Subband-Vector Quantization Coding of Color Images with Perceptually Optimal Bit Allocation

Robert E. Van Dyck

Center for Communications and Signal Processing
Department of Electrical and Computer Engineering
North Carolina State University

TR-92/12
September 1992

Abstract

VAN DYCK, ROBERT ERNEST. Subband-Vector Quantization of Color Images with Perceptually Optimal Bit Allocation (Under the direction of Professor Sarah A. Rajala)

The combination of subband coding and vector quantization can provide a powerful method for compressing color images. The use of properties of the human visual system can increase the performance of such a system and allow one to achieve very high quality reconstructed images at compression ratios exceeding 10:1. In this dissertation, we design color subband-vector quantization systems, and for each system formulate the bit allocation problem as an optimization problem. The objective function depends on the distortion-rate curves of the quantizers and on a set of perceptual weights. These weights are derived from data provided by experimental measurements [45, 71] of the mean detection threshold of the human visual system for color transitions along the luminance, red-green, and blue-yellow directions. Minimization of the objective function constrained by the desired bit rate gives a perceptually optimal bit allocation.

Two different subband configurations are designed. The rectangular configuration filters each input image into seven subbands using separable quadrature mirror filters; these subbands are rectangular regions in the two-dimensional Fourier plane. The human visual system is more sensitive to patterns oriented horizontally or vertically,

instead of diagonally. To take advantage of this fact, a new diamond configuration is proposed and analyzed. By first diagonally interpolating the input image and then filtering with separable one-dimensional filters, a decomposition into subbands that more closely match the orientation of the human visual system can be obtained [86]. For compression ratios on the order of 10:1, a five-band system with diamond-shaped subbands yields results comparable to the seven-band rectangular system.

Three subband/VQ cases are examined for each configuration. In the first two cases (Case 1 and Case 2), the color components of the lowest frequency subband are scalar quantized. Case 1 combines the three color components of each pixel of the higher frequency subbands into a three-dimensional vector, while Case 2 creates four-dimensional vectors from 2×2 blocks in each subband color component. The third case (Case 3) is the same as Case 2, except that the chrominance components of the lowest frequency subband are also vector quantized with 2×2 blocks in each component.

To obtain the required compression ratio and the high color fidelity required for HDTV applications, the vector quantization is done in two different perceptually uniform color spaces, C.I.E. $L^*a^*b^*$ space and AC_1C_2 space. Results from these color spaces are compared to results obtained in N.T.S.C. YIQ space. While the uniform color spaces provide better color reproduction, they are also subseptable to color artifacts in localized areas. The trade-offs between using the various color spaces are examined for a number of different compression ratios.

SUBBAND-VECTOR QUANTIZATION CODING OF COLOR IMAGES WITH PERCEPTUALLY OPTIMAL BIT ALLOCATION

by

ROBERT E. VAN DYCK

A dissertation submitted to the Graduate Faculty of
North Carolina State University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

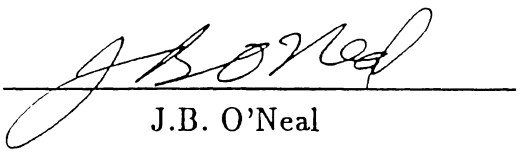
Department of Electrical and Computer Engineering

Raleigh

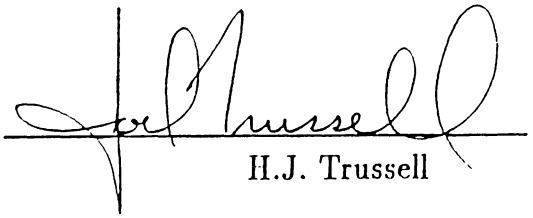
August 1992

Approved By:

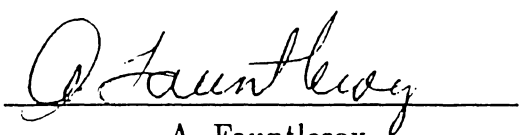
S.A. Rajala,
Chairman of Advisory Committee



J.B. O'Neal



H.J. Trussell



A. Fauntleroy

To my parents

Robert L. and Edythe A. Van Dyck

whose love and support made this possible.

Acknowledgements

I would like to give special thanks to Professor Sarah A. Rajala, who not only taught me image processing and acted as my adviser, but also suggested the areas of subband coding and vector quantization as research topics.

Thanks must also be given to Professor H. Joel Trussell, for his review of my dissertation, and for his lectures on color image processing.

Further thanks must also be given to the other members of my graduate committee, Professors J. B. O'Neal and A. Fauntleroy.

The research in this dissertation was supported by the Center for Communications and Signal Processing at North Carolina State University. The funding for the work was primarily supplied by the Eastman Kodak Co. My thanks are given to both organizations for their support.

The discussions with Mr. James Sullivan and Mr. Larry Iwan of Kodak about color image coding were greatly appreciated. Mr. Sullivan also deserves thanks for suggesting the basic idea of the diamond subband decomposition.

Finally, I would like to thank the many present and former graduate students of North Carolina State University for all of the interesting discussions about the various aspects of image processing. In particular, Mr. Michael Vrhel must be mentioned for his discussions about color science and mathematics, and Dr. Kai-Kuang Ma for his discussions about subband coding.

Contents

List of Tables

List of Figures

CHAPTER 1

INTRODUCTION

Much of the work in video compression has been aimed at reducing the bandwidth of gray-scale video sequences by a factor of thirty or more so that they can be sent through existing digital networks. Early approaches used DPCM encoding and entropy coding [63, 64] to achieve good quality images but at lower compression ratios. For teleconferencing and other applications, the initial video is of near broadcast television quality while the received video is usually degraded. This is particularly true for rapid changes in the scene. Newer methods, including some that use artificial intelligence techniques or that incorporate image segmentation algorithms [82], can achieve even higher compression ratios, but at the expense of a higher computational burden.

For the digital transmission of high definition television signals, one must contend with an extremely high bit rate. The standards proposed for HDTV video require approximately 1.2 Gbits/sec, and the intended digital channel has a capacity of 135 Mbits/sec. While it is necessary to reduce the bit rate by roughly 9:1, the large amount of raw data limits the operations that can be performed in real time. Among the possible solutions, a multiresolution signal decomposition method based on subband coding looks to be especially promising. This method filters an image into a number of subimages of different scales. At these different resolutions, the contents of the various subimages contain information about different sized structures in the

image [55].

There are four sections in the rest of this chapter. The first one gives a short history of subband coding, particularly its use in image compression. The second section discusses the objectives of this research, and the third lists the contributions made by this work. The last section provides an outline of the rest of the dissertation.

1.1 Subband Coding

Subband coding is a method in which a signal is filtered into a number of different frequency bands. These subbands are then separately coded and transmitted. At the receiver, the subbands are decoded and then added together to yield an estimate of the original signal. Crochiere *et al.* [13], presented this method for compressing speech signals, where the coding was done with adaptive-PCM. The advantage of subband coding is that one can shape the noise spectrum by allocating a different number of bits to each subband based on its perceptual importance. For a given bit rate, this allows one to adjust the bit rate of each subband so that the subjective effect of the quantization noise is minimized.

In the digital implementation of subband coding, the signal is filtered with digital bandpass filters. These filters slightly overlap each other so that no part of the spectrum is lost. To ensure that the total number of samples remains the same, the subbands are decimated before coding. This decimation process introduces aliasing into the system. After coding, transmission, and decoding, the subbands are interpolated back up to the original sampling frequency. This process causes replication of the frequency spectrum of the signal entering the interpolator. These replications are called images, and they can be removed by another application of bandpass filtering.

The subbands are then added together to create the reconstructed signal.

Esteban and Galand [17] proposed a technique using quadrature mirror filters, QMFs, that results in the elimination of the aliasing for a two band decomposition. Johnston [41] then designed a number of quadrature mirror filters using a numerical optimization algorithm. Subband systems with more than two subbands can be implemented by using a tree-structured decomposition where each level of the tree splits a subband into two subbands with half the bandwidth. Quadrature mirror filters eliminate the aliasing, but can still cause amplitude and phase distortion. Smith and Barnwell [85] showed that it is possible to design a tree-structured subband system that results in perfect reconstruction of the input signal.

The application of subband coding to image compression was done by Woods and O'Neil [104]. They filtered a monochrome image into subimages and then coded these subimages with adaptive-PCM. The required two-dimensional filtering was done separately along the horizontal and vertical directions using one-dimensional QMFs. Gharavi and Tabatabai [32], and Kim *et al.* [43, 44] extended this to color image compression. In both cases, the color image was represented in YIQ space, and most of the bits were used to code the luminance, Y, component.

The use of vector quantization has become a popular method to encode the subbands. Westerink *et al.* [102] filtered an image into 16 equal rate subbands. They created sixteen-dimensional vectors by taking pixels from the same location in each subband. Antonini *et al.* [2, 22] used a wavelet transform to obtain a set of subimages at various resolutions and for three preferential directions. Two-dimensional vectors were then used to code these subimages. Furukawa *et al.* [27] coded super high definition images using two stage vector quantization. Depending on the bit allocation for a particular subband, either no quantization, scalar quantization, or vector quan-

tization was used to encode the subband. The vectors were created from either 2×2 or 4×4 blocks in each subband. Furlan *et al.* [26] used adaptive vector quantization followed by arithmetic coding to achieve excellent quality results for monochrome images coded at rates between 0.5 and 1.0 bits/pixel.

Although this work will deal only with single-frame image compression, subband coding with vector quantization has also been applied to video coding. Fadzil and Dennis [19] coded color video that was represented in YUV space. The luminance component had a resolution of 360×288 pixels and the two chrominance components each had a resolution of 180×144 pixels. After motion compensated interframe prediction, the error image was decomposed into subbands. Ten luminance subbands and six chrominance subbands were used to create a sixteen-dimensional vector. The other six luminance and two chrominance subbands were discarded. Akansu and Kadur [1] also used a subband structure to decompose the motion compensated frame difference in a monochrome video system. Podilchuk *et al.* [70] presented a three-dimensional subband coder for video where the non-dominant subbands were vector quantized, and Irie and Kishimoto [39] designed an HDTV system where the subbands were coded with adaptive DCT and DPCM.

An error analysis of subband coding with scalar quantization was done by Westerink, Biemond, and Boekee [103]. Although quadrature mirror filters cancel aliasing, the introduction of coding errors can lead to aliasing errors in a practical subband system. They used a model for the quantizer to determine the size of the aliasing error, and then they compared this error with other coding errors. Aliasing in subband systems will be defined and discussed in detail in Chapter 3. Fischer [23] addressed the question of whether subband coding is optimal in the rate-distortion sense when compared to fullband coding. He showed that for wide sense stationary Gaussian

processes, realizable filters, and two subbands, subband coding is usually suboptimal.

The original use of subband coding for speech processing used perceptual information to achieve good quality reconstruction. The same can be done for image coding. It is the use of perceptual information that offsets the suboptimality of subband coding. A number of researchers have used human visual system information to design better subband coders. Safranek and Johnston [77] used an empirically derived perceptual masking model to set the quantization levels in a DPCM quantizer. Perkins and Lookabaugh [69] used contrast sensitivity data to compute a set of weights for use with the subbands' variances in a marginal analysis bit allocation algorithm. Johnsen *et al.* [40] derived an activity index from the baseband subimage and used this to select particular quantizers for the upper subimages.

For coding color images, various properties of the human visual system, such as the relative insensitivity to slight color errors in areas of high frequency luminance variations, can be exploited in the reduction of the bit rate. Barba and Hanen [5] used a human visual model to design vector quantizers for the chrominance components of a color subband system. For HDTV sequences, subband coding will yield a signal to noise ratio over 40 dB for color images encoded at approximately 3 bits/pixel [62].

1.2 Objectives

The goal of this research is two-fold. First, by using principles from color science and properties of the human visual system, a better understanding is sought of the perception of color errors in image compression systems. This was first examined by performing simple quantization experiments in a number of different color spaces and comparing calculated distortion numbers with a subjective assessment of image

quality. The use of subband coding with scalar quantization was then considered so that higher compression ratios could be achieved.

The subjective visual quality of compressed color images can often be improved by transforming to and then processing in a perceptually uniform color space. Test images stored as C.I.E. XYZ tristimulus values are transformed to N.T.S.C. YIQ space, C.I.E. $L^*a^*b^*$ space, and AC_1C_2 space [20]. All three of these color spaces consist of a luminance component and two chrominance components where the chrominance components are essentially red-green and blue-yellow. The latter two spaces are considered to be perceptually uniform color spaces since the Euclidean distance between two colors in these spaces is almost proportional to the perceived difference between the colors.

While the use of a uniform color space can lead to a better prediction of color errors, it was not clear when such color spaces would be practical for use in image compression. Because these color spaces involve nonlinear transformations of the input color space, the inverse transformation may map quantization errors into visually objectionable color errors. One of the contributions of this research is the evaluation of coding systems in uniform color spaces. The comparison with systems that process in YIQ space is important to determine if the improvement in overall color reproduction offsets the possibility of unacceptable color artifacts.

The second objective of this research is to design an actual subband coding system that is capable of compressing high resolution color images by a ratio of at least 10:1 while providing almost visually lossless reconstructed images. To achieve the desired compression ratios, the subbands were vector quantized, and a perceptually optimal bit allocation algorithm was derived. Two different cases are examined for creating the vectors. Case 1 creates three-dimensional vectors by taking the three color com-

ponents of each pixel as a vector. This uses the fact that the color components are usually highly correlated. Case 2 forms a vector from a block of spatially adjacent pixels in a subband as is done in monochrome vector quantization. The different color components can be coded at different resolutions by subsampling. This is effective in color spaces such as YIQ where the chrominance components can tolerate a reduction in bandwidth.

1.3 Contributions

The major contributions of this dissertation are:

- The derivation of an iterative approximation to the optimal scalar quantizer, and the use of this approximation in designing quantizers in various color spaces.

- An investigation of image compression in various color spaces including perceptually uniform ones. Images were quantized in XYZ, RGB, YIQ, $L^*a^*b^*$, and AC_1C_2 space, and the distortion was computed and compared with a subjective evaluation of the visual degradations.

- The design of a general subband system for color image compression including the use of an internal color space, XYZ space, and the transformation for the specific output device as the final step in the processing.

- The design, analysis, and simulation of a new diamond subband configuration using separable quadrature mirror filters, and the comparison of this configuration with a standard rectangular one.

- The formulation of the bit allocation process for a color subband coding system as an optimization problem, and the solution of this problem yielding the perceptually optimal bit allocation. Perceptual weights were derived based on subjective experi-

ments that determined the mean detection threshold of the human visual system.

- The design of a practical subband/vector quantization coding system that achieves very high quality reconstructed images at low bit rates.

1.4 The Outline of the Dissertation

This dissertation is arranged in the following way. The second chapter discusses color science since it is important for understanding this work. A vector space representation for color imagery is presented, and the mathematical basis for some subjective color phenomena is shown. The generalization of image compression from monochrome to color images is discussed, and a general system for color image compression is presented. Next, the choice of a distortion measure to be used in this work is given. The chapter contains a discussion about the display of color images, and concludes by providing the transformations used in the rest of the work.

The third chapter discusses subband coding including how one can eliminate aliasing and achieve perfect reconstruction. The extension to two dimensions is given, and a new system configuration using diamond-shaped subbands is presented. This configuration uses separable quadrature mirror filters to filter the image into subbands that more closely match the spatial frequency response of the human visual system. The separability of the filters is important since the design of two-dimensional nonseparable filters is not trivial, and a separable implementation can usually be made computationally more efficient. The distortion caused by the subband filterbanks is calculated for both the rectangular and diamond configurations in the various color spaces.

