-JBIG	CALIC
`cmpnd1.smp`	393,216	3,313 (**3,162**)	16,056	4,098
`target.smp`	262,144	**2,076** (2,141)	3,578	2,757
`us.smp`	229,376	9,161 (**8,385**)	11,245	8,503

Table 4.2: The results from the same experiment as the one for Table 4.1 but with different test images: the test images are obtained via segmentation with $L_T = 24$. `cmpnd1.smp` is now a true bi-level image.

using even simpler input images, we can see that IDP consistently outperforms the others. Also, histogram compaction can provide further gain for `cmpnd1.smp` and `us.smp`. However, it is interesting to observe that, for `target.smp`, the histogram compaction slightly degrades the compression performance. In this case the side information for histogram compaction overshadows the gain by IDP with the reduced bit-planes.

We also experiment with a computer-generated graphics image in Fig. 4.7. This graphics image uses as many as 151 intensity levels for the whole image and up to 115 levels locally when 64×64 tiling is applied. Tile-based segmentation is not used to preprocess the image so that the whole image is treated as a simple image. In the

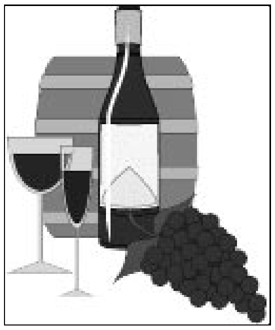


Figure 4.7: A computer generated graphics image used to evaluate the lossless performance of IDP. This 125×151 gray-scale image contains 151 intensity level for the whole image while it locally has no more than 116 levels for 64×64 tiling.

Original	IDP	I-JBIG	CALIC	GIF
18,875	**4,305**	5,596	4,729	5,254

Table 4.3: Lossless compression comparison of IDP with I-JBIG, CALIC, and GIF, for the simple computer-generated graphics image in Fig. 4.7. The numbers indicate the file sizes in bytes. Preprocessing of the image is not considered. While bit-plane reduction for histogram compaction is not available, IDP outperforms all other algorithms.

performance comparison we consider Compuserve's Graphical Interchange Format (GIF), a *de facto* lossless image compression standard, as well as I-JBIG and CALIC. In Table 4.3 the compressed file size in bytes is given for each method. As the IDP algorithm outperforms the other methods, it can be useful for compression of similar computer-generated graphics consisting of large homogeneous regions. Note that histogram compaction does not help the performance of IDP in this case since bit-plane reduction is not available.

4.4 Conclusion and remarks

We have demonstrated that the proposed IDP algorithm is more effective than CALIC or a trivial extension of JBIG to gray-scale coding, for simple images. The

IDP algorithm is also comparable or superior to CALIC and GIF, when compressing images having a moderate number of active pixel values or images consisting of large homogeneous regions. However, it is important to see that our method, unlike CALIC or GIF, is very simple and offers embedded bit streams and bit scalability so as to easily enable lossy compression. Thus, provided that there is an intelligent way to segment a compound image (e.g., `cmpnd1.raw`), this lossless compression method can be incorporated with some embedded wavelet coder for compound images, i.e., we apply the IDP algorithm to the simple tiles (e.g., the text region of `cmpnd1.raw`) and the wavelet coder to the natural image tiles (e.g., the photo region of `cmpnd1.raw`).

The IDP algorithm has been proposed to the ongoing JPEG 2000 standardization effort, as a part of the algorithm contribution of [36]. IDP is indeed used to complement the embedded wavelet coder of [35] for tiling-based coding of the compound test images. We remark that this is another example of successful use of combined backward and forward adaptation—while classification of the image tiles is run in a forward adaptive manner, a backward adaptive mode of operation can be found in IDP coding.

As a future work to extend the proposed IDP idea, we can consider more sophisticated context modeling for both C_{intra} and C_{inter}. Finally, to establish a universal image coding scheme that combines IDP and wavelet coding, we need to further investigate the segmentation procedure.

Chapter 5

Efficient Rate-Distortion Optimization Techniques with Applications to Fast SVQ Codebook Search[1]

5.1 Introduction

A key part of any efficient lossy source coder involves optimal allocation of bit rate to each of the coding units, as employed in the selected coding paradigm. This allocation problem can be classified under the general label of *budget-constrained rate-distortion optimization* which is formulated in Section 1.2.1. Examples include the traditional off-line design of vector quantizers based on optimization over representative training data [15]; more recent non-training based quantization frameworks where the encoder makes on-the-fly decisions on how many bits to assign to competing coding units in the given system [53]; and how to find the best subtree for tree-structured quantization or the best wavelet packet tree from the uniform wavelet decomposition tree [5, 29, 49]. One of the benefits of R-D optimized allocation can be found in [69] where an R-D optimization of the zerotree coding framework of [52] results in significant performance improvement.

[1]Part of this chapter represents work done jointly with Kannan Ramchandran. For related publication see [74].

Typically, an operational R-D optimization problem for practical coding systems subsumes the selection, among quantization choices available in the chosen framework, of those which minimize the coding distortion for a given bit rate budget. A key point is that the set of parameters is discrete, therefore dictating the use of discrete optimization techniques. The encoder's task is then to find a good (if not optimal) operating point from the discrete set of available operating points corresponding to quantization choices, i.e., to search through the operational R-D space for the point which has the least distortion while not exceeding the target rate. An additional requirement is obviously that the search be computationally efficient.

In this chapter we will describe efficient ways of finding this operating point for the important class of block-based coders which can change their coding parameters, i.e., quantization choices, for the composing coding units of the block. In this regard, our problem framework is similar to the case considered by Shoham and Gersho [53] with a vital distinction that we do not restrict ourselves to operating points on the convex hull of the coder's R-D characteristic. In Fig. 5.1 we repeat Fig. 1.5 to show an example of a non-convex hull solution which is *not* attainable by the Lagrange multiplier based methods of [53].

Typical examples in the scope of application of our proposed techniques include coding of intraframe (I-frame) blocks in MPEG (which typically occupy 75% of the overall budget) [34], entropy-constrained scalar/vector quantization (ECSQ/ECVQ) [11, 4], and any block-coding type framework where convex-hull operating points will not suffice in general. An illuminating example of such a framework involves the problem of optimal codebook search for scalar-vector quantization (SVQ) [30] that we have introduced early in Section 2.3.1. Also the signal decision problem in the channel decoder of the V.34 voice-band modem standard [12] would be a good candidate to apply the proposed technique, as the V.34 standard uses a dual of the SVQ design for optimal shaping of multidimensional constellations [33].

The example of the SVQ codebook search problem will guide us through the development of new discrete optimization techniques, illustrating the basic concepts.

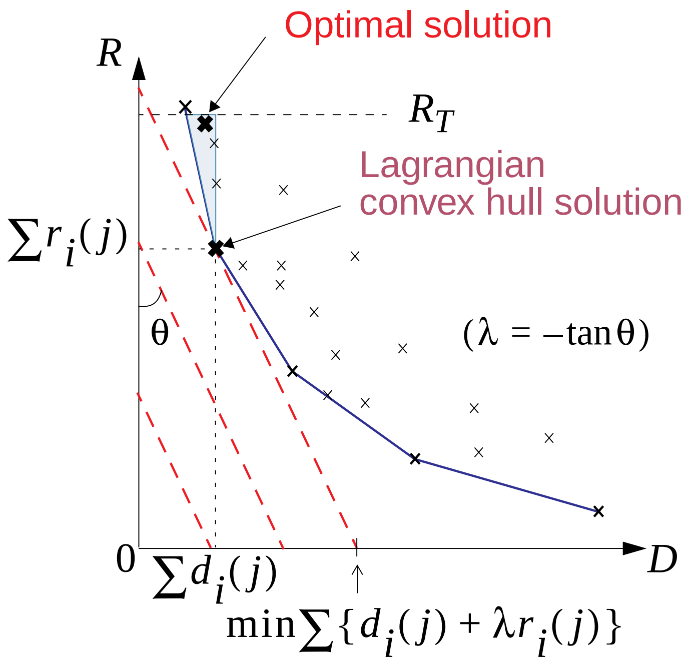


Figure 5.1: Optimal and Lagrangian solutions in the operational R-D space. The convex hull solution is determined as the first point hit by a plane wave of slope $-1/\lambda$. The D-intercept of the plane wave determines the corresponding cost. The shaded area behind the convex hull solution contains the optimal operating point that cannot be reached by this Lagrangian method.

We will present experimental results in Section 5.4 to verify the superior performance of our proposed technique over the standard optimal approach—based on dynamic programming (DP)—of [30] for SVQ codebook search.

We now provide a brief motivation for our proposed method, a hybrid between the two most popular discrete optimization techniques: Lagrangian optimization and dynamic programming.

The Lagrangian optimization technique addressed in [53] has the advantage of being fast but, as mentioned, has the drawback of achieving solutions that lie strictly on the convex hull of the available rate distortion characteristic. While this is not a major problem when the convex-hull is densely populated, sparse convex-hull sets are problematic and may result in unacceptably suboptimal performance. This can occur in practical scenarios in tree-based coding when trying to operate at low rates, as mentioned in [29]. Note that the suboptimality of the Lagrangian approach stems from the fact that the set of operating points is discrete. In a continuous optimization problem where all optimal operating points necessarily lie on the convex hull of the R-D curve, the Lagrangian approach will always yield the optimal operating point.

At the other end of the spectrum we can find the class of DP-based techniques which are guaranteed to find the optimal operating point, whether or not they are convex-hull residents (see Sec. 5.3 for details). In fact, for the SVQ codebook search problem, Laroia and Farvardin in [30] propose such a DP-based method to achieve optimal performance. The main drawback of the DP technique is that optimality comes at the price of a substantial increase in complexity, and indeed for most practical "on-line" image/video coding scenarios, dynamic programming is out of the question unless the number of candidate operating points is small or can be reduced.

Our goal is to introduce a novel hybrid method that is targeted at combining the advantages of the Lagrangian and the DP-based approaches. The core idea is to use the fast Lagrangian solution as an initial guess for the optimal solution in the DP tree search, thus reducing its complexity to a fraction of that of the regular full-search DP. We will demonstrate how the proposed technique finds application in providing a much improved way of SVQ codebook search compared to the approach of [30] based on full-search DP. Although we use SVQ as a means to demonstrate our technique, we emphasize that its scope of application is quite broad and includes any R-D optimization framework where convex hull operating points may not suffice. Indeed the scope extends beyond source coding or even engineering, as it addresses the universal problem of optimal resource allocation in a discrete optimization setting [10].

This chapter is organized as follows. In Section 5.2 we discuss the optimal SVQ codebook search using the regular DP algorithm. We also discuss the Lagrangian approach as a fast approximation technique. In Section 5.3 we describe a set of new hybrid techniques which will allow us to trade off performance and complexity, including *buffer-constrained DP optimization* and our proposed method, *Lagrangian-initialized buffer-constrained optimization*. In Section 5.4 we present the results of applying the new techniques to SVQ codebook search and compare our results with those obtained from the standard techniques. Finally, we conclude in Section 5.5 with remarks on future research topics related to this chapter.

5.2 SVQ revisited

The scalar-vector quantizer (SVQ), a fixed-rate VQ scheme, has been proposed to approximate the performance of ECSQ while being robust in noisy environments [30]. The SVQ is theoretically justified by the *asymptotic equipartition property* (AEP) for i.i.d. random variables [7]. From the AEP, a code with a fixed length close to m-times the source entropy can represent most of the sample sequences of length m with minimal error. The basic idea of SVQ is to use an underlying scalar quantizer (USQ) and associate a "length" based on the sample entropy to each of its quantization levels. Of all possible combinations of the USQ levels, only those with the total length (i.e., the sample entropy) no greater than a *threshold* are made part of the codebook. The threshold is determined such that the total number of codevectors in the codebook matches the design rate of the SVQ. The details about design of SVQ are treated in [30]. Here we focus on the codebook search issue for SVQ, i.e., the procedure to find the best codevector given an input vector and the designed SVQ codebook.