In the fourth chapter, an iterative algorithm is derived that allows one to design

one-dimensional minimum mean squared error quantizers using a discrete histogram as input. The resulting quantizer is a close approximation to the optimal one for a large enough number of quantization levels. This algorithm is used to design the quantizers for each component of a color image. The minimum mean squared error criterion is, therefore, implemented in the color space component that is being quantized. Simulations are run using these quantizers at a nominal compression ratio of 2:1 for different bit allocations among the three color components. The results for a number of different color spaces are compared to determine which color spaces are better for compressing images. This work shows that the choice of color space can make a significant difference in the performance of the coder, particularly at low bit rates. The use of a perceptually uniform color space can lead to better color fidelity in the reproduction.

One of the assumptions made in deriving the approximation to the optimal quantizer is that the number of quantizer levels is large. While this is basically true for the above simulations that used low compression ratios, this condition would be violated for higher compression ratios. A one-dimensional version of the LBG algorithm is used to design scalar quantizers for all of the components of the subbands in the various color spaces. The bit allocation for the subband coding of color images using scalar quantization is formulated as an optimization problem and solved in this chapter by using a conventional Lagrange multiplier method. Since the results are not constrained to be non-negative integers, the results are rounded to the nearest integer and adjusted to achieve the desired bit rate. More simulation results are presented that show the effects of subband/SQ in the different color spaces.

The fifth chapter provides background on vector quantization. It describes what vector quantization is, and how to design a vector quantizer using the LBG algorithm.

To reduce the computational complexity of the encoding and decoding operations, Tree-Searched vector quantization can be used. This method is explained, and a few other types of vector quantizers are also discussed. The choice of training sequences and the actual design of the codebooks used in the computer simulations are also considered.

The creation of a vector from the three color components (Case 1) has been used to design color look-up tables [11], but this work is the first application, to the author's knowledge, of this method to code the color subbands created by a subband coder. This is compared with a straight-forward extension of monochrome vector quantization (Case 2) where each vector is created by taking 2×2 blocks in each color component. Because of the large number of levels required for coding the lowest frequency subband, this subband was scalar quantized in both cases. The first system results in a simpler bit allocation problem. The simulation results show that the performance of these systems are comparable for compression ratios on the order of 10:1. At higher compression ratios, the extra degrees of freedom in Case 2 lead to superior performance. In order to achieve better results at even higher compression ratios, another case (Case 3) is simulated. Here, the chrominance components of the lowest frequency subband are also vector quantized.

A major contribution is derived in the sixth chapter. Here, the bit allocation problem is formulated as an optimization problem where the objective function is weighted by the response of the human visual system. A version of the Marginal Analysis algorithm is presented as a solution to this problem, and sufficient conditions for this to be an optimal solution are stated. A heuristic method is then given to handle the case where the bit allocation, resulting from the Marginal Analysis algorithm, is not necessarily optimal. This heuristic method, combined with the use

of the perceptual weights, still provides high quality bit allocations.

The objective function uses perceptual weights derived from experimental measurements of the mean detection threshold of the human visual system done by Krishnakumar [46]. The experiments measured the mean detection threshold as a function of spatial frequency, spatial orientation, background luminance, background color, and direction of color transition. The transformation of this data to the desired color spaces is explained, and an example is given. The perceptual weights are then derived for each subband color component for both the rectangular and diamond subband configurations.

The seventh chapter gives simulation results for the rectangular configuration of the subband/VQ system. The effects of the choice of training sequence, the color space, and the vector quantizer are discussed. Reconstructed images of very good visual quality are obtained at compression ratios of 8:1, 12:1, and 16:1, and good quality images are obtained at 20:1 and 24:1. The use of the perceptual weights leads to visually obvious improvements over the use of uniform weights, especially at the higher compression ratios.

Simulation results for the diamond subband configuration are given in the eighth chapter. For an 8:1 compression ratio, the results are very good and comparable to the rectangular configuration. At higher compression ratios, the results are not as robust. This is due to the fewer degrees of freedom available to the bit allocation algorithm, and to the larger size of the lowest frequency subband. The final chapter provides a summary of the work, draws conclusions, and points the way to further research.

CHAPTER 2

COLOR SCIENCE

2.1 Color Spaces

2.1.1 Color Matching

With the exception of a few colors of high saturation, all colors can be matched by a combination of three different color primaries [38]. By matched, it is meant that the additive combination of particular amounts of a red, a green, and a blue light appears the same to the viewer as a given color. The reason for this phenomenon is that the human eye has three different color sensitive receptors. The receptor responses are functions of the wavelength of the incident light and can be denoted by β_λ, γ_λ, and ρ_λ. The functions have maxima in the blue, green, and red regions of the spectrum, respectively.

In the color matching experiment, the subject is shown the color to be matched on one side of a split screen, and the combination of the three primary colors on the other. By adjusting the intensities of the primaries, a color match is attempted. Usually, the color to be matched has its intensity normalized to unity in whatever units are being used. These units can be radiometric ones such as watts, or photometric ones such as lumens. This color matching can be expressed symbolically as follows:

$$1(C) = R_c(R) + G_c(G) + B_c(B). \tag{2.1}$$

This equation states that one unit of color C is matched by R_c units of color R, G_c

units of color G, and B_c units of color B.

The color, C, can be a pure spectral color containing only a single wavelength of light (monochromatic), or it can be a combination of many different wavelengths. The colors R, G, and B are the primaries, and they are usually monochromatic colors. The coefficients of these primaries, R_c, G_c, and B_c, are known as the tristimulus values of color C. If the tristimulus values, R_c, G_c, and B_c are allowed to have negative values, then all colors can be matched. By performing a color match for all monochromatic colors, three tristimulus curves can be created. The R tristimulus curve contains the R tristimulus values for all of the monochromatic colors of the spectrum; the G and B tristimulus curves are defined similarly.

2.1.2 Vector Space Representation of Color Imagery

By sampling a continuous spectrum at a sufficient number of points, a vector space formulation of the color matching problem can be obtained [91]. Denote the sampled spectrum by the n-dimensional vector, $\mathbf{f}$. Since the human visual system (HVS) has three color sensors, it can be represented by a matrix, $\mathbf{S} = [\mathbf{s}_1\ \mathbf{s}_2\ \mathbf{s}_3]$, containing a set of three n-dimensional vectors. The eye's response to the spectrum $\mathbf{f}$ is then given by

$$\mathbf{c} = \mathbf{S}^T\mathbf{f} \tag{2.2}$$

where $\mathbf{c}$ is a three-dimensional vector.

The color matching experiment discussed in the previous section can now be formulated as follows. Choose three linearly independent n-dimensional primaries, $\mathbf{p}_1$, $\mathbf{p}_2$, and $\mathbf{p}_3$, and use them to create an $n \times 3$ matrix given by $\mathbf{P} = [\mathbf{p}_1\ \mathbf{p}_2\ \mathbf{p}_3]$. Denote the monochromatic colors by $\mathbf{e}_i$, $i = 1, \cdots, n$, where $\mathbf{e}_i$ has a one in the i^{th} component and zeros in all others. Again, an observer is asked to adjust the amplitudes of the

primaries until the combination matches the test color. This can be written as

$$\mathbf{S}^T\mathbf{e}_i = \mathbf{S}^T\mathbf{P}\mathbf{m}_i \tag{2.3}$$

where $\mathbf{m}_i$ is the three-dimensional vector of the gains of the primaries.

The results from matching all spectral colors can be combined into the equation

$$\mathbf{S}^T\mathbf{I} = \mathbf{S}^T\mathbf{P}\mathbf{M}^T \tag{2.4}$$

where $\mathbf{I}$ is the $n \times n$ identity matrix. The $n \times 3$ matrix $\mathbf{M}$ is known as the color matching matrix. Since both $\mathbf{S}$ and $\mathbf{P}$ are rank three matrices, Eq. (2.4) can be solved for $\mathbf{M}$ giving

$$\mathbf{M} = \mathbf{S}(\mathbf{P}^T\mathbf{S})^{-1}. \tag{2.5}$$

Using the color matching matrix, $\mathbf{M}$, one can calculate the tristimulus values, $\mathbf{t_P}$, of an arbitrary spectrum according to

$$\mathbf{t_P} = \mathbf{M}^T\mathbf{f}. \tag{2.6}$$

The n-dimensional spectrum is being projected onto a 3-dimensional subspace that defines a particular color space. Using Eq. (2.5), this subspace can be related to the human visual (sub)space. Because of this reduction in dimension, two different spectra can appear the same to a human observer. These spectra are metamers and the relation is defined mathematically as

$$\mathbf{M}^T\mathbf{f}_1 = \mathbf{M}^T\mathbf{f}_2. \tag{2.7}$$

One can derive the transformation between two color matching matrices $\mathbf{M}$ and $\mathbf{N}$ having different primaries. $\mathbf{Q}\mathbf{N}^T = \mathbf{I}$, where $\mathbf{Q} = [\mathbf{q}_1\ \mathbf{q}_2\ \mathbf{q}_3]$ is the matrix of primaries

of N, and I is the $n \times n$ identity matrix. Premultiplying both sides by $\mathbf{M}^T$ yields $\mathbf{M}^T\mathbf{Q}\mathbf{N}^T = \mathbf{M}^T$, or

$$\mathbf{N}^T = (\mathbf{M}^T\mathbf{Q})^{-1}\mathbf{M}^T. \tag{2.8}$$

This transformation also defines the mapping between the tristimulus values in the two different color spaces:

$$\mathbf{t_Q} = (\mathbf{M}^T\mathbf{Q})^{-1}\mathbf{t_P}. \tag{2.9}$$

2.1.3 C.I.E. XYZ Space

One standard set of primaries is the C.I.E. red, green, and blue primaries consisting of single wavelength light at 700, 546.1, and 435.8 nm. In the 1920's, a new color space was desired that would satisfy a number of criteria. The main criterion was that the color matching functions for this space be always positive. This implies that the primaries are non-realizable. Also, one of the color matching functions should be the luminous efficiency function. Furthermore, this color space should also be a linear transformation of the color space containing red, green, and blue primaries.

The resulting color space that met the above criteria is the 1931 C.I.E. XYZ space, in which the Y component contains the luminance of the color. A complete derivation of this space can be found in [87]. The three C.I.E. XYZ color matching functions were defined. These functions are discrete curves that determine the contribution of each monochromatic color to the three tristimulus values. Figure 2.1 shows these functions; the samples were taken every two nanometers, but are connected in this plot. Each one of these curves becomes one column of the color matching matrix M, and Eq. (2.6) can then be used compute the tristimulus values of any spectrum. The components of the three-dimensional tristimulus vector in this color space are

denoted by X, Y, and Z.

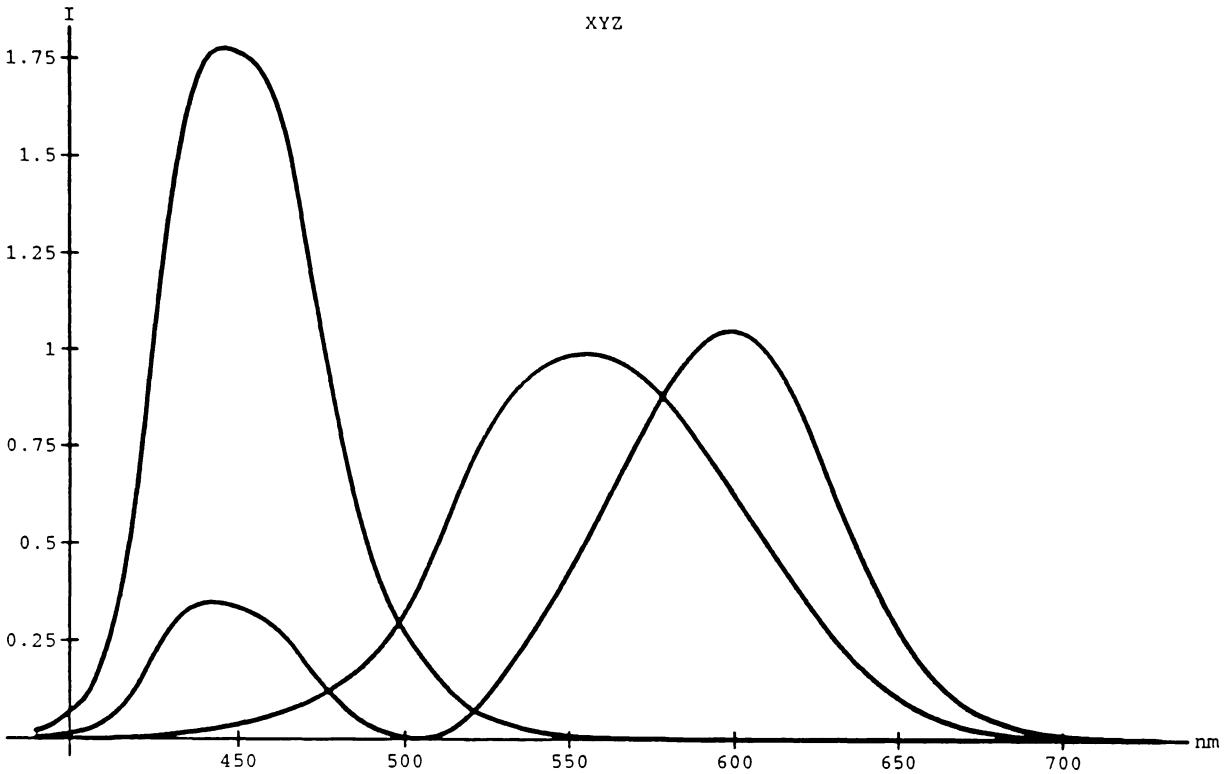


Figure 2.1: C.I.E. XYZ color matching functions.

It is often necessary to discuss only the color, or chrominance, information without regard to the luminance information. This can be done by generating chromaticity coordinates and chromaticity diagrams. The XYZ chromaticity coordinates are designated x, y, and z, and are given by:

$$x = \frac{X}{X+Y+Z} \quad y = \frac{Y}{X+Y+Z} \quad z = \frac{Z}{X+Y+Z}. \tag{2.10}$$

Since $x + y + z = 1$, it is necessary to specify only two of the three coordinates, e.g. x and y. Then, a color can be located by a point (x, y) in a two-dimensional chromaticity diagram. Chromaticity coordinates can be defined for other color spaces

in a similar way. Figure 2.2 shows a chromaticity plot of the spectral locus and the line of purples. The spectral locus is the horseshoe-shaped part of the figure, and it shows the chromaticity coordinates of the pure spectral colors whose wavelengths are given on the plot. The line of purples is the straight line connecting the ends of the horseshoe.

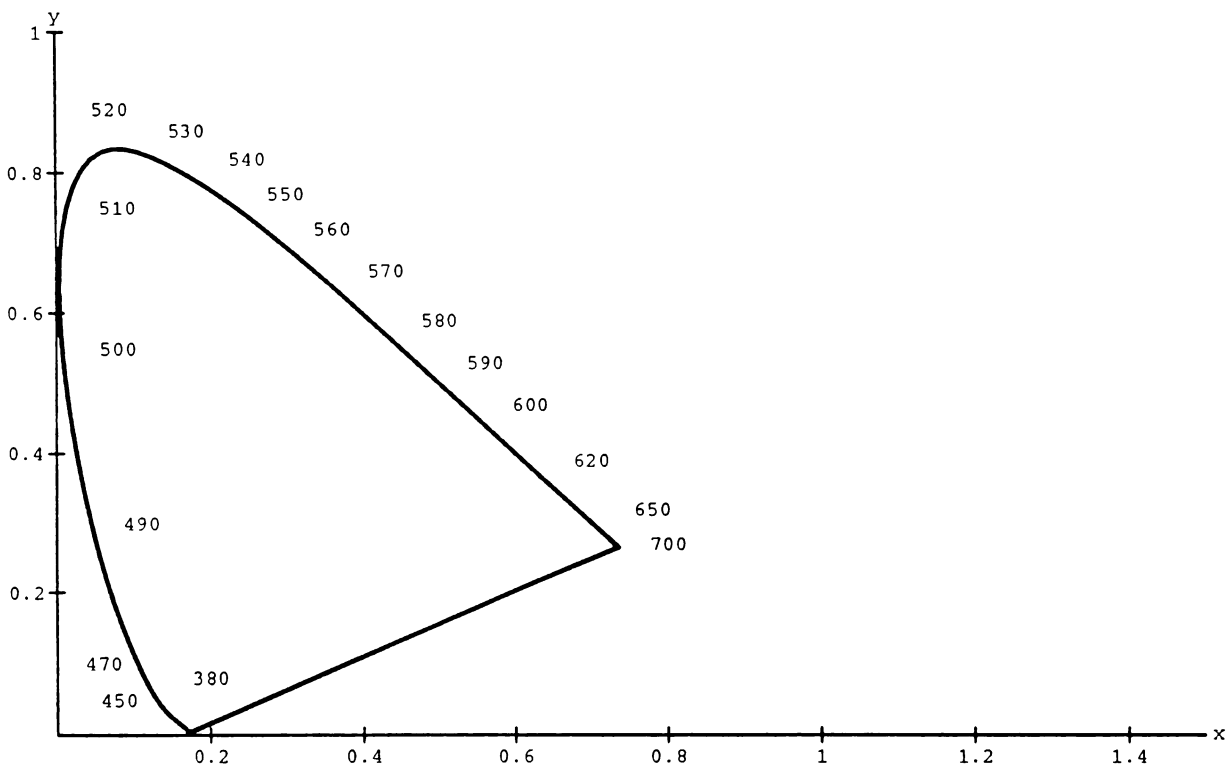


Figure 2.2: Chromaticity Plot of the Spectral Locus and the Line of Purples. The numbers on the plot refer to the wavelengths of the colors in nanometers. The line of purples is the straight line connecting 380 nm with 700 nm.

2.1.4 N.T.S.C. *RGB* Space

Starting from XYZ space, transformations to other color spaces have been defined. These color spaces include N.T.S.C. RGB, N.T.S.C. YIQ, and C.I.E. $L^*a^*b^*$ space

[87]. The first two are linear transformations of XYZ space and are versions of those used in broadcast color television. YIQ space, in particular, provides good energy compaction since the luminance component, Y, carries much of the information. The last color space results from a nonlinear transformation of XYZ space. It is considered to be a uniform color space since equal color differences correspond to almost equal visual color differences.

The typical color monitor or television set displays color images through the use of three electron guns. These guns are used to excite red, green, and blue phosphors, respectively. Although the display is in some RGB space, the processing, including coding and transmission, may be more efficiently done by first transforming this color space to another one before doing any required processing. The image is transformed back to the color space of the display just before viewing. This idea has been implemented in conventional color television where the three analog color signals are linearly transformed to a luminance and two chrominance signals, YIQ, to achieve bandwidth compression.

The National Television Systems Committee (N.T.S.C.) defined a color space by choosing the chromaticities of the three phosphors and the white point [50]. These values are given in Table 2.1 where the the white point is standard illuminant C. Standard illuminant C represents average daylight with a correlated color temperature of approximately 6774 K [105, pages 143-145]. Based on these phosphors, one can transform the XYZ color matching functions to create the N.T.S.C. color matching functions. These color matching functions are shown in Figure 2.3 where the gains of the functions were adjusted so that standard illuminant C has a luminance value of unity. The matrix transformation will be given in Section 2.4.

Color	x	y
Red	0.670	0.330
Green	0.210	0.710
Blue	0.140	0.080
White	0.310	0.316

Table 2.1: N.T.S.C. Phosphor Chromaticities

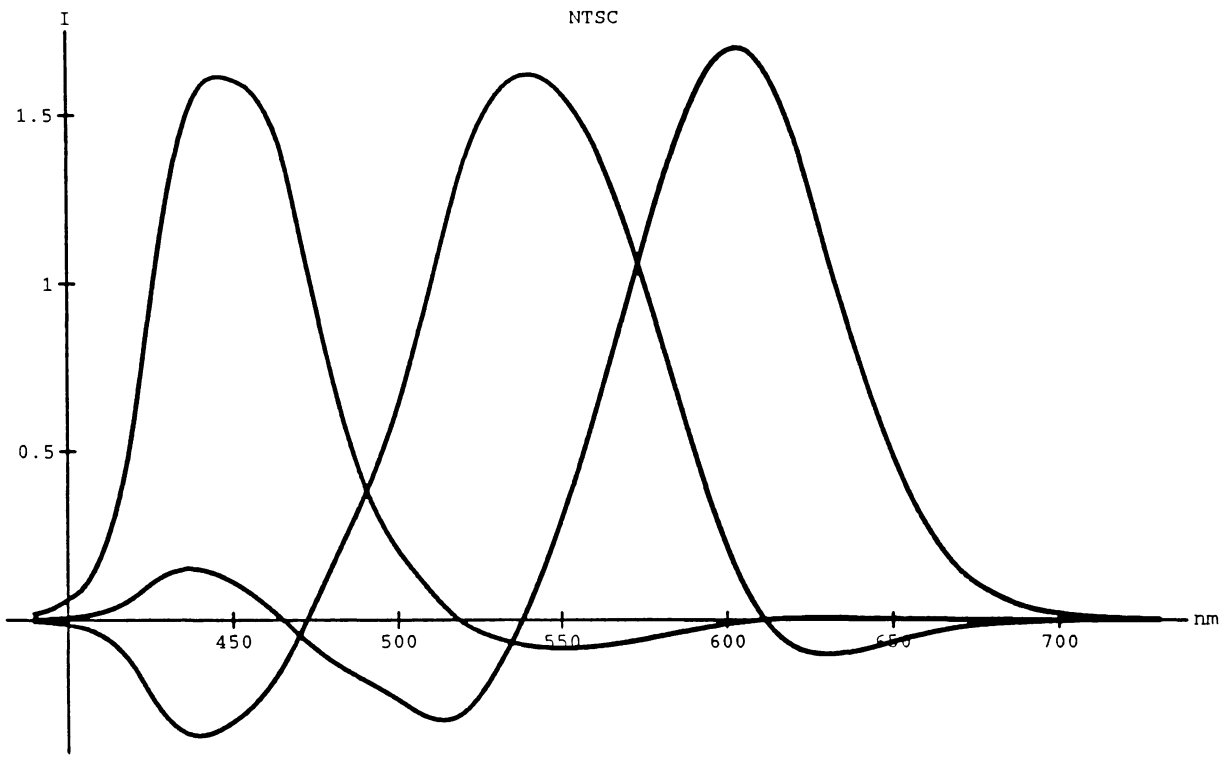


Figure 2.3: N.T.S.C. color matching functions.

2.2 Color Sensitivity

2.2.1 Color Space Processing

An underlying problem in all color image compression methods is the determination of the sensitivity of the human visual system to color errors. This is especially true in systems that are to provide almost visually lossless images such as HDTV systems.

Studies done by MacAdam [53] and others show that the human visual system is not equally responsive to the same size color error in different parts of XYZ or an RGB color space. This can be seen in Figure 2.4 which shows a chromaticity plot with MacAdam's Ellipses denoted. These ellipses represent the smallest change necessary for an observer to detect a color shift; the ellipses have been scaled by a factor of ten to make them more visible.

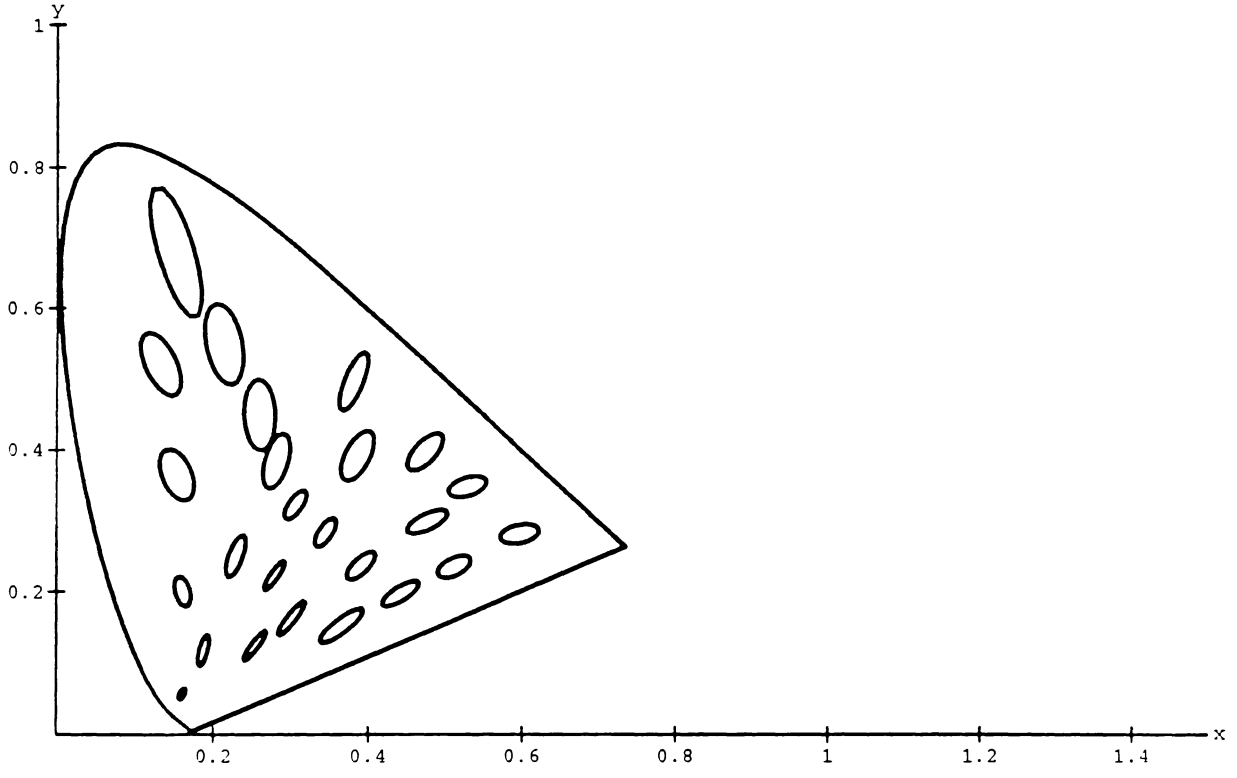


Figure 2.4: Chromaticity Plot of MacAdam's Ellipses. Spectral Locus and Line of Purples are also plotted.

One approach to this problem involves the transformation of the color space to a color space that has more perceptually uniform errors. Two such spaces are the C.I.E. $L^*a^*b^*$ space and AC_1C_2 space. Figures 2.5 and 2.6 show the MacAdam's Ellipses in these color spaces. Ideally, the ellipses are mapped to circles of constant diameter.

Even though this does not happen, the ellipses in these spaces are more nearly the same size. The ellipses in both color spaces have also been increased by a factor of ten.

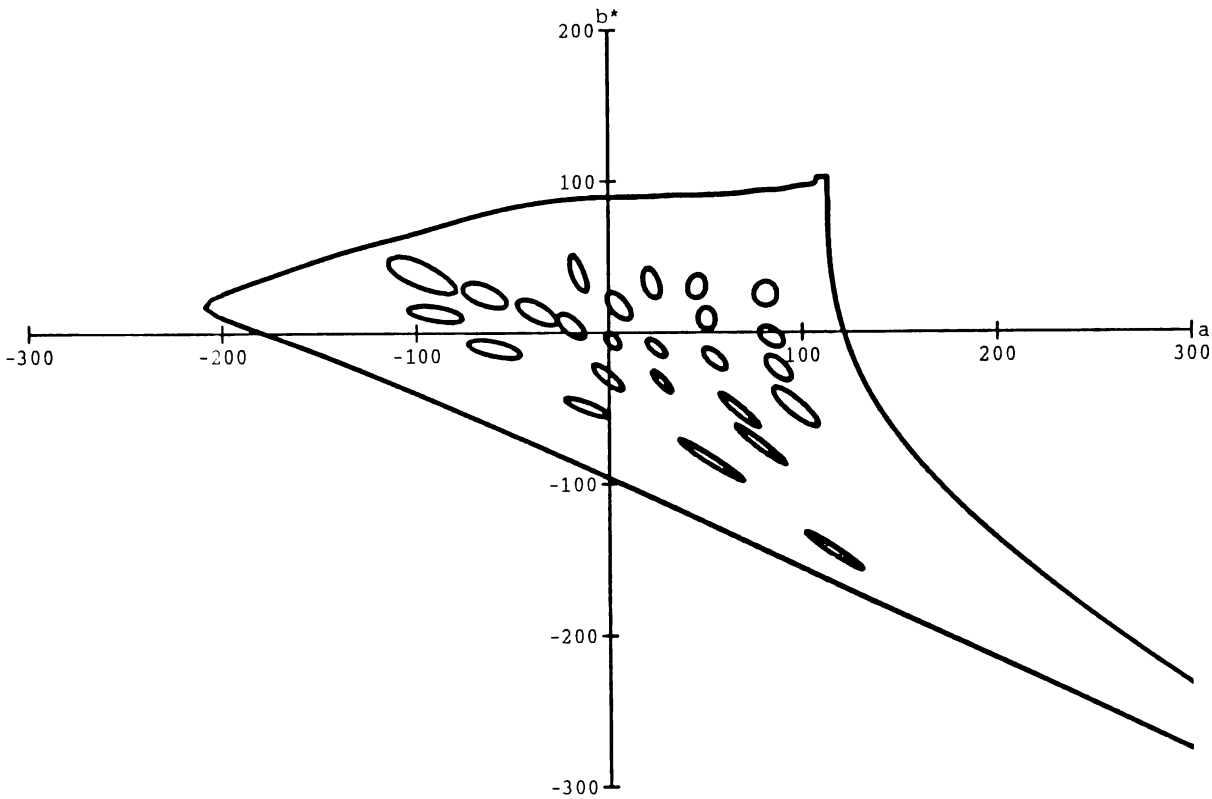


Figure 2.5: MacAdam's Ellipses in $L^*a^*b^*$ Space. $Y = 5\ cd/m^2$. Spectral Locus and Line of Purples are also plotted.