5.2.1 SVQ codebook search

The SVQ quantizes an input vector by choosing, among all combinations of USQ levels, the one which minimizes the distortion *without having the total length exceed the threshold*. More formally, an m-dimensional rate-r SVQ can be defined by its parameter triplet $(\mathcal{Q}, \mathcal{L}, L)$, i.e., (i) $\mathcal{Q}$, the set of the n ($n \geq 2^r$) levels q_j of an USQ,

Bit Allocation	**SVQ Codebook Search**
Number of input units M	Vector dimension m
Number of quantizer choices N	Number of USQ levels n
Rate caused by j-th quantizer $r(j)$	Length of j-th USQ level ℓ_j
Total rate budget R_T	Total length constraint L

Table 5.1: Analogy between SVQ codebook search and bit allocation in the general budget-constrained optimization framework. The parameters for the bit allocation application are from the R-D optimization formulation of Sec. 1.2.1. The total length constraint of SVQ codebook is equivalent to *per-vector rate constraint* mR when R is the target rate of SVQ coding.

(ii) $\mathcal{L}$, the set of the corresponding lengths ℓ_j given by the source entropy, and (iii) L, a threshold on the total length $\sum_{i=1}^{m} \ell(z_i)$ for a codevector $\mathbf{z} \equiv (z_1, \cdots, z_m)$ where $\ell(z_i) \in \mathcal{L}$ is the length of z_i. Scalar quantizing m samples is conceptually equivalent to vector quantizing an m-vector with a VQ codebook containing all n^m possible combinations in $\mathcal{Q}^m$. The SVQ codebook contains only codevectors such that

$$\sum_{i=1}^{m} \ell(z_i) \leq L \tag{5.1}$$

where L limits the codebook size ($\leq 2^{mr}$) making SVQ fixed-bit-rate. See Fig. 2.2 for an example of a 2-dimensional SVQ codebook based on the USQ for a Gaussian marginal distribution of input.

We can readily see that SVQ encoding is analogous to the allocation problem introduced in Section 1.2.1: the rate $r_i(j)$ is replaced by the length associated to each quantization level $\ell(z_i)$ and the rate budget R_T is replaced by the length threshold, L. The analogy is also summarized in Table 5.1. Note that all the described optimization techniques thus can be applied in the SVQ context. In this particular case the allocation problem results in a sparse R-D characteristic and it is thus a perfect

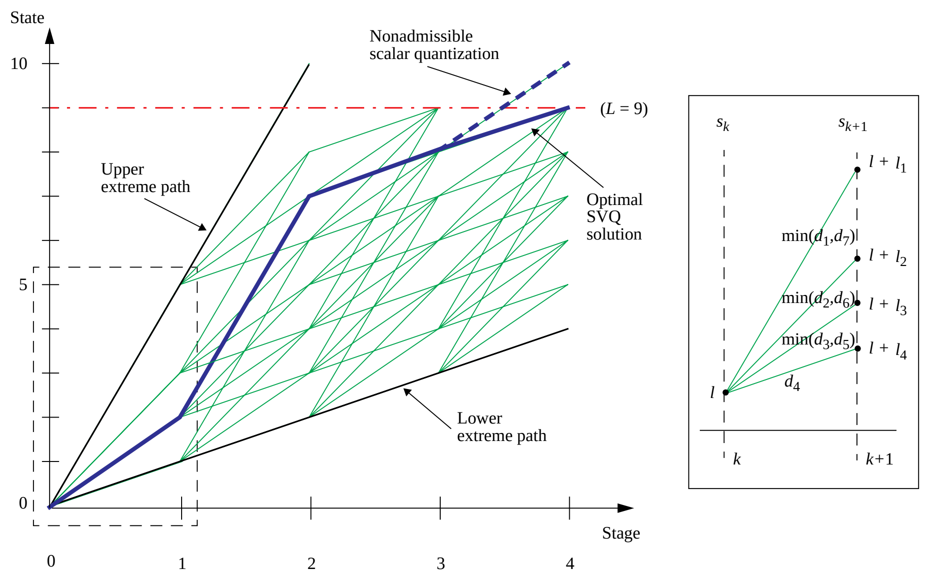


Figure 5.2: DP tree for SVQ codebook search. The whole tree is grown by attaching the possible transition given by the USQ (in the box) to each state. Each quantization choice for the i-th component of the input vector has corresponding weight (distortion, $d_1, \ldots, d_7$) and transition branch (length, $\ell_1, \ldots, \ell_7$). If some quantization levels are of the same length (e.g., $\ell_1 = \ell_7$), then the weight for such a branch is chosen as the minimum of all the distortions associated with the branch (e.g., $\min(d_1, d_7)$ as the weight of the uppermost branch). Note that the upper (*lower*) extreme path correspond to the quantization of all input components to the USQ level with the longest (*shortest*) of ℓ_j's.

testbed for the algorithms that we introduce in Section 5.3. We first describe how the SVQ codebook search problem is treated by the standard techniques.

5.2.2 Optimal codebook search by dynamic programming

As mentioned previously, the optimal SVQ codebook search proposed in [30] is a full search DP algorithm. We refer to Section 1.2.3 for the detailed procedure to grow and prune the DP tree to find the optimal solution as a path of the tree. Here we only identify the components of the DP tree for SVQ codebook search.

We redraw the DP tree of Fig. 1.6 in Fig. 5.2. Fig. 5.2 also shows the optimal codevector obtained by DP for a 4-dimensional input vector. In a separate box we include the transition diagram for state s_i. The diagram explains how the USQ $\mathcal{Q}$ acts in order to grow a whole tree. In particular, we assume that $\mathcal{Q}$ has 7 quantization levels $\{q_1, \ldots, q_7\}$ for which there are 4 distinct lengths with $\ell_1 = \ell_7$, $\ell_2 = \ell_6$, and $\ell_3 = \ell_5$. We denote the quantization distortion associated with q_j by d_j, $j = 1, \ldots, 7$. Note that the number of possible transitions (branches) is equal to the number of distinct lengths. Hence, s_{k+1} linked to s_k can have 4 different values, $\ell+\ell_1, \ldots, \ell+\ell_4$, given $s_i = \ell$.

In the tree of Fig. 5.2 the thick solid path indicates the solution codevector. The path with the thick dashed branch, deviating from the solution path at stage 3, represents the quantization corresponding to the overall minimum distortion. But it is pruned out by DP since its total length exceeds the threshold L. Note that *independent* scalar quantization by the USQ results in this non-admissible path. The solution path is associated with the minimum overall cost, among the surviving paths with the final state bounded by L, and thus guarantees optimality.

5.2.3 Fast SVQ codebook search by Lagrangian approach

The Lagrange multiplier approach prevails, as a means to approximate the optimal solution, in most constrained optimization coding problems where the computational efficiency is crucial. For example, a Lagrangian algorithm has been adopted for a fast approximation of the buffered compression in [45]. This approach can be used also in our SVQ example for efficient codebook search.

Let $\mathbf{x} = (x_1, \ldots, x_m) \in \mathbf{R}^m$ be an m-dimensional input vector. Define the cost function D as

$$D = \sum_{i=1}^{m} d(x_i, z_i) \tag{5.2}$$

where z_i is the i-th component of $\mathbf{z}$, the codevector for $\mathbf{x}$, and $d(\cdot,\cdot)$ is an additive distortion measure, typically squared error between the input and the codevector.

The corresponding codevector length $\sum_{i=1}^{m} \ell_{f(z_i)}$ must be bounded by L where $f(z_i)$ denotes the index of the quantization level corresponding to z_i.

Now we rewrite the budget-constrained optimization formulation of Section 1.2.1 in the context of SVQ codebook search. For notational simplicity, we represent SVQ codevector assignment by a mapping $v \equiv (v_1, \ldots, v_m)$ defined as

$$v : \mathbf{R}^m \to \mathcal{Z} \tag{5.3}$$

such that, for $\mathbf{x} \in \mathbf{R}^m$,

$$v(\mathbf{x}) = (v_1(x_1), \ldots, v_m(x_m)) = (z_1, \ldots, z_m) \in \mathcal{Z} \tag{5.4}$$

where $\mathcal{Z} \in \mathcal{Q}^m$ is the SVQ codebook.

Formulation 1 (Budget-constrained SVQ codebook search)
Find a mapping v which minimizes the cost

$$D(v) \equiv \sum_{i=1}^{m} d(x_i, v_i(x_i)) = \sum_{i=1}^{m} d(x_i, z_i), \tag{5.5}$$

subject to

$$R(v) \equiv \sum_{i=1}^{m} \ell_{f(z_i)} \leq L. \tag{5.6}$$

It is generally more difficult to obtain the optimal solution of constrained optimization than that of unconstrained optimization. However, with aid of the next theorem, we can justify that the solution to an unconstrained optimization is equally good for the inequality-constrained optimization in the above formulation, under a certain condition.

Theorem 1 [53] *If $v^*(\lambda)$ is the solution to the unconstrained problem*

$$\min_{v} \left\{ D(v) + \lambda R(v) \right\} \tag{5.7}$$

for a given $\lambda \geq 0$ then it is also the solution to the constrained problem in Formulation 1 with the constraint $R(v) \leq L = R(v^*(\lambda))$.

Theorem 1 implies that, for every $\lambda \geq 0$, there exists a corresponding constrained problem of which the solution is identical to that of the unconstrained problem. Thus, if $R(v^*(\lambda))$ happens to be equal to L then $v^*(\lambda)$ from the unconstrained problem becomes the optimal solution to the constrained problem. Hence the solution can be obtained by running a two-step algorithm in which we find the optimal solution to the unconstrained problem in Equation (5.7) in one step and then find a value of λ, which makes $R(v^*(\lambda))$ close enough or equal to L, in another step. The generalized Lagrange multiplier method of [10] is such a two-step algorithm which converges in λ yielding a solution

$$v^*(\mathbf{x}) = v(\mathbf{x}; \lambda^*) \tag{5.8}$$

to the new unconstrained problem.

For fast convergence in λ, Lagrangian optimization uses, for example, the bisection algorithm [48] while utilizing an optimization technique called *constant-slope optimization* to obtain the optimal solution to the unconstrained optimization problem for a given λ. Fig. 5.1 shows how the constant-slope optimization technique finds the solution SVQ codevector for a fixed λ. To briefly describe this technique, we consider a "plane wave" of slope $-1/\lambda$ which propagates from the origin toward the discrete R-D characteristics points of the SVQ, i.e., the operating points of the SVQ. The point first hit by the plane wave determines a particular quantization v which is the solution to (5.7). Note that only the points on the convex hull are the possible solutions obtained by this constant-slope method.

We include the Lagrangian optimization algorithm for SVQ codebook search in the following. For a detailed description of this optimization algorithm, as applied to the bit allocation problem and the buffer-constrained compression problem, refer to [53, 45].

Algorithm 4 *Lagrangian Optimization: SVQ Codebook Search*

1. Start with $\lambda = \lambda_l = 0$. If $R(v^*(\lambda)) \leq L$, then STOP with $v^*(\lambda)$ as the optimal assignment of codevector; Otherwise choose large enough λ_u such that

$$R(v^*(\lambda_u)) \leq L \leq R(v^*(\lambda_l)). \tag{5.9}$$

2. Set

$$\lambda_{next} = \left| \frac{D(v^*(\lambda_l)) - D(v^*(\lambda_u))}{R(v^*(\lambda_l)) - R(v^*(\lambda_u))} \right| + \varepsilon \tag{5.10}$$

   where ε is an arbitrarily small positive number to ensure the smallest rate when λ_{next} is singular.