The sizes of the color errors depend on the number of levels in the quantizer as well as the type of coder used. Gentile, *et al.* [29], have performed experimental calculations of color errors for uniform quantizers with fixed step sizes for various color spaces. We have extended this line of work by designing minimum mean squared error quantizers for each component of an image in a given color space. This was done through an iterative algorithm that provides a close approximation to the optimal

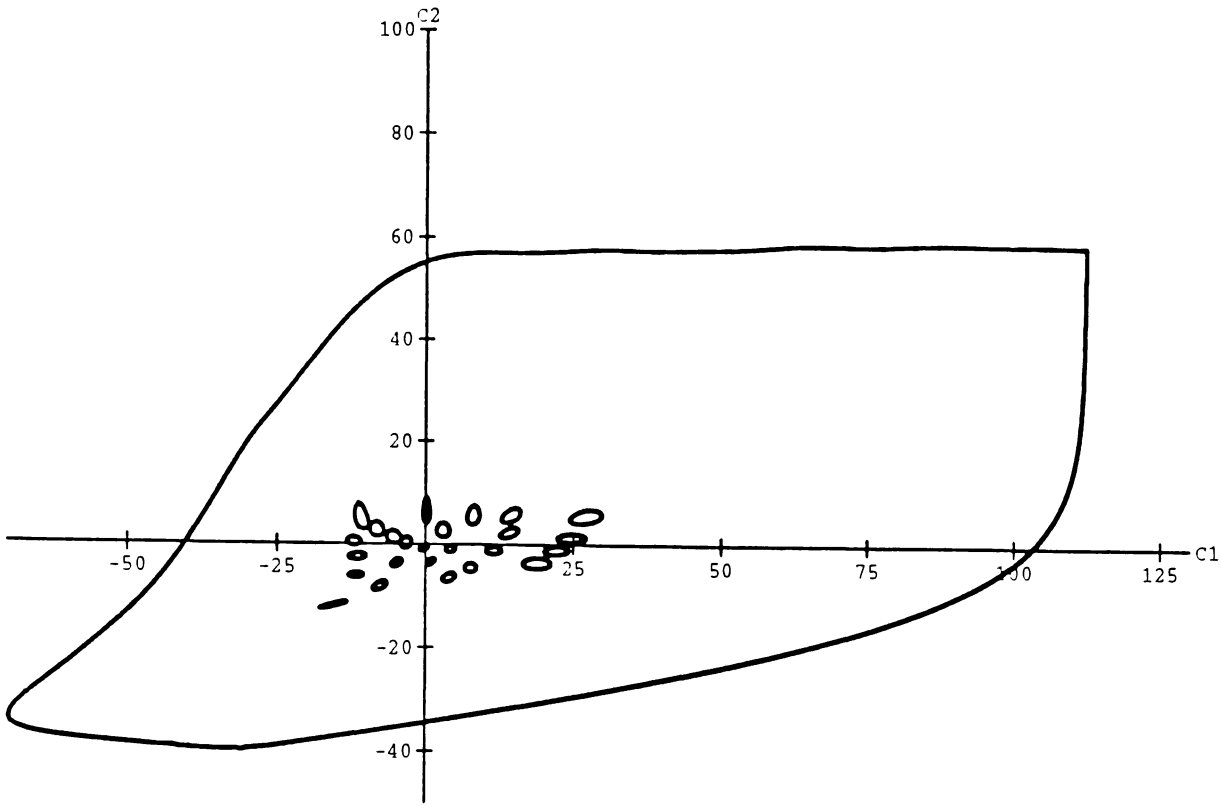


Figure 2.6: MacAdam's Ellipses in AC_1C_2 Space. $Y = 5\ cd/m^2$. Spectral Locus and Line of Purples are also plotted.

case. Both the direct quantization case and a subband coder were simulated, with the optimal quantizers designed for each component of each subband in the latter case [94].

In this work we continue to investigate the effects of color image compression in different color spaces. The input is a series of test images that are recorded as C.I.E. XYZ space tristimulus values. The image is transformed from XYZ space to some desired color space, coded, transmitted, and decoded in this space, and then transformed back to XYZ space. Figure 2.7 shows the block diagram of this coding system. The coding can be any type. Our main emphasis is on subband/VQ coding since it provides a sufficiently general framework for study, while it is also a viable

choice for high bit-rate applications. The transformation of the reconstructed XYZ image to the color space of the display is also shown in the figure. This will be discussed in the subsection on gamma correction.

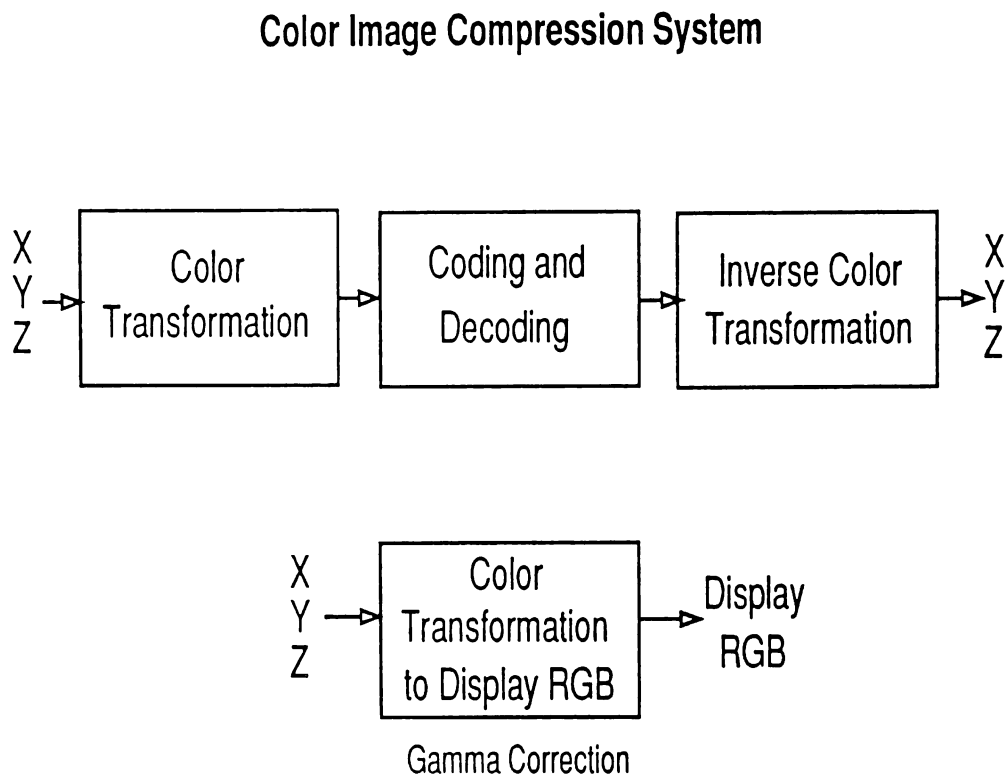


Figure 2.7: System for Compressing Color Images in Alternative Color Spaces.

2.2.2 Distortion Measures

Much of the original work in evaluating the sensitivity of the human visual system to luminance errors uses a model of the visual system to compute various distortion measures [57, 9] between an original and a processed image. Many of the models of the human visual system assume that there is an initial nonlinearity, followed by a

linear system that is frequency and orientation dependent [60]. Subjects were asked to judge the quality of the images and the subjective rating was then compared to the distortion measure to determine the usefulness of the distortion measure. This line of work has since been extended to color images [20]. A result of this work is that one can obtain a better distortion measure if the original image and the processed images are first transformed to a perceptually uniform color space and the distortion measure is calculated in this space.

To extend this approach to processing in alternative color spaces, the input and output XYZ images are transformed to a uniform color space and the distortion measure is calculated in this space. We implemented this method because it is mathematically tractable and because it provides an initial numerical assessment of the coding algorithms. $L^*a^*b^*$ space was used as the uniform color space. The distortion measure is ΔE, given by:

$$\Delta E = \|\mathbf{x} - \hat{\mathbf{x}}\| \tag{2.11}$$

where $\mathbf{x} = (L_i^* \ a_i^* \ b_i^*)^T$, $\hat{\mathbf{x}} = (L_o^* \ a_o^* \ b_o^*)^T$, $\|\cdot\|$ is the Euclidean norm, and the subscripts i and o refer to the input and output colors, respectively. ΔE measures the difference between two colors, it does not directly measure spatial errors. This distortion measure was computed on a pixel by pixel basis to provide error images showing the spatial distribution of the errors. The average value of ΔE over the image was also computed to provide a single error measure for each image.

Since the use of the ΔE distortion measure is prominent throughout the rest of this work, a few words should be said about its relation to subjective perception. A standard "rule-of-thumb" is that a ΔE error of less than three is not visible. This is slightly conservative since a mapping of the MacAdam's ellipses to $L^*a^*b^*$ space

shows that the just noticeable difference is larger than this for most regions of the color space. A comparison to standard color television is also interesting. The N.T.S.C. did not try to design a system that yielded "true color". They wanted the displayed image to be subjectively pleasing, even if there was a small change in color from the original scene. In terms of ΔE error, this change can be rather large.

For example, consider a white illuminant with equal energy of unity throughout the visible spectrum. The use of the N.T.S.C. color matching functions yields an RGB tristimulus vector of $(11.614\ 10.553\ 8.907)^T$. However, it was initially very difficult to create a camera system that replicated the negative lobes on the color matching functions while maintaining an acceptable signal to noise ratio. Instead of trying to find a different set of color matching functions that were all positive and resulted in a smaller error, as done in [100], the positive portions of the color matching functions were used. These are shown in Figure 2.8. The same equal energy illuminant now yields the tristimulus vector $(12.941\ 12.388\ 9.369)^T$. Using the second vector as the white point and transforming to $L^*a^*b^*$ space results in a ΔE error of 8.24.

2.3 Display of XYZ Images

2.3.1 Image Characteristics

An image of a toy store of size 1365×1365 pixels was recorded in XYZ tristimulus values. Each pixel is stored with 16 bits per color component. Two 256×256 windows of this image were chosen to be our input images. The first one consists primarily of a girl's face and was picked so that the coding algorithms could be tested on flesh tones. The second image contains a number of saturated colors and is used to test how accurately the coding algorithms can code these type of images. Four other images

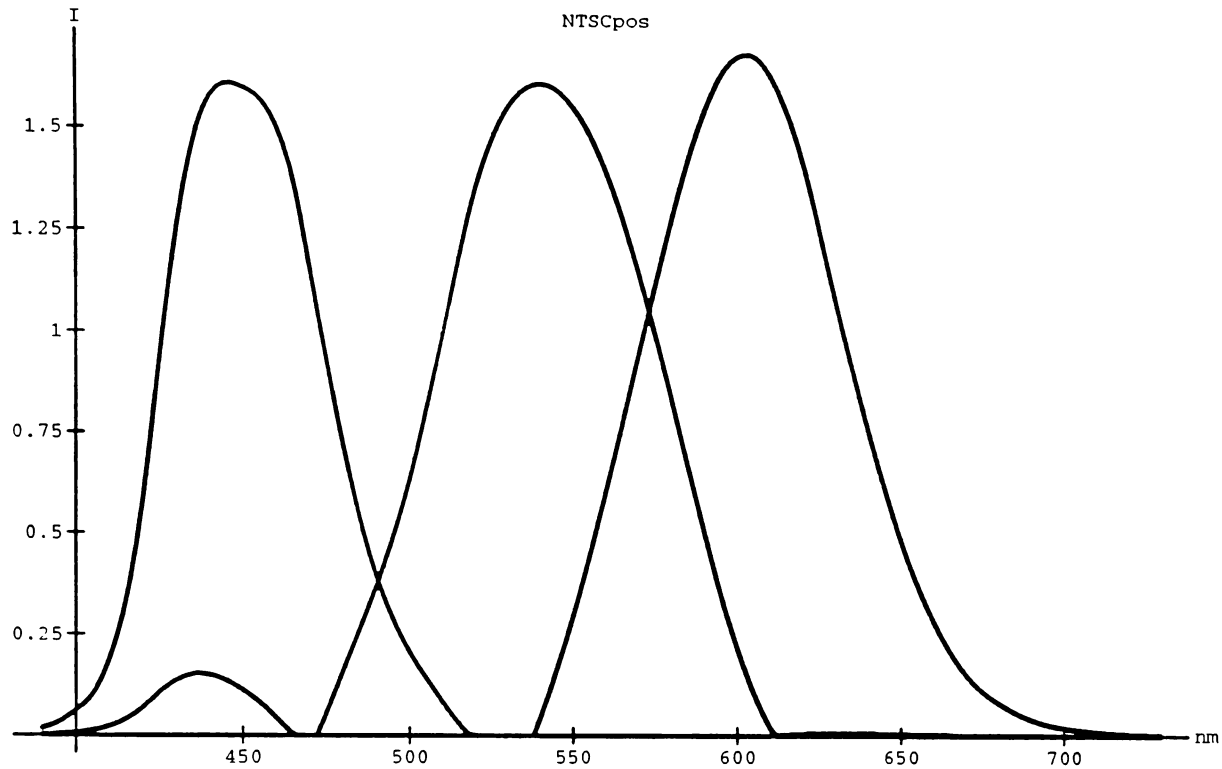


Figure 2.8: N.T.S.C. truncated color matching functions.

of this same size were taken from the toy store image to provide training sequences for the vector quantizer design.

The toy store image is too large to be displayed directly on our monitor. An image of size 682×682 pixels was created by subsampling by a factor of two in both the horizontal and vertical directions. Figure 2.9 shows a photograph of this image. Figure 2.10 shows the original, uncoded GIRL and DOLL images.

2.3.2 Calibration of the Color Monitor

In order to accurately display color images, our color monitor was calibrated. By this it is meant that a measurement of the XYZ tristimulus values transmitted from the display match the internal XYZ values of the image being displayed. The video

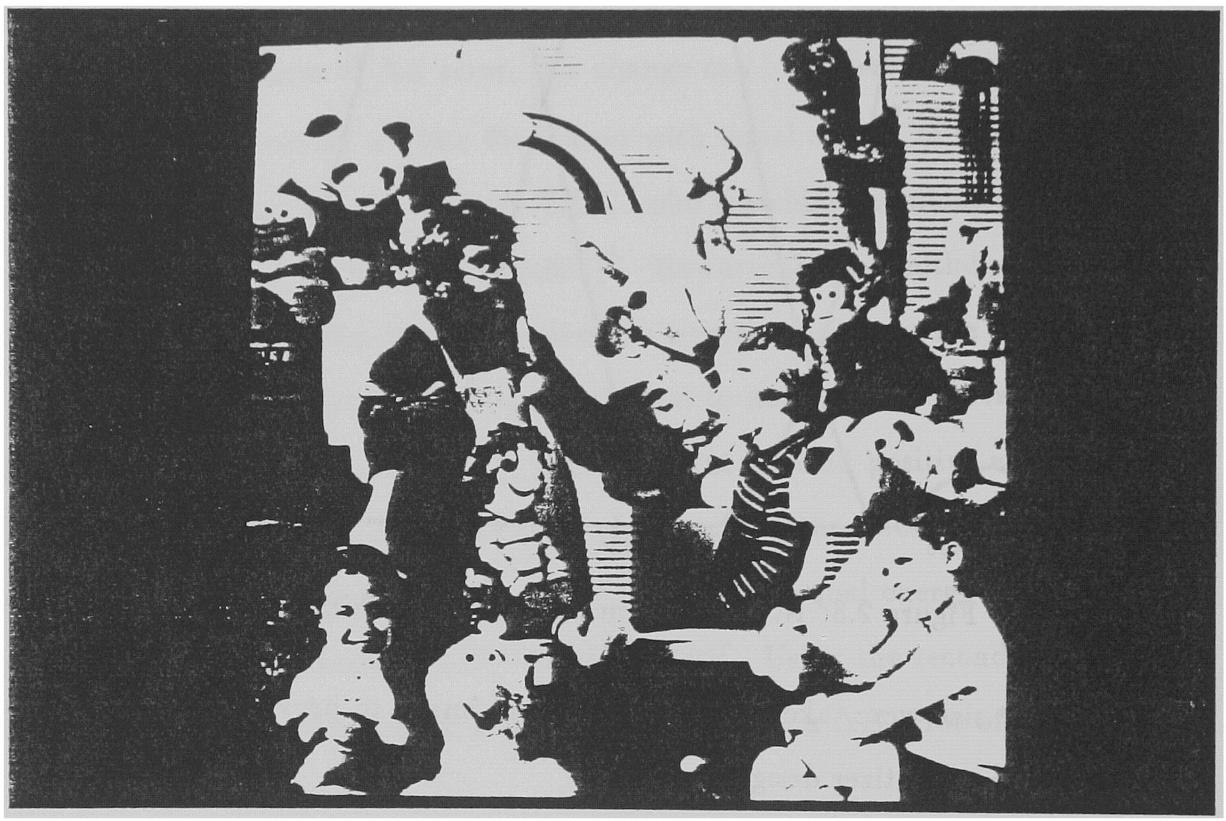


Figure 2.9: Toy Store Image subsampled by a factor of two in each direction.

buffer in the monitor drives the three digital-to-analog converters that provide the inputs to the electron guns. These guns excite the red, green, and blue phosphors, respectively. Therefore, an image is stored in the video buffer as red, green, and blue color components. Each component is stored with eight bits of accuracy per pixel, so 24 bits are required to represent the color of a pixel.

If we call the color space of the monitor Display *RGB* to distinguish it from N.T.S.C. *RGB* space, then the calibration of the monitor entails the determination

Figure 2.9: Toy Store Image subsampled by a factor of two in each direction.

buffer in the monitor drives the three digital-to-analog converters that provide the inputs to the electron guns. These guns excite the red, green, and blue phosphors,

of the transformation from XYZ space to Display RGB space. The derivation of this transformation is given in detail in [56]. The transformation to Display RGB space consists of two parts. The first part is a linear transformation from XYZ space to the red-green-blue color space of the frame buffer (rgb space). The second part is a nonlinear transformation that is needed to compensate for the transfer functions of the three electron guns.

Historically, the transfer function for an electron tube above cutoff has been modeled as a power law relationship of the form [7]

$$T = a[(E_0 - E_c) + E)]^{\gamma} \tag{2.12}$$

where T is the light output of the tube in total C.I.E. tristimulus value, E_0 is the bias voltage added to the signal, E_c is the cutoff voltage, E is the signal voltage, and a and γ are constants. Applying the inverse of Eq. (2.12) to the signal before it enters the tube is known as gamma correction.

The color display is characterized by six parameters. These include the values of gamma for each of the three colors, and the minimum visible pixel values, n_0, for each color. These parameters were experimentally determined for our particular monitor [56], and are:

$$\begin{aligned} \gamma_r &= 2.310 \quad n_{0_r} = 23.56 \\ \gamma_g &= 2.237 \quad n_{0_g} = 30.47 \\ \gamma_b &= 2.200 \quad n_{0_b} = 29.61. \end{aligned} \tag{2.13}$$

The gamma correction is done using the nonlinear transformations

$$\begin{aligned} R_D &= (255 - n_{0_r})r^{1/\gamma_r} + n_{0_r} \\ G_D &= (255 - n_{0_g})g^{1/\gamma_g} + n_{0_g} \\ B_D &= (255 - n_{0_b})b^{1/\gamma_b} + n_{0_b} \end{aligned} \tag{2.14}$$

where the subscript, D, refers to the displayed values. The rgb values are obtained from a linear transformation of XYZ space. This transformation is monitor dependent and in our case is given by:

$$\begin{bmatrix} r \\ g \\ b \end{bmatrix} = \begin{bmatrix} 0.073289 & -0.036415 & -0.010909 \\ -0.024990 & 0.042327 & 0.001208 \\ 0.001352 & -0.004255 & 0.023685 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.15}$$

where all of the rgb values lie between zero and one.

2.3.3 Monitor's White Point and Gamut

The minimum tristimulus values are zero and the maximum tristimulus values are obtained when the Display RGB values are $(255\ 255\ 255)^T$. Performing an inverse gamma correction on this vector gives the maximum XYZ tristimulus values of:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix}_{max} = \begin{bmatrix} 45.15 \\ 48.90 \\ 48.43 \end{bmatrix} \tag{2.16}$$

These values correspond to the white point of the display.

It was desired that the color errors resulting from processing be allowed to be both above and below the white point. The Display RGB values were reduced by 10

percent and then transformed to XYZ space. The resulting tristimulus values are now defined to be the white point of the display. These values are:

$$\begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix} = \begin{bmatrix} 34.51 \\ 37.36 \\ 37.16 \end{bmatrix}. \tag{2.17}$$

The corresponding chromaticity coordinates of the monitor's white point can now be calculated. They are given in Table 2.2 along with the chromaticity coordinates of the phosphors. The phosphor chromaticities can be measured directly with a colorimeter, or they can be calculated by gamma correcting the Display RGB vectors $(255\ 0\ 0)^T$, $(0\ 255\ 0)^T$, and $(0\ 0\ 255)^T$. Notice that they are not the same as the N.T.S.C. phosphors. Figure 2.11 shows the gamut of the monitor for the luminance values $Y = 5$, $Y = 10$, and $Y = 20\ cd/m^2$. One can see that the monitor cannot reproduce saturated blues at the higher luminance values.

Color	x	y
Red	0.6106	0.3596
Green	0.3110	0.5991
Blue	0.1495	0.0659
White	0.3165	0.3427

Table 2.2: Monitor Phosphor Chromaticities

2.4 Color Transformations

2.4.1 Transformation to N.T.S.C. RGB Space

The mapping from the XYZ tristimulus values to the RGB values is a linear transformation that is based on the N.T.S.C. phosphors. It uses standard illuminant C as

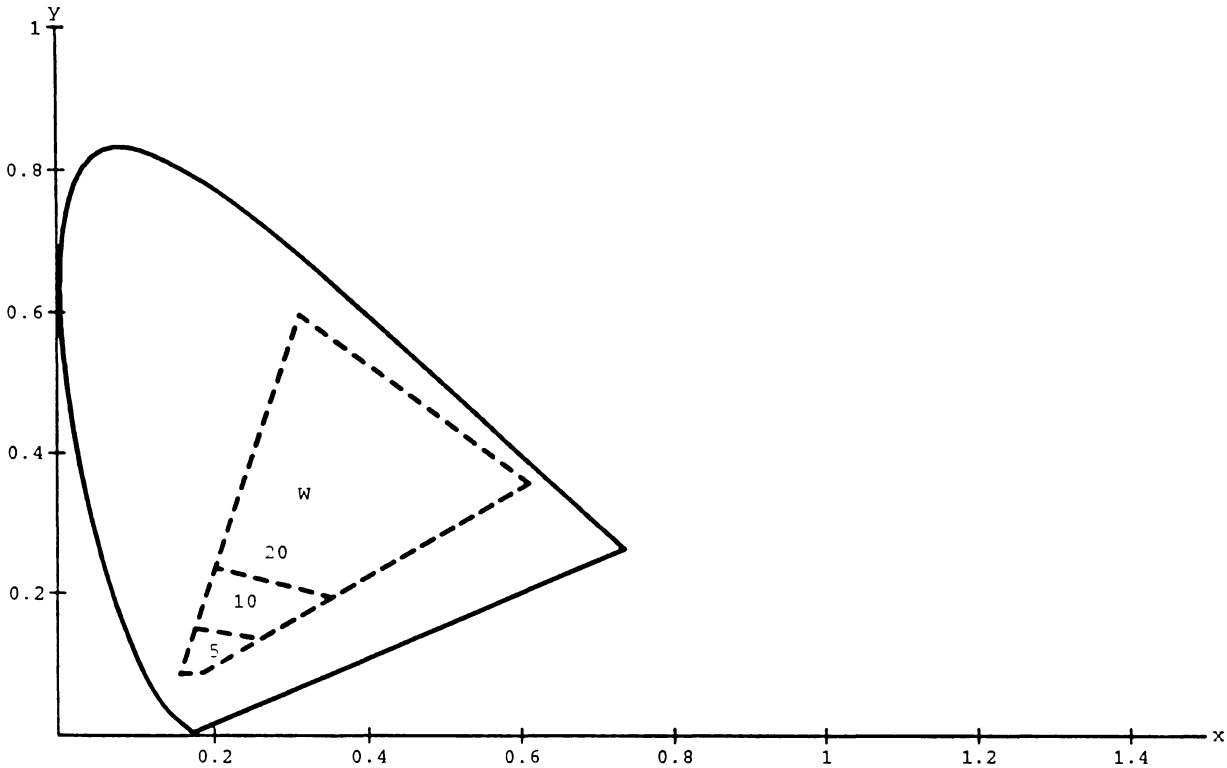


Figure 2.11: Monitor's Gamut for Luminance Values $Y = 5$,$Y = 10$, and $Y = 20\ cd/m^2$. The gamuts for the different luminance values are denoted by the dashed lines. The white point has chromaticity coordinates (0.3165,0.3427), and is denoted by the symbol W. The Spectral Locus and the Line of Purples are also plotted.

the white point and is given by [87]:

$$\begin{bmatrix} R \\ G \\ B \end{bmatrix} = \begin{bmatrix} 1.9098 & -0.5324 & -0.2882 \\ -0.9846 & 1.9991 & -0.0283 \\ 0.0583 & -0.1184 & 0.8980 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} \tag{2.18}$$

2.4.2 Transformation to N.T.S.C. YIQ Space

Because of a desire for compatibility with the existing monochrome system, the N.T.S.C. decided to create a color system with one luminance channel, Y, and two chrominance channels, I and Q. The new color signal had to occupy the same band-

width as the existing signal, so the chrominance channels were quadrature modulated onto a subcarrier that was then added to the monochrome signal. The chrominance channels are derived from scaled and rotated versions of $(R-Y)$ and $(B-Y)$, where R and B are the red and blue components in N.T.S.C. RGB space. The bandwidth of the chrominance channels had to be reduced to minimize interference with the luminance channel. To reduce the visual degradations as much as possible, a series of experiments were conducted to determine the rotation angle used to create the I and Q axes. An angle of 33 degrees was chosen.

The transformation from XYZ to YIQ space takes place in two steps. The first step is the transformation from XYZ to RGB shown above. The second step is from RGB to YIQ. This is a linear formulation of the N.T.S.C. transformation [87]. The two matrices can be combined to give:

$$\begin{bmatrix} Y \\ I \\ Q \end{bmatrix} = \begin{bmatrix} 0.000 & 1.000 & 0.000 \\ 1.389 & -0.827 & -0.453 \\ 0.938 & -1.195 & 0.233 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{2.19}$$

2.4.3 Design of YI^*Q^* Space

A recent study of the sensitivity of the human visual system to color-varying stimuli measured the minimum detection thresholds for color transitions along a red-green and a blue-yellow direction [46]. These directions were chosen because they are visually, but not mathematically, orthogonal. This means that a transition along one of these color directions does not affect the HVS's response in the other direction. The mean detection threshold data derived from this study will be used to derive a perceptually optimal bit allocation for a color subband coding system.

To make the best use of this data, the chrominance axes of the color space used should lie along the red-green and blue-yellow directions used in the study. Unfortunately, the I and Q axes in YIQ space do not lie along these directions, but are rotated by roughly 33 degrees. A new color space, YI^*Q^*, was desired that is similar to N.T.S.C. YIQ space except that the chrominance axes are closer to the directions used in the visual experiment. The derivation of this color space will now be given. Chapter 6 will discuss the visual experiment in more detail, and will show the directions of the color transitions in the various color spaces.

The new color space is based on the N.T.S.C. phosphors and the white point of our color monitor. First a transformation to XYZ space from a new RGB space is derived. This RGB space is created by determining the gains on each of the red, green, and blue channels so that an RGB vector of $(1\ 1\ 1)^T$ maps to an XYZ vector equal to the monitor's white point with the luminance scaled to unity. The transformation is:

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.5734 & 0.1842 & 0.1659 \\ 0.2824 & 0.6228 & 0.0948 \\ 0.0000 & 0.0702 & 0.9243 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{2.20}$$

The I^* and Q^* channels are given by:

$$\begin{aligned} I^* &= (R-Y)/1.14 \\ Q^* &= (B-Y)/2.03. \end{aligned} \tag{2.21}$$

Notice that so far, the only differences from N.T.S.C. YIQ space is that a different white point is used and there is no rotation of the $(R-Y)/1.14$ and $(B-Y)/2.03$ axes.

Since the Y in XYZ space is the same as the Y in YI^*Q^* space, one can combine

Eqs. (2.20) and (2.21) to get:

$$\begin{bmatrix} Y \\ I^* \\ Q^* \end{bmatrix} = \begin{bmatrix} 0.2824 & 0.6228 & 0.0948 \\ 0.6295 & -0.5463 & -0.0832 \\ -0.1391 & -0.3068 & 0.4459 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix}. \tag{2.22}$$

Replacing the $(R\ G\ B)^T$ vector in Eq. (2.22) with the inverse of Eq. (2.20) gives the desired result:

$$\begin{bmatrix} Y \\ I^* \\ Q^* \end{bmatrix} = \begin{bmatrix} 0.0000 & 1.0000 & 0.0000 \\ 1.7733 & -1.3715 & -0.2676 \\ 0.0347 & -0.5630 & 0.5339 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{2.23}$$

2.4.4 Transformation to C.I.E. $L^*a^*b^*$ Space

$L^*a^*b^*$ space is an almost perceptually uniform color space created by a nonlinear transformation of XYZ space. The transformation needs the XYZ coordinates of the desired white point. In our simulations, we used the defined white point of our display; the numerical values are given in Eq. (2.17).

The transformation from XYZ to the CIELAB color space $L^*a^*b^*$ is given by the equations [105]:

$$\begin{aligned} L^* &= 116(Y/Y_0)^{1/3} - 16 \\ a^* &= 500[(X/X_0)^{1/3} - (Y/Y_0)^{1/3}] \\ b^* &= 200[(Y/Y_0)^{1/3} - (Z/Z_0)^{1/3}] \end{aligned} \tag{2.24}$$

where X_0, Y_0, and Z_0 are the tristimulus values of white. These formulae are correct only for values of X/X_0, Y/Y_0, and Z/Z_0 greater than 0.008856. For lower values of these ratios, $L^* = 903.29(Y/Y_0)$, and a^* and b^* are also changed.

2.4.5 White Point Mapping of AC_1C_2 Space

The second perceptually uniform color space that we shall use is based on a logarithmic transformation of cone space derived by Faugeras [21]. The cone space, LMS, is obtained from a linear transformation of XYZ space [105] using the equation:

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 661.0 & 1260.0 & -112.0 \\ -438.0 & 1620.0 & 123.0 \\ 0.708 & 0.0 & 417.0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{2.25}$$

The transformation to LMS space given by Eq. (2.25) does not map the desired white point to the origin of the coordinate system. To be consistent with the transformation to $L^*a^*b^*$ space, the transformation was changed so that the monitor's white point is mapped to the origin. Changing the white point is known as white point mapping, and the procedure to calculate the new matrix is now given.

Define the matrix E to be the transformation matrix in Eq. (2.25). Let the vector w be the XYZ tristimulus values and let l be the corresponding LMS tristimulus values defined by $\mathbf{l} = \mathbf{E}\mathbf{w}$. It is desired that

$$\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \mathbf{l}_m = \mathbf{E}_1\mathbf{w}_m, \tag{2.26}$$

where $\mathbf{E}_1$ is the new transformation matrix, and $\mathbf{w}_m$ contains the tristimulus values of the monitor's white point with the luminance scaled to unity. The logarithmic processing in Eq. (2.31) will map the vector $(1\ 1\ 1)^T$ to $(0\ 0\ 0)^T$. The white point vector is given by:

$$\mathbf{w}_m = \begin{bmatrix} 0.9235 \\ 1.0000 \\ 0.9945 \end{bmatrix}. \tag{2.27}$$

Now define $\mathbf{E}_1^{-1} = \mathbf{E}^{-1}\mathbf{\Lambda}$ where $\mathbf{\Lambda}$ is the diagonal matrix of the gains needed to map the white point to the origin. Once $\mathbf{\Lambda}$ is found, one can calculate $\mathbf{E}_1$ and use it instead of $\mathbf{E}$ in Eq. (2.25). Premultiplying both sides of Eq. (2.26) by $\mathbf{E}_1^{-1}$ yields:

$$\mathbf{w}_m = \mathbf{E}_1^{-1} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \mathbf{E}^{-1}\mathbf{\Lambda} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}. \tag{2.28}$$

This gives:

$$\mathbf{\Lambda} \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \mathbf{E}\mathbf{w}_m = \begin{bmatrix} 1759.32 \\ 1337.67 \\ 415.57 \end{bmatrix}. \tag{2.29}$$

Finally, the new matrix is: $\mathbf{E}_1 = \mathbf{\Lambda}^{-1}\mathbf{E}$.