3. Repeat the optimization of Equation (5.7) for $\lambda = \lambda_{next}$. If $R(v^*(\lambda_{next})) = L$, STOP. Else set $\lambda_l = \lambda_{next}$ if $R(v^*(\lambda_{next})) \geq L$, or set $\lambda_u = \lambda_{next}$ if $R(v^*(\lambda_{next})) \leq L$: Go to **Step 1**.

5.3 Efficient DP-based optimization techniques

Now we go back to the general budget-constrained optimization problem and develop two improved techniques based on the DP algorithm. While the first technique, *buffer-constrained DP optimization*, introduces the basic idea of limiting complexity of the optimal DP search, the second technique, *Lagrangian-initialized buffer-constrained DP optimization*, further improves the efficiency over buffer-constrained DP without compromising the optimal performance. Note that we use the same terms and notations as those used to describe the DP procedure in Section 1.2.

5.3.1 Buffer-constrained DP optimization

The fact that the number of states in the DP formulation increases linearly as a function of the DP stage number is the major obstacle to applicability of DP for

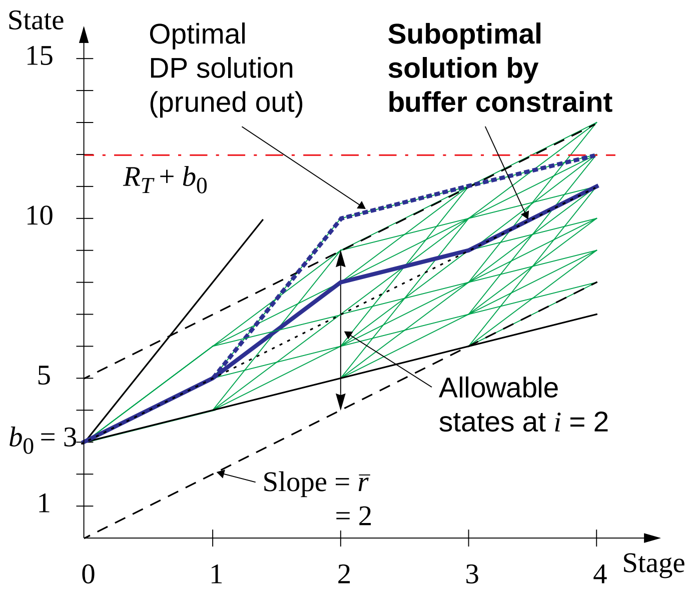

State
15
10
5
$b_0 = 3$
1
0
1
2
3
4
Stage
Optimal DP solution (pruned out)
Suboptimal solution by buffer constraint
$R_T + b_0$
Allowable states at $i = 2$
Slope = $\bar{r}$ = 2

Figure 5.3: Standard DP and buffer-constrained DP. In the standard DP case the number of states grows linearly. By introducing "buffering" constraint, we can keep the number of states at each stage constant. The states are allowed only between the buffer state bounds denoted by the dashed lines. In this example, for the chosen buffer size, the optimal solution cannot be achieved. However note that one could always find a sufficiently large buffer that would not prune out the optimal solution.

large input dimensions. Thus we first propose a method to maintain a fixed number of states for all stages in order to reduce the computational burden of DP.

Suppose that the average rate $\bar{r} = R_T/M$ (or $\bar{r} = L/m$ for SVQ codebook search) is allocated to every input unit by the solution of optimization. Then the "average" solution path through the DP tree can be defined as the one in which all branches use rate equal to $\bar{r}$. Note that this average solution path does not necessarily correspond to an actual path in the tree since $\bar{r}$ may not be integer. In Fig. 5.3 the average solution path is denoted by the hidden line of slope $\bar{r} = 2$ starting from (0,3).

Now the basic idea is to assume that most paths will not deviate too far from the average path. We introduce a new state variable b_i to represent the state of a *virtual* buffer at the i-th stage. The role of the virtual buffer is to keep track of how much each path deviates—in terms of the accumulated rate—from the average path. Since most paths tend to be close to the average path, we can impose an additional constraint so that the admissible solutions not only need to have the total rate bounded by R_T but also need to stay within certain finite buffer state bounds at every stage i. Hence we now search for the minimum distortion solution satisfying

$$b_{min} \leq b_i \leq b_{max} \quad \text{for all } i, \tag{5.11}$$

as well as the constraint (1.2), where b_{min} and b_{max} are appropriately chosen lower and upper buffer state bounds,[2] respectively. With this change, the DP technique can be used in the usual way except now with a constant number of states at each stage. Note that the number of states is equal to the chosen buffer size, $B = b_{max} - b_{min} + 1$.

The main benefit of this formulation which is analogous to the buffer constrained optimization of [45] is that the number of states can be bounded and thus the complexity is kept reasonable even for a large M. However the additional constraint may potentially eliminate the full-search optimal solution (see Fig. 5.3). In general, we can expect that this would not occur often (and thus suboptimality will be limited). Note that we can *always* find the optimal buffer size—the minimum buffer size for which the buffer-constrained solution is equal to the optimal DP solution.

Now we describe the buffer-constrained DP for the example of SVQ codebook search. We then consider a procedure to choose the optimal buffer size and discuss the complexity of the algorithm.

5.3.1.1 Buffer-constrained SVQ codebook search

For the i-th element z_i of an m-dimensional SVQ codevector $\mathbf{z} = (z_1, \ldots, z_m)$, the virtual buffer state b_i is expressed by the recursive equation

$$b_i = b_{i-1} + \ell_{f(z_i)} - \bar{r} \qquad \text{for} \quad i = 1, \ldots, m, \tag{5.12}$$

[2]We emphasize that we assume a virtual buffer only to clarify our idea. If the selected encoder has an actual buffer, a separate buffering constraint can be added to the problem formulation. Also note that b_{min} of the virtual buffer can be negative.

where b_{i-1} is the state for the previous element z_{i-1} and b_0 is the initial state. The buffer state equation (5.12) models a buffer which is emptied at a constant rate $\bar{r} = L/m$ while the buffer occupancy is incremented by the length associated with the quantization level, $\ell_{f(z_i)}$, for the input element. Then, given the buffering constraints b_{min} and b_{max}, we formulate the buffer-constrained SVQ codebook search problem as follows:

Formulation 2 (Buffer-constrained SVQ codebook search)
Find the SVQ codevector $\mathbf{z} = (z_1, \cdots, z_m)$ *for an input* $\mathbf{x} = (x_1, \cdots, x_m)$ *which minimizes the distortion*

$$D(\mathbf{x}, \mathbf{z}) = \sum_{i=1}^{m} d(x_i, z_i), \tag{5.13}$$

subject to

$$\sum_{i=1}^{m} \ell_{f(z_i)} \leq L \tag{5.14}$$

and

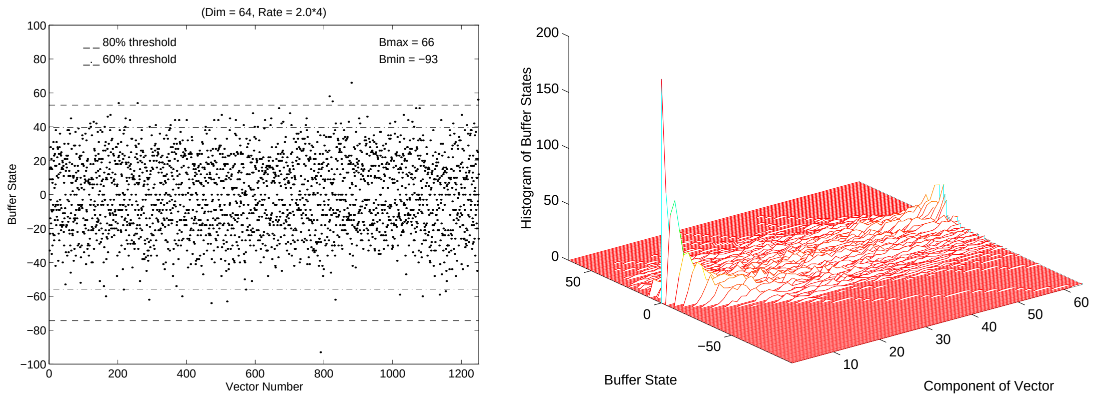
$$b_{min} \leq b_i \leq b_{max} \quad \textit{for} \quad i = 1, \ldots, m, \tag{5.15}$$

where b_0 *is the initial buffer state and* b_i *is given by* (5.12).

It is straightforward to modify the DP algorithm for the new formulation with added constraints, as shown in Fig. 5.3. In the figure $b_{max} = 2$ and $b_{min} = -3$. This particular choice emphasizes the fact that the buffer size can affect the encoding quality of SVQ. In what follows we discuss the minimum buffer size requirement for optimal or *lossless* SVQ codebook search.

5.3.1.2 Determining optimal buffer bounds

While developing the buffer-constrained formulation of budget-constrained optimization, we have assumed that the maximum and the minimum buffer occupancies do not deviate far from the initial state due to the constant buffer output. We can first verify this hypothesis with a simple experiment to track the buffer state change through quantization of a large number of input vectors.



(a) (b)

Figure 5.4: (a) Maximum and minimum buffer states for a Gaussian test sequence of 80,000 samples optimally quantized by a 64-dimensional SVQ. For each of the 1,250 vectors, the maximum and the minimum states are plotted with dots. The overall maximum and minimum buffer states for all vectors are $b^*_{max} = 66$ and $b^*_{min} = -93$. Dashed and dash-dotted lines denote 80% and 60% thresholds of b^*_{max} and b^*_{min}, respectively. (b) 3-D histogram of buffer states at each vector component. The buffer state histogram is generated for 200 (between the 751st and the 950th) vectors. Note that most the buffer states lie in $[-50, 50]$ range.

Fig. 5.4(a) is a plot of the maximum and the minimum buffer states incurred by optimal DP codebook search for a test sequence, assuming an infinite buffer. For the entire test sequence of 1,250 (64-dimensional) vectors from a Gaussian source, the buffer occupancy never exceeds $b^*_{max} = 66$ (the overall maximum) nor underflows below $b^*_{min} = -93$ (the overall minimum). Hence, if the DP codebook search is constrained with a finite buffer of size $(b^*_{max} - b^*_{min} + 1) = 160$ or larger, then the resulting buffer-constrained DP is "lossless," i.e., there is no additional distortion incurred by using the buffer-constrained algorithm instead of the optimal DP. In any case the buffer-constrained DP is expected to search the SVQ codebook much faster than the full-search DP since it considers only 160 states per stage while the full search considers $L = 518$ states per stage to meet the given rate constraint of $r = 2.0$.

m	$b^*_{max}(b^c_{max})$	$b^*_{min}(b^c_{max})$	L	SNR
8	27 (31)	-21 (-24)	74	9.982
16	36 (52)	-33 (-48)	138	10.204
32	45 (95)	-57 (-96)	265	10.347
48	56 (137)	-72 (-144)	392	10.411
64	66 (178)	-93 (-192)	518	10.416

Table 5.2: Optimal buffer state bounds necessary for lossless performance. b^*_{max} and b^*_{min} are the input-dependent optimal buffer bounds for lossless performance while b^c_{max} and b^c_{min} are the codebook-dependent optimal bounds good for arbitrary input sequences. Note that the SNR numbers are the same for both of the optimal and the buffer-constrained codebook search.

The dashed and dash-dotted lines represent buffer state bounds reduced to 80% and 60% of the optimal buffer bounds (b^*_{max}, b^*_{min}) for the lossless codebook search, respectively. We will later consider the reduced bounds in the "lossy" mode of DP to further speed up codebook search.

Fig. 5.4(b) is a 3-D histogram of the buffer states for 200 vectors from the same test sequence. The histogram can be regarded as a series of 64 2-D histograms for the 64 components of vector or the 64 stages of DP, each of which represents the usage of the buffer states at the corresponding stage. Although the histogram has non-zero values for the extreme buffer states like 66 and -93, as depicted by Fig. 5.4(a), most buffer states are populated in the range of $[-50, 50]$ throughout the entire stages. This observation implies that we could further accelerate the buffer-constrained algorithm by reducing the buffer size to a certain extent, without any significant degradation from the optimal performance.

We repeat the above experiment using 8-, 16-, 32-, and 48-dimensional optimal SVQ's. The SVQ parameters $\mathcal{Q}$ and $\mathcal{L}$ and the test sequence are the same as the ones used for the 64-dimensional case of Fig. 5.4. Table 5.2 summarizes the results. Again, from the table, b^*_{max} and b^*_{min} specify the minimum buffer size requirement as $b^*_{max} - b^*_{min} + 1$ which is required for the SVQ to quantize the test sequence without altering the solution path even when the buffer is introduced. The buffer size requirements obtained from this experiment are much smaller than L, especially for larger input dimensions.

However these b^*_{max} and b^*_{min} are valid only for our test input sequence based on which they are obtained. These particular parameters may be slightly lossy or redundant (i.e., smaller or larger than the optimal bounds) for other input sequences even if the sequences are generated from the same input source. Nonetheless, it is useful to include a step to find the minimum buffer size requirement for a training set provided that the training set be large enough to represent the source statistics sufficiently well. Then the buffer size estimate from the training set can be used in codebook search for the actual inputs, assuming that the minimum buffer size requirement does not vary much among different sets of input vectors from the same signal source.

Alternatively, given a SVQ codebook (or SVQ parameters), b^c_{max} and b^c_{min} can be determined as the optimal buffer bounds of buffer-constrained SVQ codebook search, which is lossless for arbitrary inputs, as follows:

$$b^c_{min} = m(\ell_{min} - r_b) \tag{5.16}$$

$$b^c_{max} = m'(\ell_{max} - r_b) \tag{5.17}$$

with $m' = \frac{L - m\,\ell_{min}}{\ell_{max} - \ell_{min}}$ where ℓ_{max} and ℓ_{min} represent the largest and the smallest lengths in $\mathcal{L}$, respectively. Then, with $b_{max} = b^c_{max}$ and $b_{min} = b^c_{min}$, the "codebook-dependent" buffer-constrained algorithm can find optimal SVQ codevectors for any input vectors although the improvement in search complexity is not as significant as the improvement by the "input-dependent" optimal buffer bounds. We compare the codebook-dependent bounds (b^c_{max}, b^c_{min}) and the input-dependent bounds (b^*_{max}, b^*_{min}) in Table 5.2.

5.3.1.3 Complexity Analysis

The computational complexity of SVQ codebook search using the full-search DP and the buffer-constrained DP can be assessed in terms of the number of USQ levels n, the dimension of the vector m, the length constraint L, and the buffer size $B = b_{max} - b_{min} + 1$. The number of distinct code lengths for the USQ also affects the complexity of codebook search. For simplicity in determining the complexity of the DP algorithms, we use n for this number—although it is effectively less than n—assuming that each USQ level has a different code length.

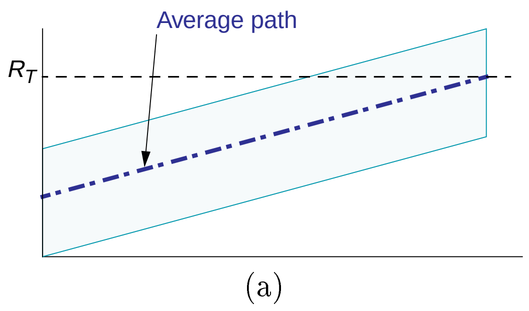
For the optimal DP codebook search, there are L nodes per stage so that a total of mL nodes has to be considered for the complete sequence. To grow a branch to each individual node (i, s_i), we choose the branch with minimum cost among at most n branches arriving to that node, which requires n comparisons per node. Since the cost associated with an incoming branch is computed as the sum of the cost accumulated in the path up to the previous stage and the cost of the branch itself, the complexity for each node increases linearly with n. Once DP has grown the full tree, determining the optimal path involves backtracking from the final stage to the initial stage. This requires one addition per stage, which is negligible in estimating the order of the complexity. Thus the overall complexity of the optimal DP is approximated as

$$\mathcal{C}_{opt} = \mathcal{O}(nmL). \tag{5.18}$$

The complexity of the buffer-constrained DP depends on the buffer size B instead of L since now the algorithm considers only B states per stage. Then we modify (5.18) to calculate the complexity of the buffer-constrained case as

$$\mathcal{C}_b = \mathcal{O}(nmB). \tag{5.19}$$

Table 5.2 shows that the buffer size B can be less than 1/3 of the threshold L for large m in a typical case. Hence, from (5.18) and (5.19), we can expect



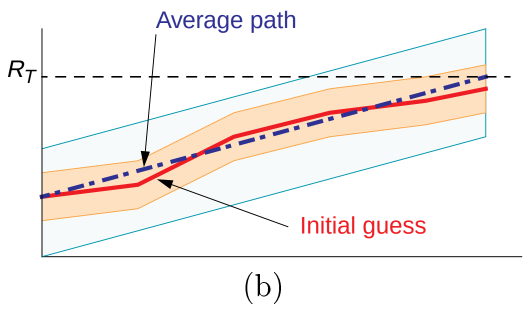


Figure 5.5: Approximated solution paths assumed in two hybrid DP optimizations. (a) Buffer-constrained DP optimization assumes an average solution path of linearly changing states with slope $\bar{r}$. (b) Lagrangian-initialized buffer-constrained DP optimization uses the path corresponding to Lagrangian convex-hull solution as the guiding path of DP search. By using a close approximation to the optimal solution, this requires smaller buffer size than the average solution path case, leading to improved search speed.

reduction in the SVQ codebook search time by a factor of 3 for large m using the buffer-constrained DP. This will be verified through the experiments in Section 5.4.

5.3.2 Lagrangian-initialized buffer-constrained optimization

We finally propose a hybrid optimization technique as the major achievement in the chapter. The new technique is based on the intuitively obvious fact: even though a convex hull solution may not be optimal it will typically be *close* to the optimal solution. We thus first run the Lagrangian algorithm and then use this solution as our initial guess to guide the DP search. This is accomplished by employing the buffer-constrained optimization introduced in Section 5.3.1 *but now using the Lagrangian solution as the virtual channel rate instead of* $\bar{r}$. See Fig. 5.5 for how the virtual channel rate assumption alters the average solution path to approximate the optimal solution.

More formally, let $r_i(\lambda)$ be the optimal rate assignment by the Lagrangian technique for the value of λ which results in a total rate closest to (but smaller than) the budget. With a new virtual buffer state variable b'_i for block i, we can apply the buffer-constrained optimization technique of Section 5.3.1 to find the optimal solution with appropriately chosen buffering constraint

$$b'_{min} \le b'_i \le b'_{max}, \quad i = 1, \ldots, m. \tag{5.20}$$

In this manner we are bounding the solution to be *close* to the Lagrangian solution.

To apply this new technique to the SVQ codebook search example, we first replace the buffer state equation of (5.12) by

$$b'_i = b'_{i-1} + \ell_{f(z_i)} - \bar{r}_i(\lambda), \quad i = 1, \ldots, m. \tag{5.21}$$

Then, with a new buffering constraint of

$$b'_{min} \le b'_i \le b'_{max}, \quad i = 1, \ldots, m, \tag{5.22}$$

in place of the constraint of (5.15), we can directly use Formulation 2 and the buffer-constrained DP method to obtain solution codevectors.

For a sufficiently large buffer size, we can guarantee that the optimal solutions are always achievable with this new buffer-constrained DP algorithm. The important point to note is that *the required buffer size to guarantee optimality will typically be small* since the Lagrangian technique already gives a good approximation to the optimal solution. However a drawback of the new method is that the codebook-dependent optimal buffer bounds are not available in general. Thus practical use of the Lagrangian-initialized buffer-constrained DP may have to resort to the input-dependent optimal bounds based on a training set. We have observed from an extensive simulation of the new algorithm for SVQ codebook search that the input-dependent buffer size derived from a large enough training set is *practically* optimal in finding codevectors. Note that our optimization technique can be favorably compared to the improvement ideas for Lagrangian optimization shown in [53], which can access non-convex hull points but cannot guarantee to achieve the optimal points.

5.4 Experimental Results

We now compare four SVQ codebook search schemes which represent the four budget-constrained optimization algorithms discussed in this chapter. The algorithms are:

(i) Optimal (full search) DP (SVQ), the method originally proposed in [30].

(ii) Lagrangian approximation (L-SVQ).

(iii) Buffer-constrained DP (B-SVQ).

(iv) Lagrangian-initialized buffer-constrained DP (LB-SVQ).

Note that our experiments involve only the codebook search, and not the codebook design.

Results are provided for a sequence of 80,000 i.i.d. samples having Gaussian distribution $N(0, 1^2)$. In all four schemes we use identical SVQ parameters $(\mathcal{Q}, \mathcal{L}, L)$ designed for a distinct training set of 40,000 samples with $N(0, 1^2)$ using the methods in [30]. The test SVQ rate is $r = 2.0$ (bits/sample) and the number of USQ quantization levels is $n = 7$.

As discussed above the buffer size in buffer-constrained DP environments can be selected so as to guarantee the optimal solution. In the first experiment regarding B-SVQ we find b^*_{max} and b^*_{min} for the test sequence in order to verify that the encoding is not affected by the appropriately chosen buffer size. The SVQ parameters are the same as the ones used for Table 5.2 while the test sequence is different. We summarize the results in Table 5.3. Note that, since the input sequence is changed, the values of b^*_{max} and b^*_{min} differ from the data in Table 5.2 but only slightly. This observation justifies use of the buffer parameters found for the training set to quantize different input vectors.

In the next experiment we design b^*_{max} and b^*_{min}, together with the SVQ parameters, based on the training sequence. Then we use the parameters $(\mathcal{Q}, \mathcal{L}, L, b^*_{max}, b^*_{min})$

m	L	**SVQ**		**B-SVQ**			
		SNR	Time	b^*_{max}	b^*_{min}	SNR	Time
8	74	10.001 (10.11)	0:22	27	-21	10.001	0:14
16	138	10.225 (10.33)	0:49	35	-36	10.225	0:23
32	265	10.358 (10.40)	1:43	45	-57	10.358	0:38
48	392	10.427 (N/A)	2:49	63	-63	10.427	0:52
64	518	10.437 (N/A)	3:35	65	-72	10.437	1:00

Table 5.3: Codebook search speeds of lossless DP-based SVQ's. For the full search DP algorithm (SVQ), SNRs are compared with the data given in [6] (in the parentheses). The same SVQ parameters to those for Table 5.2 are used here while the test sequence is different. Note that the test sequence gives different buffer parameters for the buffer-constrained SVQ (B-SVQ) from the ones in Table 5.2. *Time* is the execution time to run the C implementation of each method on SUN Sparc-5, measured in *min:sec*.

m	b^*_{max}	b^*_{min}	**B-SVQ$_{100\%}$**		**B-SVQ$_{80\%}$**		**B-SVQ$_{60\%}$**		**L-SVQ**	
			SNR	Time	SNR	Time	SNR	Time	SNR	Time
8	27	-21	10.001	0:14	9.980	0:13	9.855	0:10	9.836	0:01
16	36	-36	10.225	0:24	10.221	0:21	10.168	0:19	10.102	0:02
32	48	-57	10.358	0:39	10.357	0:35	10.337	0:28	10.271	0:02
48	64	-56	10.427	0:49	10.426	0:44	10.409	0:35	10.357	0:02
64	56	-66	10.437	0:55	10.434	0:47	10.410	0:39	10.384	0:03

Table 5.4: Comparison of B-SVQ for various buffer sizes and the Lagrangian-based SVQ (L-SVQ). The SVQ parameters and test sequence are the same as the ones for Table 5.3 but now the buffer parameters are determined also from the training sequence. Buffer parameters dependent of the training set result in practically lossless quantization for B-SVQ$_{100\%}$ and only slightly lossy B-SVQ$_{60\%}$. L-SVQ shows extremely fast codebook search.

to quantize the input sequence using the buffer-constrained algorithm. To see the effect of a buffer size smaller than the minimum requirement for lossless search, we tried three different buffer sizes for the buffer-constrained SVQ. For example, B-SVQ$_{P\%}$ represents the buffered SVQ with $P\%$ of the b^*_{max} and b^*_{min} as its maximum and minimum of the allowed buffer state. See also Fig. 5.4. From the table, we can see that the performance degradation due to the reduced buffer size is negligible while the search efficiency is improved.