2.4.6 Transformation to AC_1C_2 Space

The transformation from XYZ to AC_1C_2 space occurs in three steps. The first step is a linear transformation from XYZ space to cone space using matrix $\mathbf{E}_1$ derived in the last section. This is followed by a logarithmic mapping and another linear transformation. The processing is as follows:

$$\begin{bmatrix} L \\ M \\ S \end{bmatrix} = \begin{bmatrix} 0.37512 & 0.71618 & -0.06366 \\ -0.32743 & 1.21106 & 0.09195 \\ 0.00170 & 0.00000 & 1.00344 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}. \tag{2.30}$$

$$\begin{aligned} l &= \ln(L) \\ m &= \ln(M) \\ s &= \ln(S) \end{aligned} \tag{2.31}$$

$$\begin{bmatrix} A \\ C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} 13.8310 & 8.3394 & 0.4294 \\ 64.0000 & -64.0000 & 0.0000 \\ 10.0000 & 0.0000 & -10.0000 \end{bmatrix} \begin{bmatrix} l \\ m \\ s \end{bmatrix} \tag{2.32}$$

The last transformation is the achromatic-chromatic separation. It creates the A channel containing the luminance information and two chrominance channels. C_1 contains red-green information and C_2 contains blue-yellow information. Frequency selective filtering can be applied to this color space to create the color space $A^*C_1^*C_2^*$. Since the subband coding is also frequency selective, this filtering was not done. The human visual system's frequency dependence will be considered in the bit allocation algorithm.

CHAPTER 3

SUBBAND CODING

3.1 Introduction

Multiresolution transformations such as the Laplacian Pyramid [10] and wavelets [55] have obtained an important place in image coding. While the former method is both conceptually straightforward and simple to implement, it increases the number of pixels that must be coded by a factor of 4/3. The latter method does not suffer from this problem, but it requires that a particular scaling function be found. A type of multiresolution transformation that has proven to be of particular utility is known as subband coding. This technique predates the wavelet analysis and was originally used for speech processing [13].

A short history of subband coding was given in Chapter 1, where the emphasis was on its use in image coding. In this chapter, we concentrate on the theoretical aspects of the filtering of a one-dimensional signal or a two-dimensional image into subbands. Without any coding of the subbands, there are still three types of errors that can occur. The first type of error is due to the aliasing caused by the decimation process. For a two-band decomposition, Esteban and Galand [17] showed how to eliminate this by using quadrature mirror filters. Even after removing the aliasing, there can still be amplitude and phase distortion. Smith and Barnwell [85] showed that it is possible to remove all three types of distortion in a tree-structured subband system. In a tree-structured system, an M band decomposition is achieved by cascading a

number of two band decompositions.

A subband system that does not have any aliasing, amplitude distortion, or phase distortion is said to have perfect reconstruction. The output is a delayed version of the input. A matrix formulation for the problem of achieving perfect reconstruction for multirate filterbanks was separately derived by Smith and Barnwell [86], and Vetterli [98]. The former refer to this as the AC-Matrix Formulation. The AC-Matrix Formulation is more general than the tree-structured subband decomposition since it can be used to analyze and design an M band decomposition where $M \geq 2$. Further properties are given in [99].

The extension of subband coding to images was done by Woods and O'Neil [104] using separable one-dimensional filters. Although this leads to a simple implementation, there are a number of researchers who advocate the use of non-separable ones. Mahesh and Pearlman [54] linearly interpolated the 256×256 "Lena" image and sampled it on a hexagonal lattice to obtain an image that contained 364×128 samples. The image was then sent through a subband coding system that used hexagonal filters. The resulting subimages are directionally selective. The advantage of such a system is that the human visual system is also directionally selective. There are indications that the retinal photoreceptors are arranged on a non-uniform hexagonal lattice. More bits can be allocated to the subbands that are the most visually important.

Simoncelli and Adelson [84] have also implemented subband systems with hexagonal quadrature mirror filters. They report good results for gray-scale images at 0.5 bits per pixel. One can also sample the image with a quincunx lattice and use non-separable diamond shaped filters to generate a quincunx pyramid. If one desires to have directional selectivity while still using separable filters, the method of Li and

He [48] can be used. After separable filters are used to split the image into four subimages, the band-pass subimages are filtered with directional filters.

Recently, much attention has been focused on general two-dimensional systems. Vaidyanathan [93] showed that it is possible to design two-dimensional quadrature mirror filterbanks with perfect reconstruction through the use of lossless transfer matrices, and Karlsson and Vetterli [42] presented conditions for alias-free and perfect reconstruction filterbanks. Bamberger [3, 4] has designed two-dimensional non-separable filters, and used them to create a number of new subband decompositions. In this chapter, we present a new diamond subband decomposition using separable quadrature mirror filters. The basic idea for this system comes from Sullivan [88].

The format of this chapter is as follows. In the next section, the one-dimensional theory is presented for both quadrature mirror filters and perfect reconstruction filters. The first subsection contains a review of decimators and interpolators, since their understanding is essential in studying subband coders. The next three subsections give the conditions necessary to achieve aliasing cancellation, and those necessary to obtain perfect reproduction. The following section extends the discussion to the two-dimensional case using separable quadrature mirror filters. The fourth section derives the new diamond subband coder. The fifth section presents the rectangular and diamond configurations used in our computer simulations, and discusses some of their implementation aspects.

3.2 One-Dimensional Theory

A subband coding system is an example of a multirate digital system since there is more than one sampling rate present in the system. The theory of such systems is

presented in depth in the book by Crochiere and Rabiner [14]. These systems are, in general, time-varying, but can be made time-invariant if the aliasing is cancelled. The subband system consists of two sections, the analysis section and the synthesis section. The analysis sections splits the input signal into a number of signals by using a bank of filters; each signal is then decimated. The synthesis section reconstructs a signal by interpolating a number of signals, filtering them with a bank of filters, and then combining the signals on a sample by sample basis. A two band system is shown in Figure 3.1.

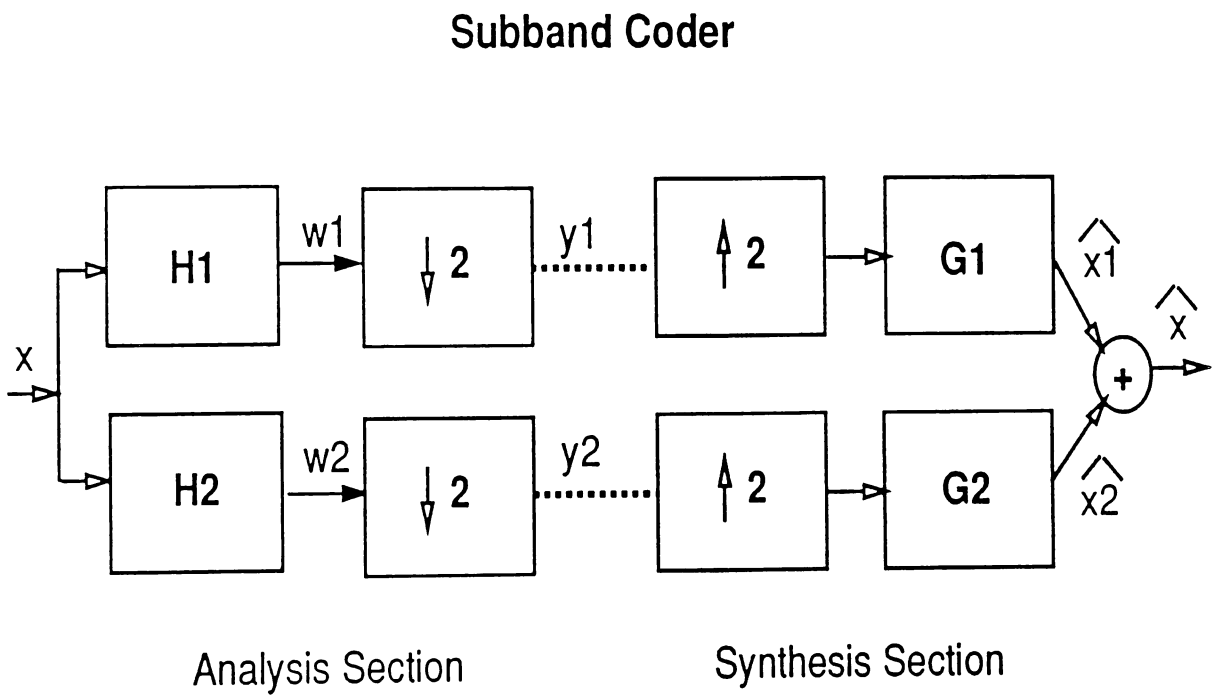


Figure 3.1: Two Band One-Dimensional Subband System.

3.2.1 Decimators and Interpolators

Time and Frequency Responses

The digital decimator is a device the subsamples a digital sequence by some fixed number, M. That is, it keeps every M^{th} sample and discards the rest, resulting in a lower bit rate. The time domain representation of this operation is given by

$$y(n) = w(Mn) \tag{3.1}$$

where $w(n)$ is the input and $y(n)$ is the output of the decimator. The z-transform domain representation of the decimator is given by

$$Y(z) = \frac{1}{M} \sum_{l=0}^{M-1} W(e^{-j2\pi l/M} z^{1/M}) \tag{3.2}$$

where $Y(z)$ and $W(z)$ are the z-transforms of $y(n)$ and $w(n)$.

Examining Eq. (3.2) shows that the output signal contains a subsampled version of the input signal plus $M-1$ modulated versions of the subsampled input. If $w(n)$ has a small enough bandwidth, these aliased components will not overlap, and the original signal can be recovered. This can be achieved by first filtering the input signal by a lowpass or bandpass filter. Figure 3.2 shows the block diagrams of decimation and interpolation. The filter, H, is used to band-limit the input signal, $x(n)$, to get the input to the decimator, $w(n)$.

The digital interpolator upsamples the input signal by a factor of L by inserting $L-1$ zero samples between each input sample. This can be written in the time domain as

$$y(n) = \begin{cases} w(\frac{n}{L}) & \text{if } n = 0, \pm L, \pm 2L, \ldots \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

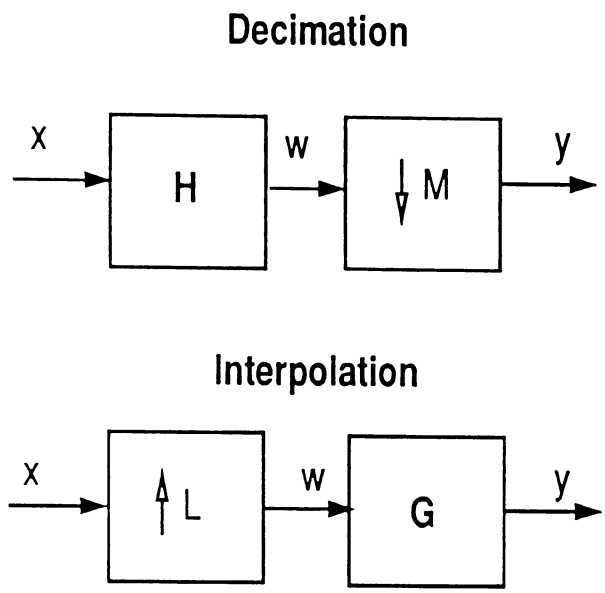


Figure 3.2: Block Diagrams for Decimation by M and Interpolation by L.

and in the z-transform domain as

$$Y(z) = W(z^L). \tag{3.4}$$

Because of the periodic nature of the spectrum of digital signals, the interpolation process causes imaging. The images are removed by filtering the interpolated signal; the block diagram for this operation is shown in Figure 3.2. Substituting $e^{j\omega}$ for z in the above equations gives the discrete Fourier transform representation of decimation and interpolation.

Polyphase Implementation

In the two subband system of Figure 3.1, the analysis filters preceded the decimators and the synthesis filters followed the interpolators. Therefore, all of the filtering was done at the higher sampling rate. It is possible to construct a more computationally efficient system where the filtering is done at the lower sampling rate. To do this, the

concept of polyphase filters is needed. For decimation, the polyphase filters are based on subsampling the impulse responses of the analysis filters by the decimation factor, M. The M polyphase filters, $P_0, P_1 \ldots P_{M-1}$, are created for each analysis filter by shifting the starting place of the sampling by one sample (for more details see [14]). Figure 3.3 shows the polyphase implementation of the decimation system shown in Figure 3.2. Notice that the decimation is now done first, and the filtering is done in parallel at the lower sampling rate.

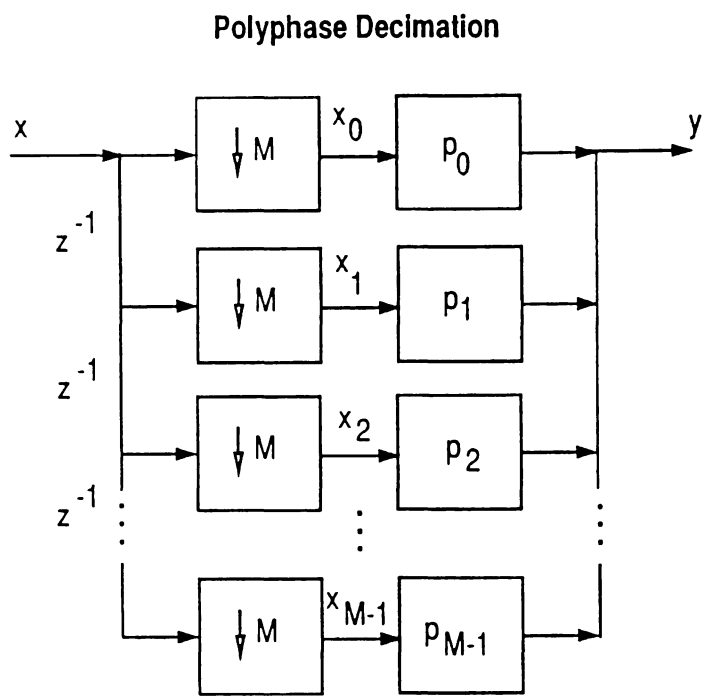


Figure 3.3: Polyphase Implementation of Decimation by a factor of M.

Similarly, the polyphase filters for the synthesis section are obtained by sampling the synthesis filters in the same way. Again, L polyphase filters are obtained from each synthesis filter. Figure 3.4 shows the polyphase implementation of the interpolation system. Here, the filtering is done in parallel before the interpolation, and, therefore,

at the lower sampling rate.

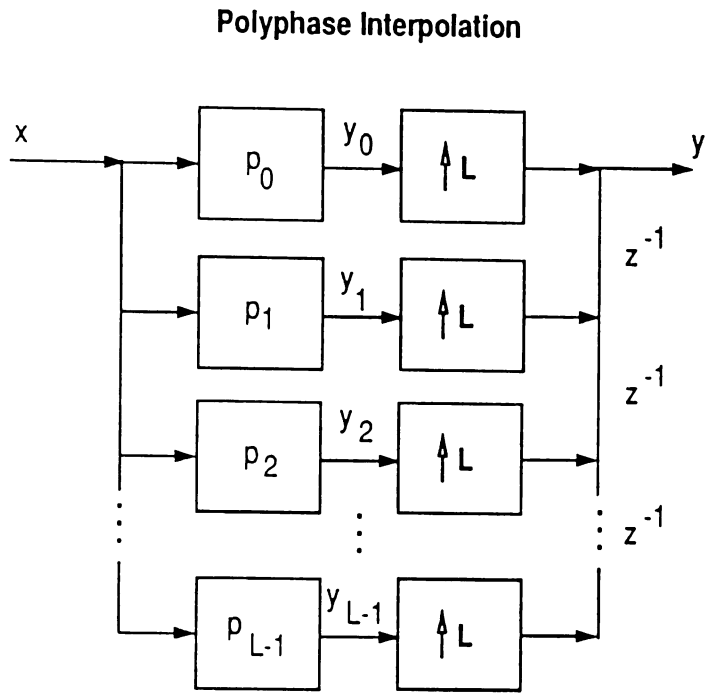


Figure 3.4: Polyphase Implementation of Interpolation by a factor of L.