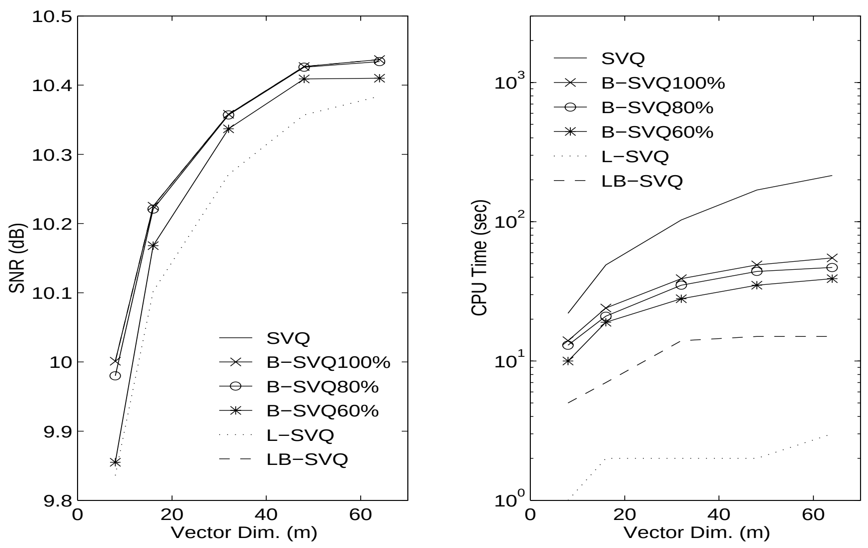


Figure 5.6: Comparison of codebook search algorithms: SVQ with optimal full-search DP, B-SVQ with buffer-constrained DP, L-SVQ with Lagrange multiplier method, and LB-SVQ with Lagrangian-initialized buffer-constrained scheme. Note that B-SVQ$_{100\%}$ and LB-SVQ achieve the optimal (DP) operating point, thus three curves are superimposed. LB-SVQ guarantees optimality with a reduction of an order of magnitude in search time. L-SVQ is the fastest scheme but is limited to non-convex hull points and thus cannot achieve optimality.

We also consider the fast approximation of the codebook search by using the Lagrangian optimization technique for which the result is shown also in Table 5.4 as L-SVQ. The experimental data in Table 5.4 are plotted in Fig. 5.6. Note that the execution time has log scale in the plot. Compared to the other schemes, codebook search by L-SVQ is extremely fast. However we can observe non-negligible performance degradation due to the suboptimality associated with the convex hull solution.

We finally experiment with LB-SVQ where we use training-set-based buffer state bounds in the buffering constraint. The performance of LB-SVQ is, as we expected, equal to the optimal performance for all vector dimensions while the reduction in search time is significant (over an order of magnitude with respect to the full search DP case), as shown in Fig. 5.6. We summarize the experimental result for LB-SVQ

m	SVQ	B-SVQ$_{100\%}$	LB-SVQ	SNR$_{\text{opt}}$
8	0:22	0:14	0:04	10.001
16	0:49	0:23	0:09	10.225
32	1:43	0:38	0:21	10.358
48	2:49	0:52	0:31	10.427
64	3:35	1:00	0:40	10.437

Table 5.5: Codebook search time comparison of optimal schemes. For optimal performance, we find the minimum buffer size requirement for the training set, i.e., b^*_{max} and b^*_{min}, for each of B-SVQ$_{100\%}$ and LB-SVQ.

in comparison with the other optimal SVQ's in Table 5.5. There is no performance loss observed for LB-SVQ throughout the experiments.

5.5 Conclusion and future work

We have introduced new budget-constrained optimization tools based on the standard techniques of Lagrangian optimization and dynamic programming. Using the first of two proposed technique, *buffer-constrained DP optimization*, we can guarantee the optimal performance of DP with improved search efficiency. Then we have established an intelligent way to constrain DP search with aid of suboptimal solutions instantly available from Lagrangian optimization. The resulting *Lagrangian initialized buffer-constrained DP optimization* provides a practical means to obtain optimal performance in source coding problems which can be formulated as budget-constrained R-D optimization.

SVQ codebook search has been considered as an example of the R-D formulated source coding problem in order to illustrate the basic ideas and the algorithms related to the standard techniques and the new techniques. Convex hull solutions of SVQ codebook search are found to result in noticeable degradation in SNR performance compared to optimal DP solutions so that Lagrangian approximation is less attractive in this particular example of R-D optimization, regardless of its unsurpassed advantage in computational complexity. Through application to SVQ codebook search in Section 5.4, we have demonstrated that our proposed optimization techniques are very helpful in achieving both optimal performance and fast search when they are not simultaneously available from the standard techniques. Especially, LB-SVQ codebook search, which is based on Lagrangian-initialized buffer-constrained optimization, combines the speed of Lagrangian approach with the optimality of DP and successfully extracts the "best of both worlds."

As we have developed the novel optimization techniques in the general framework of budget-constrained discrete optimization, we can apply our proposed techniques to a broad class of source coding and other allocation problems where essential performance degradation from the Lagrangian method is expected. Thus extended application of our optimization technique to other relevant optimization problems is our main interest in the future work.

Chapter 6

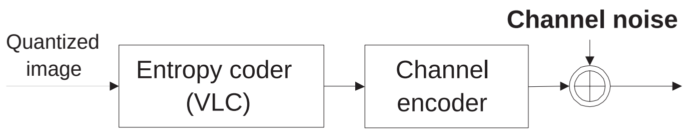
Constrained Bit Allocation for Error Resilient JPEG Coding[1]

6.1 Introduction

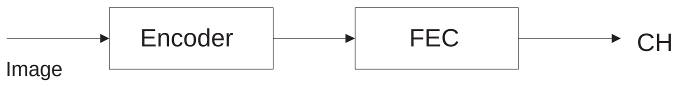
Robustness in the face of channel errors is always a major concern for many image coding systems which employ entropy coders utilizing variable-length codes or run-length codes. The traditional approach to increase the robustness of these image transmission systems has been to use error protection at the bit level, i.e., to add redundancy to the variable-length/run-length coded bit stream. Recently, there has been renewed interest in methods that allow increased robustness through use of image domain techniques, i.e., introducing the error robustness features at the encoder rather than on the encoded bit stream. Fig. 6.1 shows two different error protection image coding schemes. Note that an image domain approach can be combined with the bit level error protection method and, in this case, the redundancy by bit level protection can be reduced.

The image domain method can be potentially more beneficial than a bit level error protection since error protection can be more intelligent by understanding the contents or meaning of bits in the bit stream. Examples of the image domain

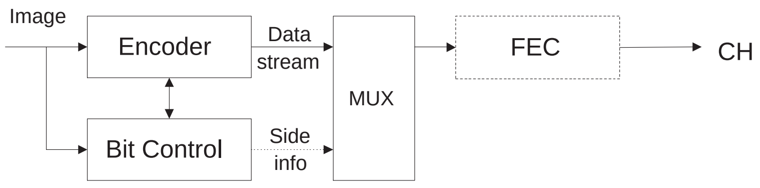
[1]For related publication see [72]. Part of the simulation program code (the R-D optimal thresholding code) is provided by Matthew Crouse and Kannan Ramchandran.



(a) Entropy coder with noisy channel



(b) Bit level approach



(c) Image domain approach

Figure 6.1: Bit level approach vs. image domain approach for error protection in image coding systems. (a) When an image coder generates variable-length codes via entropy coding, bit errors by channel noise during transmission can cause a loss of synchronism at the decoding end. (b) The bit level approach introduces redundancy through, e.g., forward error correction (FEC) in the channel encoder. The level of protection can be adapted to the channel condition, but not to the contents of the coded image. (c) Depending on the image content, the bit control module interacts with the encoder to produce bit stream inherently robust to the transmission errors. The image domain approach can lower the necessary redundancy by FEC.

approach include adding redundancy in the image domain as in [28], where an additional row of blocks (obtained by averaging the existing DCT blocks along the column direction) is added to provide H.263 video sequences with error resiliency. An alternative approach, exemplified by [80], is based on transmitting image domain side information to enhance the error concealment process.

Although we focus on the case of JPEG image coding, we can carry out similar analyses for other image/video coding algorithms. The major concern in a lossy environment is that, due to the use of variable-length entropy codes, a single bit error may cause a complete loss of synchronism at the decoder, and thus potentially result in a completely unusable picture from the point of the bit error. Bit errors in

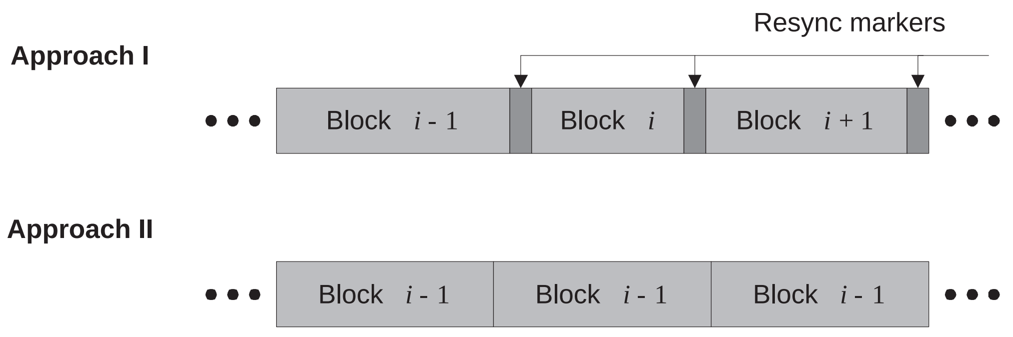


Figure 6.2: Bit streams obtained from two different approaches to limit error propagation in variable-length codes. Approach I uses explicit resynchronization markers (e.g., JPEG standard). At the cost of resynchronization overhead, it allows bit allocation over blocks of coded bits. Approach II uses blocks of the same number of bits so that the decoder determines the *end of block* without side information. However, coding suffers from the penalty by the "constant rate" constraint in this case.

a variable-length coded bit stream are bound to result in quality degradation, and thus the goal of an error protection algorithm should be first to correct as many errors as possible and second to limit propagation of the effect of errors.

In this chapter we consider approaches to limit error propagation in a variable-length coded bit stream. Two basic approaches have been previously proposed, namely,

1. Insertion of explicit synchronization markers in the bit stream; and

2. Use of constant rate for blocks of data.

In Fig. 6.2 we compare the bit streams resulting from these two approaches.

An example of the first approach is found in the JPEG standard [47], where a reserved word (i.e., the resynchronization marker) is inserted in between sets of encoded DCT blocks at a pre-specified interval. The decoder attempts to locate the resynchronization markers so that if an error occurs its propagating effect will be limited to at most the span between two consecutive synchronization points. While it is possible that at high error rates the decoder would still lose synchronization, by being unable to recognize the synchronization word, this event will tend to be rare. To achieve sufficient robustness it will be necessary to have resynchronization markers that are frequent (to limit the maximum length of error propagation) and sufficiently long (for example 2 bytes of length as in JPEG). At the same time, these requirements also point to the major drawback of explicit synchronization, namely the substantial overhead required by the resynchronization markers (up to 0.25 bit per pixel in JPEG if each resynchronization marker is placed at the end of every DCT block!).

An alternative to the explicit synchronization technique consists of imposing constraints on the output of the encoder so that each block of data has a constant (and known) total number of bits. This approach could be used in conjunction with a standard decoder based on explicit synchronization, since a preprocessor of the decoder can introduce the synchronization points into the received bit stream. Scalar-vector quantization (SVQ) [30, 31] is an example of this approach, as is the algorithm of [79]. In both cases blocks of scalar input data are coded such that the overall rate of each block is constant. In [30, 31] constant rate is achieved with a vector encoder where the vector codebook consists of the codewords with the same length. In [79] the encoder can assign one of several possible reproduction levels to each input (with a corresponding variable length code designed by the Huffman procedure). The encoder selects the quantization levels for each input such that the overall distortion is minimized and the rate for the whole block is constant. This constrained optimization can be achieved using dynamic programming.

In this chapter we apply the constrained bit allocation approaches to a practical image coding scenario. We deal mainly with a JPEG-like DCT-based coding scheme. In the description of our algorithm we will call *block* the 8x8 DCT block and *group of blocks* (GOB) the set of consecutive DCT blocks over which the rate constraint has been imposed (or which are separated by resynchronization markers). See Fig. 6.3.

The approaches such as [30, 31] or [79] rely on allocating a fixed number of bits to each set of inputs. Thus, given the standard JPEG tables and a choice of quantization scale, the algorithm in [79] can be used to encode a GOB to meet a required

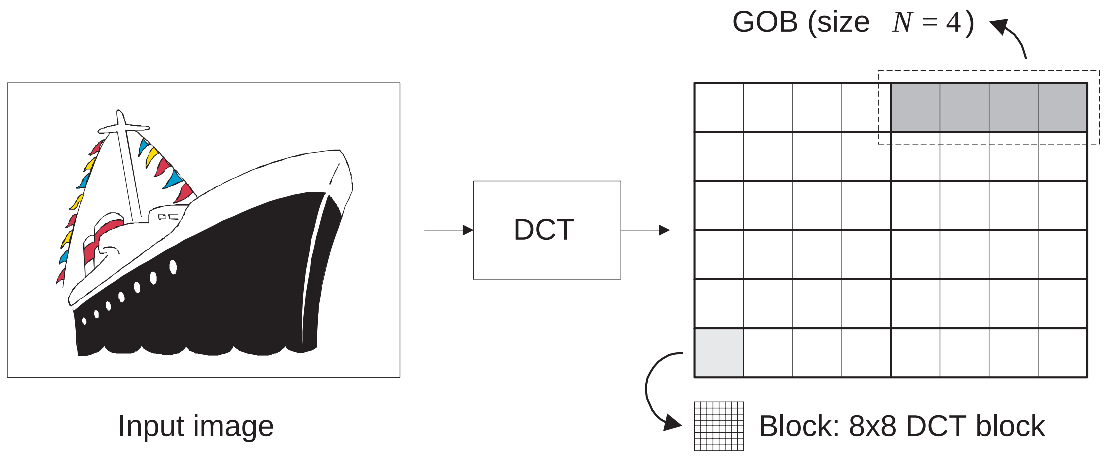


Figure 6.3: Definitions for our constant GOB rate coding. A *block* is an 8×8 DCT block and a *group of blocks* (GOB) is a set of consecutive blocks over which the rate constraint is imposed. The GOB size (N) is the number of blocks in a GOB.

number of bits. However, in practice, using the same number of bits for every GOB in an image would result in poor image quality. Conversely, making the GOB size large (e.g., a complete row of image blocks) would not affect much the image quality but would result in reduced error protection capabilities (e.g., one complete row of blocks could be lost). We propose methods to allow a variable number of bits to be allocated to each GOB, while still preserving the resynchronization properties of the constrained allocation with manageable overhead.

The remainder of this chapter is organized as follows. In Section 6.2 we discuss our constrained bit allocation approach for error robustness of JPEG coded images. The details of our proposed system is presented in Section 6.3 and the results from experiments in Section 6.4. We conclude this chapter in Section 6.5.

6.2 Constrained bit allocation for error robustness

Image coding techniques based on the DCT naturally tend to assign a different number of bits to each image block depending on its frequency contents. This is true if a single quantizer/entropy coder is used for the whole picture, as in the baseline JPEG, but also if an optimal R-D based allocation is performed based on a discrete set of quantizers [53].

Assume that we start with a particular allocation which is given by using a fixed quantization table as in the baseline JPEG, or by a constant quality slope as in [53] for a discrete set of quantizers. Resynchronization markers at the boundaries between GOBs will introduce overhead. This overhead would have to be compensated by a corresponding reduction of the coding rate. As mentioned above, this overhead can be significant (e.g., a 2-byte word per GOB in JPEG). Thus we propose to not use explicit resynchronization markers but instead transmit *overhead to specify the allocated rate of each GOB*. This is equivalent to using a *position code* that specifies the starting points of GOBs in the coded bit stream.

For simplicity let us assume that all GOBs have the same size, N. Consider an image with M blocks and let R_i be the i-th GOB rate, i.e., the number of bits allocated to the i-th GOB. We will transmit all R_i, $i \in \{1, \ldots, \lceil M/N \rceil\}$, as side information. If at most 2^b different values of R_i occur the overall overhead will then be $b\lceil M/N \rceil$ in bits, by using a fixed length code for the GOB rate overhead. As in other cases, the overhead can be reduced by letting the size of the GOBs grow, with corresponding reductions in error robustness.

A trivial implementation of this system would be to encode an image with a predetermined algorithm and then transmit the resulting sizes R_i as side information. However, in the extreme case where the GOB size is one, the overhead of sending the GOB rates would be significant and thus it will be useful to introduce additional constraints that allow to reduce the overhead.

Thus we propose to introduce the selection of GOB rates in the allocation loop. That is, our goal will be to find a set of 2^b different GOB rates such that the coding overhead is minimized and the overall distortion is as close as possible to that of the original coded image.

Moreover, for an implementation based on JPEG, we also need a rate allocation algorithm which can access a set of smoothly varying optimal R-D operating points

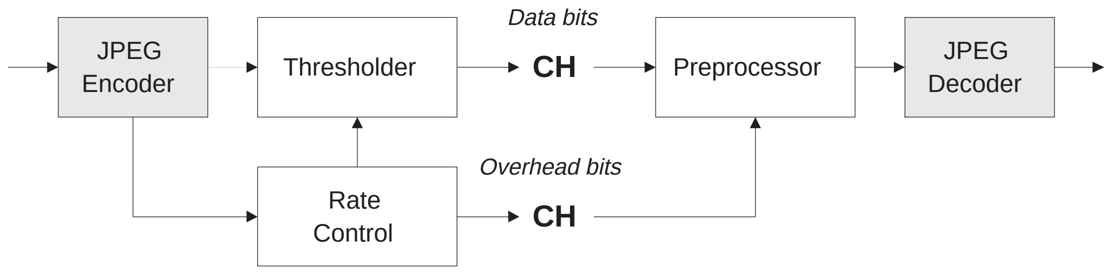


Figure 6.4: Block diagram of the proposed error-robust coding scheme for the JPEG bit stream. At the encoding end, the JPEG coded data is modified to satisfy the rate constraint through thresholding. The rate constraint is imposed by a rate control algorithm and the resulting GOB rates are sent as overhead. At the decoder, a preprocessor inserts JPEG resynchronization markers into the transmitted data bit stream based on the GOB rate side information. The output of the preprocessor is completely compliant to the JPEG decoder.

so that we can achieve the overall target rate. Otherwise we will have to rely on an iterative search for the R-D optimized JPEG quantization matrix satisfying the new reduced rate budget due to the overhead, which is not practical. Thus, in our implementation, we utilize the thresholding technique of [50, 8] to modulate the rate of each GOB and its DCT blocks. The implementation of the proposed scheme is described in detail in the next section. Then, in the experiments, we will demonstrate the advantage of our approach with respect to methods based on explicit synchronization.

6.3 Proposed system

The proposed system is depicted in Fig. 6.4. At the encoder, a two-pass algorithm generates a set of b-bit GOB rate information and a JPEG bit stream. In the first pass the input image is JPEG-encoded with the resynchronization interval equal to the GOB size. Based on the generated GOBs, b is determined as the number of the bits required to specify the largest GOB rate, i.e.,

$$b = \lceil \log_2(\max_i R_i) \rceil. \tag{6.1}$$

Then the second pass modifies the JPEG-encoded data using the thresholding algorithm (see [50, 8] for the details of the R-D optimal thresholding algorithm for the DCT coefficients) to represent the coded image with a reduced rate budget. That is, given the overall target rate R_T, the new rate budget available for allocation through thresholding (or, the rate constraint for thresholding) is

$$R_{thr} = R_T - b\lceil M/N \rceil \quad \text{(bits)} \tag{6.2}$$

where $b\lceil M/N \rceil$ is the number of reserved bits for the GOB rate side information.

Note that the encoder output can be separated into two parts: *overhead to specify the data length for each GOB* and *the JPEG bit stream without resynchronization markers/byte stuffing.*[2] Since we directly encode the DC level at the first block of GOB, rather than the differential DC level, the second part of the JPEG bit stream is not completely compliant with the JPEG format. However we can reproduce a JPEG-compatible bit stream through a simple preprocessing prior to JPEG decoding. The preprocessor parses the received data bit stream into GOBs based on the GOB rate information. Then it stuffs the last GOB byte with zeros, if necessary, and puts the resynchronization marker at the beginning of each GOB. In this way, we can use a JPEG decoder without any modification after preprocessing.

To further reduce b by limiting the allowable GOB rates, we consider a simples case of restricting the GOB rate to be an *even* number. Then the side information about the GOB rates requires a reduced precision of $b-1$ bits/GOB. However the GOBs with odd rates have to be specified as if they have one more bit (i.e., GOB stuffing to an even data length), the expected average reduction of overhead is finally settled at 0.5 bits/GOB.

[2]Since every resynchronization marker must be placed at the beginning of a new byte in the JPEG bit stream, byte stuffing is required when the number of bits used to encode the given synchronization interval (or GOB) is not a multiple of 8, the byte size in bits.

GOB Size (blocks)	b (bits)	$b\lceil N/M \rceil$ (bits)	Total overhead for error protection (bytes)	PSNR (dB)
1	6	24576	4070+	33.59
2	7	14436	2301+	34.33
4	8	8192	1312+	34.53
8	9	4608	705+	34.57
12	9	3078	459+	34.58
JPEG	-	-	-	34.58
JPEG+Resync	-	-	8192+	27.43

Table 6.1: Side rates and error-free PSNR results for various GOB sizes. b denotes the side rate to specify the GOB data length while $b\lceil N/M \rceil$ is the corresponding total overhead for the whole image. Total overhead for error protection includes the rate increase due to GOB or byte stuffing. For the result in the bottom row, JPEG encoding has a resync marker at every block.

6.4 Experiments

In the experiment we first test the proposed coding scheme on the Lena image (512×512), assuming noise-free transmission. Throughout the experiment, we fix the target compression rate at 0.5 bpp. The PSNR results for various GOB sizes are summarized in Table 6.1. We can see from the table that the proposed scheme requires only a small amount of overhead so as to yield PSNR performance close to that of JPEG coding without resynchronization. The extreme case of 1 block/GOB can be compared with JPEG coding with resynchronization markers for every block. Due to efficient use of overhead, our coding scheme retains a significant PSNR performance gain over the baseline JPEG at the same level of compression and error protection.