3.2.2 Quadrature Mirror Filters

The first implementations of all digital subband coding used FIR filters to separate the subbands. Originally, there were transition regions between the subbands in order to ensure frequency separation. Unfortunately, this method removed some of the audio information and caused noticeable degradation. By allowing the frequency bands to overlap, this degradation can be removed but in doing so, aliasing is introduced. In addition to the aliasing, the filtering process can also introduce both amplitude and phase distortion into the reconstructed signal.

It was found that by specifying certain conditions on the filters, the effects of

aliasing could be eliminated [17]. The derivation of these conditions will now be given; it essentially follows the derivation in Smith and Barnwell [85]. The block diagram for the two band system is shown in Figure 3.1. The input signal, $x(n)$, with discrete Fourier transform $X(e^{j\omega})$ is split into two signals $w_1(n)$ and $w_2(n)$ by filtering with $H_1(e^{j\omega})$ and $H_2(e^{j\omega})$. These signals are decimated by a factor of two to create the signals $y_1(n)$ and $y_2(n)$ whose Fourier transforms are given by

$$\begin{aligned} Y_1(e^{j\omega}) &= 1/2[H_1(e^{j\omega/2})X(e^{j\omega/2}) + H_1(-e^{j\omega/2})X(-e^{j\omega/2})] \\ Y_2(e^{j\omega}) &= 1/2[H_2(e^{j\omega/2})X(e^{j\omega/2}) + H_2(-e^{j\omega/2})X(-e^{j\omega/2})]. \end{aligned} \tag{3.5}$$

The signals, $\hat{x}_1(n)$ and $\hat{x}_2(n)$, are created by interpolating and filtering $y_1(n)$ and $y_2(n)$

$$\begin{aligned} \hat{X}_1(e^{j\omega}) &= Y_1(e^{j\omega 2})G_1(e^{j\omega}) \\ \hat{X}_2(e^{j\omega}) &= Y_2(e^{j\omega 2})G_2(e^{j\omega}). \end{aligned} \tag{3.6}$$

The reconstructed signal, $\hat{x}(n)$, is then created by adding together the signals $\hat{x}_1(n)$ and $\hat{x}_2(n)$. In the frequency domain this yields

$$\begin{aligned} \hat{X}(e^{j\omega}) &= 1/2[H_1(e^{j\omega})G_1(e^{j\omega}) + H_2(e^{j\omega})G_2(e^{j\omega})]X(e^{j\omega}) + \\ &\quad 1/2[H_1(-e^{j\omega})G_1(e^{j\omega}) + H_2(-e^{j\omega})G_2(e^{j\omega})]X(-e^{j\omega}). \end{aligned} \tag{3.7}$$

The first term represents the desired signal and the second term contains the aliasing. The aliasing can be removed by defining the reconstruction filters to be

$$G_1(e^{j\omega}) = H_2(-e^{j\omega}), \;\; G_2(e^{j\omega}) = -H_1(-e^{j\omega}). \tag{3.8}$$

The resulting system transfer function is then given by

$$T(e^{j\omega}) = \frac{1}{2}[H_1(e^{j\omega})H_2(-e^{j\omega})] - \frac{1}{2}[H_2(e^{j\omega})H_1(-e^{j\omega})]. \tag{3.9}$$

The reconstruction filters that cancel the aliasing are known as Quadrature Mirror Filters or QMF's. The are usually chosen so that they have an even number of taps and

$$H_2(e^{j\omega}) = H_1(-e^{j\omega}). \tag{3.10}$$

The exact reconstruction condition now becomes

$$T(e^{j\omega}) = \frac{1}{2}H_1^2(e^{j\omega}) - \frac{1}{2}H_2^2(e^{j\omega}) = e^{-j\omega n_0}. \tag{3.11}$$

The delay, n_0, is added to allow the filters to be causal. For image processing, the filters can be non-causal and the delay reduces to 0. Johnston designed a number of QMF filters that approximately meet this condition [41], but there are two cases that exactly meet it. The first is the ideal half-band filter and the second consists of the two-tap filters

$$H_1(z) = 1 + z^{-1}, \quad H_2(z) = 1 - z^{-1}. \tag{3.12}$$

If $T(e^{j\omega}) \neq e^{-j\omega n_0}$, then there will be distortion even though the aliasing has been eliminated. This distortion can be either amplitude and/or phase distortion. If $|T(e^{j\omega})| = K$, where K is some constant, then there will be no amplitude distortion. The phase distortion can be eliminated if $arg[T(e^{j\omega})] = K\omega$. This can be achieved if the transfer function is a linear phase FIR function. If $H_1(e^{j\omega})$ is a linear phase FIR filter of order $N-1$, it can be written in the form

$$H_1(e^{j\omega}) = e^{-j\omega(N-1)/2}H_{1,a}(e^{j\omega}) \tag{3.13}$$

where $H_{1,a}(e^{j\omega})$ is the amplitude response [92]. Then,

$$T(e^{j\omega}) = \frac{e^{-j\omega(N-1)}}{2}[\ |H_1(e^{j\omega})|^2 - (-1)^{N-1}|H_2(e^{j\omega})|^2\ ]. \tag{3.14}$$

The restriction that the order of the filter be odd (even number of taps) is necessary to prevent Eq. (3.14) from having a value of zero at $\omega = \pi/2$. When this condition is true:

$$|T(e^{j\omega})| = \frac{1}{2}[\ |H_1(e^{j\omega})|^2 + |H_2(e^{j\omega})|^2\ ]. \tag{3.15}$$

3.2.3 Perfect Reconstruction Filters

One would like to be able to eliminate both amplitude and phase distortion as well as the aliasing. Except for the two tap case, this is not possible with the QMF filters defined in the last section. By specifying different conditions on the relationships among the analysis and reconstruction filters, all distortion can be eliminated [85]. The reconstruction filters are now chosen to be

$$G_1(e^{j\omega}) = H_2(-e^{j\omega}) \quad G_2(e^{j\omega}) = -H_1(-e^{j\omega}) \tag{3.16}$$

where the analysis filters are related by

$$H_2(e^{j\omega}) = -H_1(-e^{-j\omega})e^{-j\omega N} \tag{3.17}$$

with N an odd number. The transfer function is now given by

$$\begin{aligned} T(e^{j\omega}) &= \frac{1}{2}[H_1(e^{j\omega})H_1(e^{-j\omega}) + H_1(-e^{j\omega})H_1(-e^{-j\omega})]e^{-j\omega N} \\ &= \frac{1}{2}[F_1(e^{j\omega}) + F_2(e^{j\omega})]e^{-j\omega N}. \end{aligned} \tag{3.18}$$

Eq. (3.18) implies that the product filters are

$$F_1(e^{j\omega}) = H_1(e^{j\omega})H_1(e^{-j\omega}) \quad F_2(e^{j\omega}) = F_1(-e^{j\omega}). \tag{3.19}$$

The condition in the time domain for perfect reconstruction is now

$$t(n) = \frac{1}{2\pi}\int_{-\pi}^{\pi} T(e^{j\omega})e^{j\omega n}\,d\omega = \delta(n-N). \tag{3.20}$$

Since we are not concerned with causality, we can set the above equation equal to $\delta(n)$ instead. This makes the first product filter

$$f_1(n)[\frac{1+(-1)^n}{2}] = \delta(n). \tag{3.21}$$

To achieve this, decompose $f_1(n)$ into $f_1(n) = v(n) + A\delta(n)$ where $v(2n) = 0, \forall n$. Then

$$F_1(e^{j\omega}) = V(e^{j\omega}) + A \tag{3.22}$$

where $A = 1$, and $V(e^{j\omega})$ is a filter that is antisymmetric about $\pi/2$.

To design the perfect analysis filter $H_1(e^{j\omega})$ of order $N-1$, one first designs the product filter $F_1(e^{j\omega})$ of order $2(N-1)$. This can be done using one of many conventional digital filter design methods such as the McClellan-Parks algorithm [66], the Hofstetter algorithm [71], or a Kaiser window method [36]. The product filter is then factored to yield $H_1(e^{j\omega})$ which is no longer necessarily linear phase. It is a good idea to do the factorization so that the analysis filter is close to linear phase in order to minimize the effects of quantization.

3.2.4 AC-Matrix Formulation

The use of matrix notation in formulating the conditions necessary for removing aliasing and achieving perfect reconstruction can lead to a greater insight into the problem. This approach was taken by both Vetterli [98], and Smith and Barnwell [86].

The former defined the modulated filter matrix, $\mathbf{H}_m(z)$, as

$$\mathbf{H}_m(z) = \begin{bmatrix} H_0(z) & H_0(e^{-j2\pi/N}z) & \cdots & H_0(e^{-j2\pi(N-1)/N}z) \\ H_1(z) & H_1(e^{-j2\pi/N}z) & \cdots & H_1(e^{-j2\pi(N-1)/N}z) \\ \vdots & \vdots & \ddots & \vdots \\ H_{M-1}(z) & H_{M-1}(e^{-j2\pi/N}z) & \cdots & H_{M-1}(e^{-j2\pi(N-1)/N}z) \end{bmatrix} \tag{3.23}$$

where N is the decimation factor, M is the number of subbands, and the H_i are the analysis filters.

Critical sampling occurs when $N = M$, and in this case the modulated filter matrix is square. Critical sampling is assumed in the rest of the derivation. This matrix is the transpose of the aliasing components matrix (AC-matrix) of Smith and Barnwell. $\mathbf{H}_m(z)$ is also related to the polyphase components of the analysis filters according to

$$\mathbf{H}_m(z) = N\mathbf{H}_p(z)\mathbf{F}^{-1} \tag{3.24}$$

where $\mathbf{H}_p(z)$ is a matrix of the analysis polyphase filters, and $\mathbf{F}$ is the Fourier matrix with elements $\mathbf{F}_{ik} = e^{-j2\pi ik/N}$.

The synthesis filters can be written in vector notation as

$$\mathbf{g}(z) = [G_0(z)\ G_1(z) \cdots G_{M-1}(z)]^T \tag{3.25}$$

and the input can be expressed as

$$\mathbf{x}(z) = [X(z)\ X(e^{-j2\pi/N}z) \cdots X(e^{-j2\pi(N-1)/N}z)]^T. \tag{3.26}$$

The system equation can then be expressed as

$$\hat{\mathbf{x}}(z) = \frac{1}{N}[\mathbf{g}(z)]^T \cdot \mathbf{H}_m(z) \cdot \mathbf{x}(z). \tag{3.27}$$

All of the columns of $\mathbf{H}_m(z)$, except the first, lead to aliasing in the reconstructed signal. Therefore, to achieve aliasing cancellation it is necessary that

$$[\mathbf{g}(z)]^T \mathbf{H}_m(z) = [F(z)\ 0\ 0 \cdots 0] \tag{3.28}$$

where $F(z)$ is some filter. To achieve perfect reconstruction, it is required that

$$[\mathbf{g}(z)]^T \mathbf{H}_m(z) = [z^{-k}\ 0\ 0 \cdots 0] \tag{3.29}$$

where z^{-k} is an arbitrary delay. Since image processing can be non-causal, k is allowed to be zero.

Much research has been done to discover ways to solve Eqs. (3.28) and (3.29). The approaches usually entail placing constraints on the analysis and synthesis filters similar to those discussed in the previous two subsections. Often, the analysis filters are chosen, and the inverse (or pseudo-inverse) of the modulated filter matrix given in Eq. (3.23) is used to find the synthesis filters. Further discussion of this can be found in [92] and [98, 99].

3.3 Two-Dimensional Theory - Quadrature Mirror Filters

The extension of subband coding to two dimensions using QMF's follows. The procedure is from Woods and O'Neil [104] but the notation has been changed to match the above. The filter $H_{11}(e^{j\omega_1}, e^{j\omega_2})$ is a lowpass filter in each dimension and corresponds to a two-dimensional extension of $H_1(e^{j\omega})$. The two filters, $H_{12}(e^{j\omega_1}, e^{j\omega_2})$ and $H_{21}(e^{j\omega_1}, e^{j\omega_2})$, are lowpass in one direction and bandpass in the other direction. The remaining filter, $H_{22}(e^{j\omega_1}, e^{j\omega_2})$, is bandpass in each dimension. To avoid aliasing due to the decimation process, the filters on both the transmitter and the receiver side

must be properly related. On the transmitter side,

$$H_{12}(e^{j\omega_1}, e^{j\omega_2}) = H_{11}(e^{j\omega_1}, -e^{j\omega_2}), \quad H_{21}(e^{j\omega_1}, e^{j\omega_2}) = H_{11}(-e^{j\omega_1}, e^{j\omega_2}) \tag{3.30}$$

$$H_{22}(e^{j\omega_1}, e^{j\omega_2}) = H_{11}(-e^{j\omega_1}, -e^{j\omega_2}).$$

The receiver filters must be chosen so that the aliasing terms cancel. This can be achieved if

$$G_{11}(e^{j\omega_1}, e^{j\omega_2}) = 4H_{11}(e^{j\omega_1}, e^{j\omega_2}), \quad G_{12}(e^{j\omega_1}, e^{j\omega_2}) = -4H_{12}(e^{j\omega_1}, e^{j\omega_2}) \tag{3.31}$$

$$G_{21}(e^{j\omega_1}, e^{j\omega_2}) = -4H_{21}(e^{j\omega_1}, e^{j\omega_2}), \quad G_{22}(e^{j\omega_1}, e^{j\omega_2}) = 4H_{22}(e^{j\omega_1}, e^{j\omega_2}).$$

The gain of four is necessary since the interpolation by a factor of two in each dimension reduces the amplitude by a factor of four [14]. For nonseparable filters, the following condition is also needed to prevent aliasing

$$H_{11}(e^{j\omega_1}, e^{j\omega_2})H_{11}(-e^{j\omega_1}, -e^{j\omega_2}) = H_{11}(e^{j\omega_1}, -e^{j\omega_2})H_{11}(-e^{j\omega_1}, e^{j\omega_2}). \tag{3.32}$$

This result can be achieved if $H_{11}(e^{j\omega_1}, e^{j\omega_2})$ could be limited to the range $-\pi/2 \leq \omega_1 \leq \pi/2$ and $-\pi/2 \leq \omega_2 \leq \pi/2$. This simple method will not work in practice since non-ideal filters can not meet this condition without losing part of the signal. For a linear phase, symmetric filter $h_{11}(m,n)$, Eqs. (3.30), (3.31), and (3.32) require that

$$|H_{11}^2(e^{j\omega_1}, e^{j\omega_2})| + |H_{11}^2(e^{j\omega_1}, -e^{j\omega_2})| + |H_{11}^2(-e^{j\omega_1}, e^{j\omega_2})| + |H_{11}^2(-e^{j\omega_1}, -e^{j\omega_2})| = 1. \tag{3.33}$$

This condition prevents the limiting of the region of support of $H_{11}(e^{j\omega_1}, e^{j\omega_2})$ as discussed above unless an ideal filter can be used.

While one could design a nonseparable filter that approximately satisfies these constraints (and a hexagonal one has been implemented), the use of separable filters solves this problem much more easily. If $h_{11}(m,n) = h_1(m)h_1(n)$, then the aliasing terms will all cancel.