We then consider noisy channel environments with different bit error rates (BERs). As indicated by the plot in Fig. 6.5, our scheme offers graceful degradation in decoded image quality with worsening channel conditions. The plot also shows that coding with small GOB sizes can be particulary useful for the channel with high BERs.

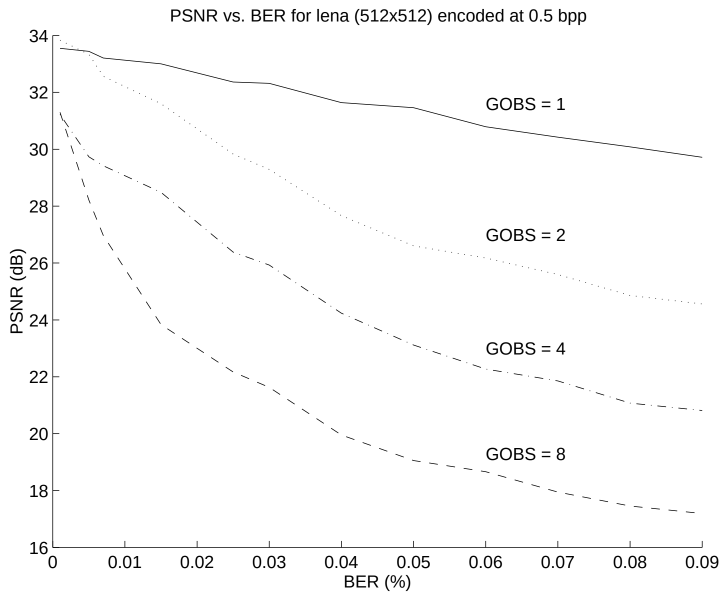


Figure 6.5: PSNR vs. BER obtained by the proposed scheme for various GOB sizes. The Lena image (512x512) is encoded at the target rate of 0.5 bpp for each simulation. The error robustness of the coding scheme yields graceful degradation of PSNR with worsening channel conditions and the small GOB sizes are particularly useful at high BERs.

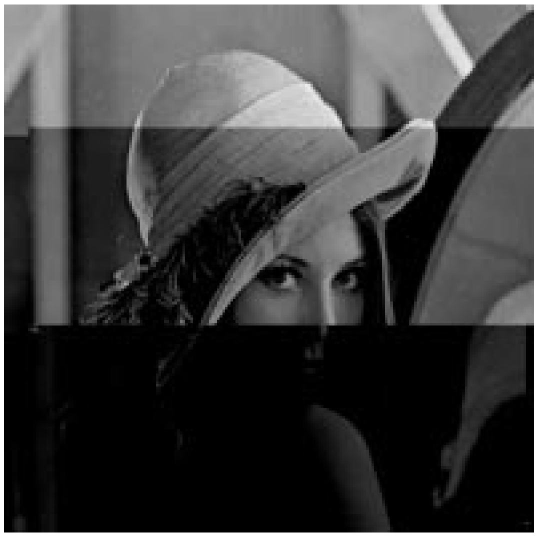
In Fig. 6.6 the decoded Lena image is compared with those from JPEG coding with and without resynchronization markers, at $\mathrm{BER} = 10^{-5}$. Fig. 6.6 (a) shows that, for JPEG coding, the effect of bit errors can be disastrous, when resynchronization markers are not used. Figs. 6.6 (b) and (c) are obtained with resynchronization at every block ($N = 1$). We can see that the proposed scheme effectively protects the error propagation across the GOB boundaries (see Fig. 6.6 (c)) while maintaining the image quality of the original JPEG image in regions with no bit error. Fig. 6.6 (b) shows how severely the JPEG coded image is degraded due to excessive use overhead for explicit resynchronization, though catastrophic propagation of bit errors is prevented.

All experimental results are based on error-free transmission of side information about the GOB rates. For typical rates of compression, the level of redundancy required for error-free overhead transmission is only modest. For example, suppose that we arrange the GOB rates R_i into a matrix according to the spatial location of the corresponding GOBs in the image. Considering the binary representation of the GOB rates using b bits per GOB, we can obtain a block-diagonal binary matrix. Then it is possible to transmit the parity check bit for each row and column of the binary matrix, i.e., the i-th parity bit for a row (column) carries the parity information of the i-th significant bits of the matrix entries in the row (column). The parity bits can correct any error patterns as long as no two or more matrix entries in the same row or column have bit errors at the same bit significance position. This would be sufficient for error levels of 5.2×10^{-3} in our experiment. Nonetheless the redundancy by the parity bits is negligible: it reaches the maximum of only 0.1875 bits/block when the GOB size $N = 1$ is used.

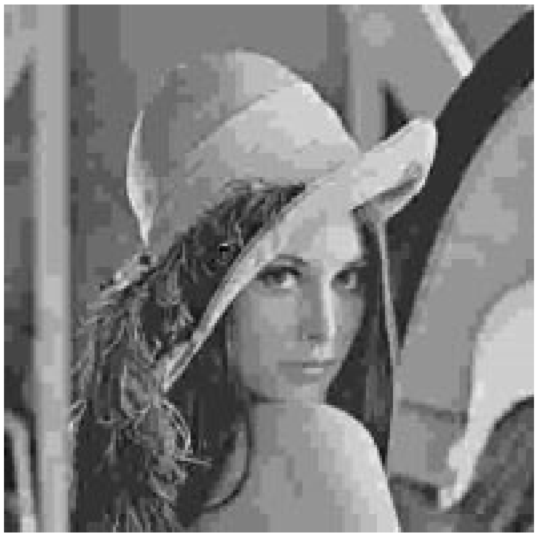
6.5 Conclusion

We have proposed a JPEG-based image coding scheme that is robust against bit errors in noisy transmission environments. In particular, we are interested in the ability to limit error propagation in the variable-length or run-length coded bit stream. In the proposed scheme the number of coded bits in each group of DCT blocks, or each GOB rate, is sent to the decoder as side information so that the decoder can parse the data bitstream into GOBs and introduce resynchronization markers at approapriate positions. The required overhead to achieve robustness is significantly less than that of the typical explicit synchronization schemes (e.g., the baseline JPEG). In addition, it is possible to further constrain the allowable rates for GOB in order to reduce the overhead. We have included experimental results where the constrained rate allocation is performed using thresholding techniques on images encoded by the baseline JPEG. The proposed scheme offers superior reconstructed image quality comparing to the baseline JPEG with resynchronization at the same level of protection against error propagation.

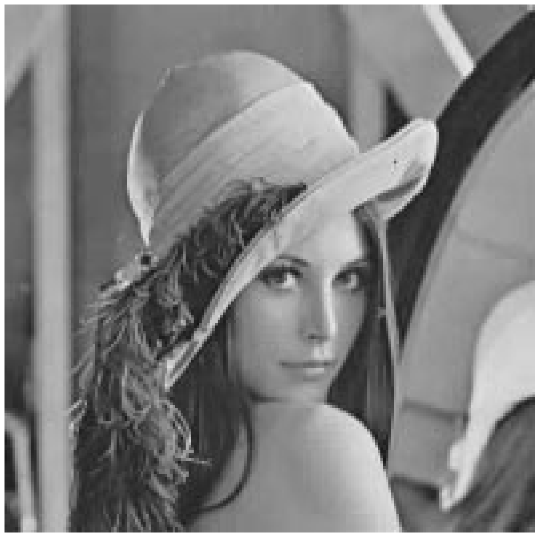
In future work we can apply the idea presented in this chapter to DCT-based video coders such as H.261/263 or MPEG. Another interesting issue is to use a more efficient technique to modulate GOB rates such as *soft thresholding* since the *hard* thresholding method in the current proposed system either keeps or rounds to zero the quantized DCT coefficients to control the GOB rate, while quantization to an intermediate level is possible. Developing an error-robust coding scheme for wavelet-based image coders is a related research topic.



(a)



(b)



(c)

Figure 6.6: Comparison of reconstructed images from corrupted bit stream with BER = 0.001%. For all three cases, the encoding rate is 0.5 bpp. (a) JPEG without resync markers (PSNR = 9.43 dB). Corrupted DPCM DC coefficients locally affect the brightness and, in the lower part, block shift is observed due to block loss. (b) JPEG with resync markers at every block (PSNR = 18.08 dB). Use of resync prevents error from propagating across blocks. To accommodate the explicit resynchronization overhead, coarser quantization is used so that the blocking artifacts are observed. (c) The proposed scheme successfully protects the bit stream from error propagation. Its efficient use of overhead allows fine quantization and good image quality (PSNR = 33.00 dB).

Reference List

[1] R. Bellman. *Dynamic Programming.* Princeton University Press, 1957.

[2] K. A. Birney and T. R. Fischer. On the modeling of DCT and subband image data for compression. *IEEE Trans. on Image Proc.*, 4(2):186–193, Feb. 1995.

[3] R. W. Buccigrossi and E. P. Simoncelli. Image compression via joint statistical characterization in the wavelet domain. GRASP Laboratory Technical Report #414, University of Pennsylvania, May 1997. Available at `http://www.cis.upenn.edu/pub/eero/buccigrossi97.ps.gz`.

[4] P. A. Chou, T. Lookabaugh, and R. M. Gray. Entropy-constrained vector quantization. *IEEE Trans. Acoustics, Speech, and Signal Processing*, 37:31–42, Jan. 1989.

[5] P. A. Chou, T. Lookabaugh, and R. M. Gray. Optimal pruning with applications to tree-structured source coding and modeling. *IEEE Trans. Inform. Theory*, 35:299–315, Mar. 1989.

[6] C. Chrysafis and A. Ortega. Efficient context-based entropy coding for lossy wavelet image compression. In *DCC, Data Compression Conf.*, Snowbird, UT, Mar. 1997.

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

[8] 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.

[9] I. Daubechies. Orthonormal bases of compactly supported wavelets. *Commun. on Pure and Appl. Math.*, 41:909–996, Nov. 1988.

[10] H. Everett. Generalized Lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research*, 11:399–417, 1963.

[11] N. Farvardin and J. W. Modestino. Optimum quantizer performance for a class of non-Gaussian memoryless sources. *IEEE Trans. Inform. Theory*, IT-30:485–497, May 1984.

[12] G. D. Forney, Jr., L. Brown, M. V. Eyuboglu, and J. L. Moran III. The V.34 high-speed modem standard. *IEEE Communications Magazine*, 34(12):28–33, Dec. 1996.

[13] R. S. Gallager. Variations on a theme by Huffman. *IEEE Trans. Inform. Theory*, IT-24(6):668–674, Nov. 1978.

[14] A. Gersho and D. J. Goodman. A training mode adaptive quantizer. *IEEE Trans. Inform. Theory*, IT-20:746–749, Nov. 1974.

[15] A. Gersho and R. M. Gray. *Vector Quantization and Signal Compression.* Kluwer Academic Press, Boston, MA, 1992.

[16] ISO/IEC JTC1/SC29/WG1. Call for Contributions for JPEG2000 (JTC 1.29.14, 15444): Image Coding System. ISO/IEC JTC1/SC29/WG1 N505, 1997.

[17] ISO/IEC JTC1/SC29/WG1. WG1 Draft Meeting Report, 12th Sydney Meeting. ISO/IEC JTC1/SC29/WG1 N723, Dec. 1997.

[18] ITU-T Recommendation H.261. Video Codec for Audiovisual Services at *p*x64 kbit/s, 1993.

[19] ITU-T Recommendation H.263. Video Coding for Low Bitrate Communication, 1996.

[20] ITU-T Recommendation T.82. Information Technology – Coded Representation of Picture and Audio Information – Progressive Bi-level Image Compression, 1993.

[21] N. S. Jayant. Adaptive quantization with a one-word memory. *Bell Sys. Tech. J.*, 52(7):1119–1144, Sep. 1973.

[22] N. S. Jayant and P. Noll. *Digital Coding of Waveforms: Principles and Applications to Speech and Video.* Prentice-Hall, 1984.

[23] J. D. Johnston. A filter family designed for use in quadrature mirror filter banks. In *IEEE Int'l Conf. Acoust., Speech, and Signal Proc.*, pages 291–294, Apr. 1980.

[24] R. Joshi, T. Fischer, and R. Bamberger. Optimum classification in subband coding of images. In *Proc. of ICIP'94*, volume 2, pages 883–887, Austin, Texas, Oct. 1994.

[25] R. L. Joshi, V. L. Crump, and T. R. Fischer. Arithmetic coded (uniform) trellis coded quantization. In *Proc. of The Second International Workshop on Intelligent Signal Processing and Communication Systems*, pages 274–279, Sep. 1993.

[26] R. L. Joshi, V. L. Crump, and T. R. Fischer. Image subband coding using arithmetic coded trellis coded quantization. *IEEE Trans. on Circ. and Syst. for Video Tech.*, 5:515–523, Dec. 1995.

[27] R. L. Joshi, H. Jafarkhani, J. H. Kasner, T. R. Fischer, N. Farvardin, M. W. Marcellin, and R. H. Bamberger. Comparison of different methods of classification in subband coding of images. *IEEE Trans. on Image Proc.*, 6:1473–1486, Nov. 1997.

[28] M. Khansari and V. Bhaskaran. A low-complexity error-resilient H.263 coder. In *Proc. of ICASSP '97*, Munich, Germany, Apr. 1997.

[29] S.-Z. Kiang, R. L. Baker, G. J. Sullivan, and C.-Y. Chiu. Recursive optimal pruning with applications to tree structured vector quantizers. *IEEE Trans. on Image Proc.*, 1:162–169, Apr. 1992.

[30] R. Laroia and N. Farvardin. A structured fixed rate vector quantizer derived from a variable-length scalar quantizers: Part I - Memoryless sources. *IEEE Trans. Inform. Theory*, 39:851–867, May 1993.

[31] R. Laroia and N. Farvardin. A structured fixed rate vector quantizer derived from a variable-length scalar quantizers: Part II - Vector sources. *IEEE Trans. Inform. Theory*, 39:868–876, May 1993.

[32] R. Laroia and N. Farvardin. Trellis-based scalar-vector quantizer for memoryless sources. *IEEE Trans. Inform. Theory*, 40:860–870, May 1994.

[33] R. Laroia, N. Farvardin, and S. A. Tretter. On optimal shaping of multidimensional constellations. *IEEE Trans. Inform. Theory*, 40(4):1044–1056, Jul. 1994.

[34] D. LeGall. MPEG: A video compression standard for multimedia applications. *Communications of the ACM*, 34(4):46–58, Apr. 1991.

[35] J. Liang. Highly scalable image coding for multimedia applications. In *ACM Multimedia Conference*, Seattle, WA, Oct. 1997.

[36] J. Liang, C. Chrysafis, A. Ortega, Y. Yoo, K. Ramchandran, and X. Yang. The predictive embedded zero-tree wavelet (PEZW) coder: A highly scalable image coder for multimedia application. ISO/IEC JTC1/SC29/WG1 N803, Nov. 1997.

[37] Y. Linde, A. Buzo, and R. M. Gray. An algorithm for vector quantizer design. *IEEE Trans. Commun*, COM-28:84–95, Jan. 1980.

[38] S. P. Lloyd. Least squares quantization in PCM. *IEEE Trans. Inform. Theory*, IT-28:127–135, Mar. 1982.

[39] S. M. LoPresto, K. Ramchandran, and M. T. Orchard. Image coding based on mixture modeling of wavelet coefficients and a fast estimation-quantization framework. In *VCIP*, San Jose, CA, Feb. 1997.

[40] S. G. Mallat. A theory for multiresolution signal decomposition: The wavelet representation. *IEEE Trans. PAMI*, 11(7):674–693, Jul. 1989.

[41] M. Marcellin and J. Rountree. Wavelet/TCQ technical details. ISO/IEC JTC1/SC29/WG1 N632, Nov. 1997.

[42] M. W. Marcellin and T. R. Fischer. Trellis coded quantization of memoryless and Gauss-Markov sources. *IEEE Trans. Commun.*, 38:82–93, Jan. 1990.

[43] J. Max. Quantizing for minimum distortion. *IRE Trans. Inform. Theory*, IT-6:7–12, Mar. 1960.

[44] J. M. Mendel. *Lessons in Digital Estimation Theory.* Prentice Hall, Englewood Cliffs, NJ, 1987.

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

[46] A. Ortega and M. Vetterli. Adaptive scalar quantization without side information. *IEEE Trans. on Image Proc.*, 6:665–676, May 1997.

[47] W. Pennebaker and J. Mitchell. *JPEG Still Image Data Compression Standard.* Van Nostrand Reinhold, 1994.

[48] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes in C.* Cambridge University Press, second edition, 1992.

[49] K. Ramchandran and M. Vetterli. Best wavelet packet bases in a rate-distortion sense. *IEEE Trans. on Image Proc.*, 2:160–175, Apr. 1993.

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

[51] A. Said and W. A. Pearlman. A new, fast, and efficient image codec based on set partitioning of hierarchical trees. *IEEE Trans. on Circ. Syst. for Video Tech.*, 6(3):243–250, Jun. 1996.

[52] J. M. Shapiro. Embedded image coding using zerotrees of wavelet coefficients. *IEEE Trans. on Signal Proc.*, 41:3445–3462, Dec. 1993.

[53] Y. Shoham and A. Gersho. Efficient bit allocation for an arbitrary set of quantizers. *IEEE Trans. Acoustics, Speech, and Signal Processing*, 36:1445–1453, Sep. 1988.

[54] A. S. Spanias. Speech coding: A tutorial review. *Proc. of the IEEE*, 82(10):1541–1582, Oct. 1994.

[55] G. S. Sullivan. Efficient scalar quantization of exponential and Laplacian random variables. *IEEE Trans. Inform. Theory*, 42(5):1365–1374, Sep. 1996.

[56] N. Tanabe and N. Farvardin. Subband image coding using entropy-coded quantization over noisy channels. *IEEE J. Selected Areas Commun.*, 10(5):926–943, Jun. 1992.

[57] D. Taubman and A. Zakhor. Multirate 3-D subband coding of video. *IEEE Trans. on Image Proc.*, pages 572–588, Sep. 1994.

[58] G. Ungerboeck. Channel coding with multilevel/phase signals. *IEEE Trans. Inform. Theory*, IT-28:55–67, Jan. 1982.

[59] M. Vetterli and K. M. Uz. Multiresolution coding techniques for digital video: A review. *Special Issue on Multidimensional Processing of Video Signals, Multidimensional Systems and Signal Processing*, 3:161–187, 1992.

[60] J. D. Villasenor, B. Belzer, and J. Liao. Wavelet filter evaluation for image compression. *IEEE Trans. on Image Proc.*, 4:1053–1060, Aug. 1995.

[61] M. J. Weinberger, J. J. Rissanen, and R. B. Arps. Applications of universal context modeling to lossless compression of gray-scale images. *IEEE Trans. on Image Proc.*, 5(4):575–586, Apr. 1996.

[62] P. H. Westerlink, J. Biemond, and D. E. Boekee. Subband coding of color images. In J. W. Woods, editor, *Subband Image Coding*. Kluwer Academic, Norwell, MA, 1991.

[63] I. H. Witten, R. M. Neal, and J. G. Cleary. Arithmetic coding for data compression. *Commun. ACM*, 30:520–540, Jun. 1987.

[64] J. W. Woods and S. D. O'Neil. Subband coding of images. *IEEE Trans. on ASSP*, 34:1278–1288, Oct. 1986.

[65] X. Wu. High-order context modeling and embedded conditional entropy coding of wavelet coefficients for image compression. In *the 31st Asilomar Conf. on Signals, Sys., and Computers*, Pacific Grove, CA, Nov. 1997.

[66] X. Wu. Lossless compression of continuous-tone images via context selection, quantization, and modeling. *IEEE Trans. Image Proc.*, 6(5):656–664, May 1997.

[67] X. Wu and J.-h. Chen. Context-modeling and entropy coding of wavelet coefficients for image compression. In *Proc. of ICASSP'97*, pages 3097–3100, Munich, Germany, Apr. 1997.

[68] X. Wu and N. Memon. Context-based, adaptive, lossless image codec. *IEEE Trans. on Commun.*, 45(4):437–444, Apr. 1997.

[69] Z. Xiong, K. Ramchandran, and M. Orchard. Joint optimization of scalar and tree-structured quantization of wavelet image decomposition. In *the 27th Asilomar Conf. on Signals, Sys., and Computers*, pages 891–895, Pacific Grove, CA, Nov. 1993.

[70] Z. Xiong, K. Ramchandran, and M. Orchard. Space-frequency quantization for wavelet image coding. *IEEE Trans. on Image Proc.*, 6:677–693, May 1997.

[71] Y. Yoo and A. Ortega. Adaptive quantization without side information using SVQ and TCQ. In *the 29th Asilomar Conf. on Signals, Sys., and Computers*, Pacific Grove, CA, Oct.–Nov. 1995.

[72] Y. Yoo and A. Ortega. Constrained bit allocation for error resilient JPEG coding. In *the 31st Asilomar Conf. on Signals, Sys., and Computers*, Pacific Grove, CA, Nov. 1997.

[73] Y. Yoo and A. Ortega. Image domain compression of simple images. In *the 32nd Asilomar Conf. on Signals, Sys., and Computers*, Pacific Grove, CA, Nov. 1998.

[74] Y. Yoo, A. Ortega, and K. Ramchandran. A novel hybrid technique for discrete rate-distortion optimization with applications to fast codebook search for SVQ. In *Proc. of ICASSP 96*, volume 4, pages 2042–2045, Atlanta, GA, May 1996.

[75] Y. Yoo, A. Ortega, and B. Yu. Adaptive quantization of image subbands with efficient overhead rate selection. In *Proc. of ICIP'96*, Lausanne, Switzerland, Sep. 1996.

[76] Y. Yoo, A. Ortega, and B. Yu. Image subband coding using progressive classification and adaptive quantization. *IEEE Trans. on Image Proc.*, Jun. 1997. Submitted.

[77] B. Yu. A statistical analysis of adaptive quantization based on causal past. In *IEEE Int'l Symp. on Info. Th., ISIT'95*, 1995.

[78] B. Yu. A statistical analysis of adaptive scalar quantization based on quantized past. *IEEE Trans. Inform. Theory*, 1996. Submitted.

[79] D. Yu and M. W. Marcellin. A fixed-rate quantizer using block-based entropy-constrained quantization and run-length coding. *IEEE Trans. Inform. Theory*, Dec. 1997. Submitted, Available in `http://www-spacl.ece.arizona.edu/`.

[80] G. Yu, M. W. Marcellin, and M. M.-K. Liu. POCS based error concealment for packet video using multi-frame overlap information. *IEEE Trans. on Circ. and Syst. for Video Technology*, Aug. 1996. Submitted, Available in `http://www-spacl.ece.arizona.edu/`.

[81] A. Zandi, M. Boliek, E. L. Schwartz, M. J. Gormish, and J. D. Allen. CREW lossless/lossy image compression – Contribution to ISO/IEC JTC 1.29.12. ISO/IEC JTC1/SC29/WG1 N196, Jun. 1995.

[82] A. Zandi, M. Boliek, E. L. Schwartz, and A. Keith. CREW second evaluation - ISO/IEC JTC1/SC29/WG1. ISO/IEC JTC1/SC29/WG1 N252, Nov. 1995.