3.4 Diamond Subband Coder

3.4.1 Basic Concepts

Most of the previous work in image compression using subband coding has filtered the input image so that the frequency domain is partitioned into rectangular regions. While this subband coder based on separable quadrature mirror filters leads to an easy implementation, it does not take into account the sensitivity of the human eye at various angular orientations. For example, studies have shown that the eye has a higher sensitivity to horizontal or vertical edges than to edges at 45 and 135 degrees. To take advantage of this decreased sensitivity in order to decrease the transmission rate of the system, one could use non-separable diamond-shaped filters [4]. The low pass filter would have higher cut-off frequencies along both the horizontal and vertical directions than along the diagonals. The problem with this method is that it is not as easily implementable. The desire to have separable filters has led to a new method of diamond subband coding that will now be described. The concept of doing a diagonal interpolation was first introduced by Sullivan [88].

The original image, $x(m,n)$, of size $M \times M$ is interpolated by a factor of two along the diagonals. The four corners are filled with zeros at the higher sampling rate so that a block of size $2M \times 2M$ is formed. This image is called $\check{x}(m,n)$, and is now a square when viewed along the diagonals. Figure 3.5 shows this diagonal interpolation in the spatial domain for an 8×8 image. The interpolated image is then filtered into four subimages using the filters $H_{11}(e^{j\omega^T})$, $H_{12}(e^{j\omega^T})$, $H_{21}(e^{j\omega^T})$, and $H_{22}(e^{j\omega^T})$. The filtering is done in the normal way on $\check{x}(m,n)$. This corresponds to filtering along the r and c directions in Figure 3.5. The subimages are then decimated to yield $\check{y}_{11}(m,n)$, $\check{y}_{12}(m,n)$, $\check{y}_{21}(m,n)$, and $\check{y}_{22}(m,n)$. The centers of these subimages will

contain diagonally filtered and decimated versions of $x(m,n)$. The four corners of these images will be filled with zeros.

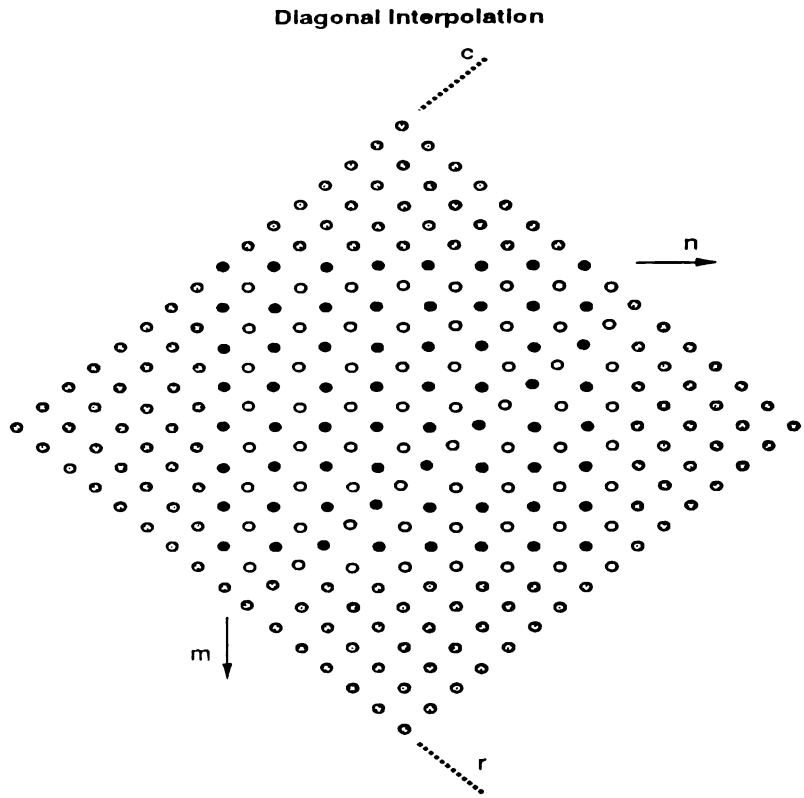


Figure 3.5: Spatial Domain View of the Diagonal Interpolation. m and n correspond to the rows and columns of the original image, and r and c correspond to the rows and columns of the diagonally interpolated image. The dark dots are the original pixels of an 8×8 image. The shaded dots are the zero pixels on the four corners, and the white dots are the zero pixels inside the image.

The desired result is that

$$\check{y}_{11}(m,n) = \check{y}_{22}(m,n) \quad \check{y}_{12}(m,n) = \check{y}_{21}(m,n). \tag{3.34}$$

The reason for this is that the initial interpolation doubled the number of useful pixels without adding any new information. The actual number of pixels has been quadrupled since the four corners are filled in with zeros. There is a factor of two

redundancy in each spatial dimension as will be shown in the next subsection. The centers of the subimages $\check{y}_{11}(m,n)$ and $\check{y}_{12}(m,n)$ can be coded and transmitted. The four corners are all zeros so they do not have to be sent. In this way, the analysis system takes a single image of size $M \times M$ and creates two subimages with $M^2/2$ pixels. Therefore, the total number of pixels remains constant. The advantage of this system is that the low pass subimage is diamond-shaped.

At the receiver in this two subband system, the two subimage centers are decoded and the corners are filled in with zeroes. They are then interpolated and filtered to yield $\check{x}_{11}(m,n)$ and $\check{x}_{12}(m,n)$. These images are both of size $2M \times 2M$ with the four corners still all zeros. The final step is to decimate along the diagonals including throwing away the four corners to yield the subimages $x_{11}(m,n)$ and $x_{12}(m,n)$, each of size $M \times M$. These two images are combined to reconstruct $x(m,n)$.

This system can be directly implemented if the analysis filters are separable and created from the two tap QMF filters

$$H_1(z) = 1 + z^{-1} \quad H_2(z) = 1 - z^{-1}. \tag{3.35}$$

When using longer QMF filters, problems arise because of boundary effects. There will be ringing caused by a discontinuity at the transition between the diagonally interpolated image and the four corners that have been filled in with zeros. This ringing is much more serious than the corresponding ringing in the rectangular configuration, and failure to reduce it had prevented this system from being implemented. Fortunately, the ringing can be essentially eliminated by proper boundary extension, as will be discussed in the section on implementation.

3.4.2 Diagonal Interpolation

The Nyquist region of a digital image extends from $-\pi$ to π in both the horizontal and vertical directions. This is shown by the square box in Figure 3.6 where the horizontal and vertical frequencies are denoted by ω_1 and ω_2, respectively. The diagonal interpolation will yield an image with the same Nyquist region of $-\pi$ to π, but this will now be in terms of the new frequency axes ω_1' and ω_2'. In order to determine the mapping of the image's Fourier transform from (ω_1, ω_2) to (ω_1', ω_2'), it is convenient to use the vector notation defined by Dudgeon and Mersereau in [15]. In this notation, the column vectors $\mathbf{n} = (n_1\ n_2)^T$, $\mathbf{u} = (u_1\ u_2)^T$, $\omega = (\omega_1\ \omega_2)^T$, and $\omega' = (\omega_1'\ \omega_2')^T$.

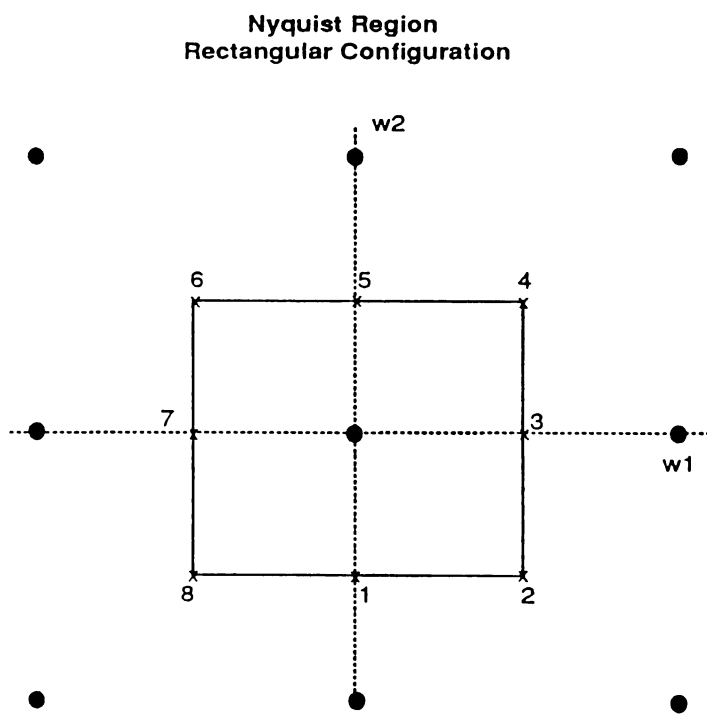


Figure 3.6: Nyquist Region of Digital Image. The square box corresponds to the range $-\pi \leq \omega_1 \leq \pi$ and $-\pi \leq \omega_2 \leq \pi$. The dots show the centers of the original image and the aliased images.

The image, $x(\mathbf{n})$, with Fourier transform

$$X(e^{j\omega^T}) = \sum_{\mathbf{n}\in\Omega} x(\mathbf{n}) e^{-j\omega^T \mathbf{n}} \tag{3.36}$$

is resampled using the sampling matrix

$$\mathbf{V} = \begin{bmatrix} 0.5 & -0.5 \\ 0.5 & 0.5 \end{bmatrix}, \tag{3.37}$$

where the summation is over the two-dimensional integer lattice, Ω. The interpolated image is $w(\mathbf{u})$, and it is given by

$$w(\mathbf{u}) = \begin{cases} x(\mathbf{Vu}) & \text{if } u_1, u_2 \text{ even} \\ 0 & \text{otherwise.} \end{cases} \tag{3.38}$$

Figure 3.7 shows this resampling operation in the spatial domain.

The Fourier transform of $w(\mathbf{u})$ is given by

$$W(e^{j\omega'^T}) = \sum_{\mathbf{u}\in\Omega} x(\mathbf{Vu}) e^{-j\omega'^T \mathbf{u}}. \tag{3.39}$$

To get this in terms of the Fourier transform of $x(\mathbf{n})$, let $\mathbf{n} = \mathbf{Vu}$. This yields

$$W(e^{j\omega'^T}) = \sum_{\mathbf{n}\in\Omega} x(\mathbf{n}) e^{-j\omega'^T \mathbf{V}^{-1}\mathbf{n}}. \tag{3.40}$$

The final result is

$$W(e^{j\omega'^T}) = X(e^{j\omega'^T \mathbf{V}^{-1}}) \tag{3.41}$$

where the inverse of the sampling matrix is given by

$$\mathbf{V}^{-1} = \begin{bmatrix} 1 & 1 \\ -1 & 1 \end{bmatrix}. \tag{3.42}$$

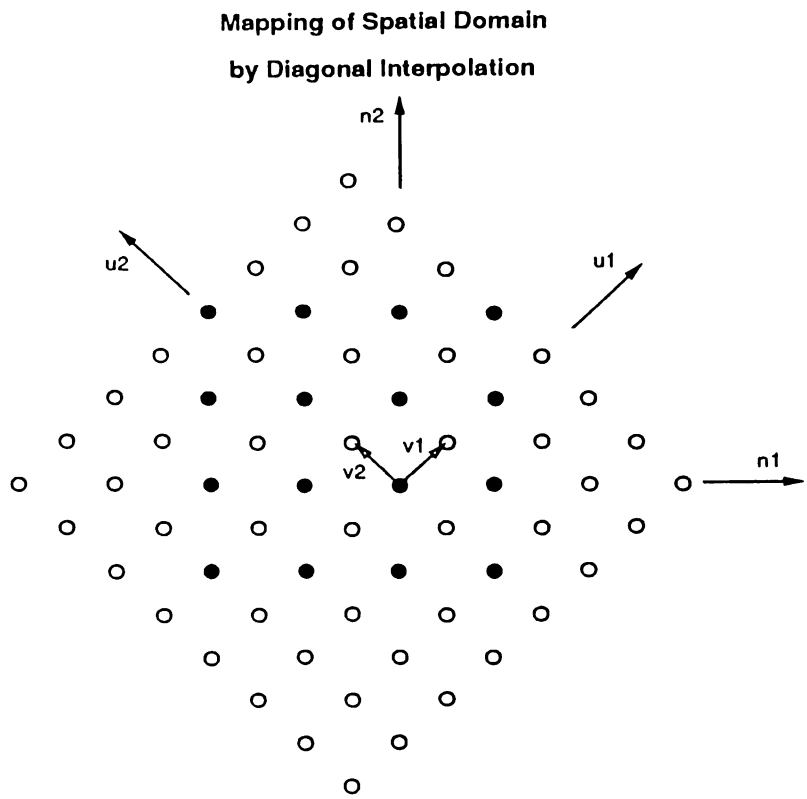


Figure 3.7: The Mapping of the Coordinate Axes in the Spatial Domain by the Diagonal Interpolation. The vectors $\mathbf{v}_1$ and $\mathbf{v}_2$ are the sampling vectors. They become columns of the sampling matrix, V. The coordinate axes of the original image are n_1 and n_2, and the coordinate axes of the interpolated image are u_1 and u_2. The dark dots are the original pixels of a 4×4 image. The white dots are zero pixels.

In Figure 3.8, one can see the results of the diagonal interpolation in the frequency domain. The box made from dashed lines shows the new Nyquist region. Notice that the rectangular Nyquist region of Figure 3.6 is mapped into a diamond in this new frequency domain, as shown by the numbers in the two figures.

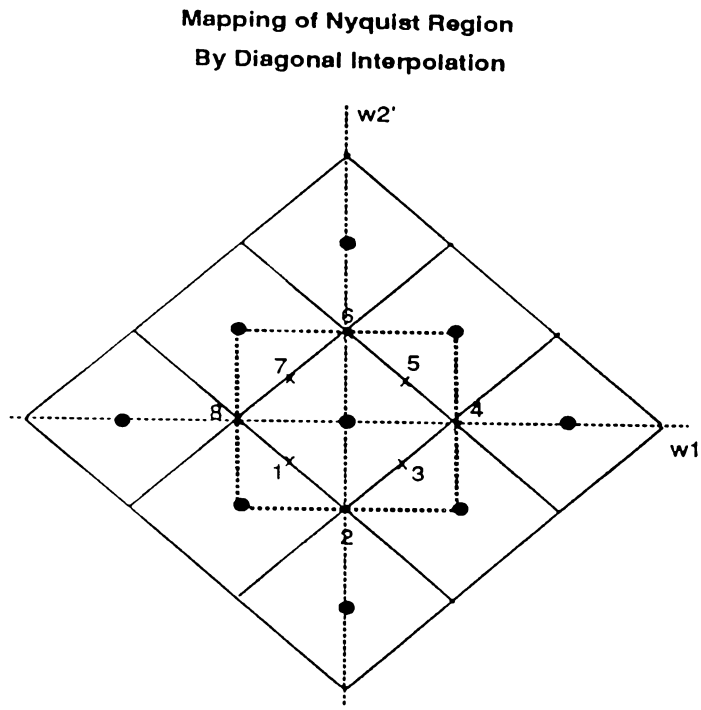


Figure 3.8: Frequency Domain after Diagonal Interpolation. The dashed square box corresponds to the Nyquist region of $-\pi \leq \omega_1 \leq \pi$ and $-\pi \leq \omega_2 \leq \pi$. The dots show the centers of the original image and the aliased images.

3.5 System Implementation

3.5.1 Rectangular Subband Configuration

A seven subband system was implemented using the 32 tap filter known as 32D in [41]. The input image was filtered into four subbands where one subband was a lowpass filtered version of the input, and the other three were bandpass filtered versions. The lowpass subband was then filtered into four more subbands using a 16 tap filter to yield the total of seven. The resulting partition of the two-dimensional frequency domain is shown in Figure 3.9. All filtering was done in the frequency domain using fast Fourier transforms. The monochrome subband coder was extended to color images by creating three such systems in parallel. Each system processed one

of the three color components. This system is flexible enough to allow processing in alternative color spaces since the filter structure remains constant. The only change between color spaces is in the set of quantizers used.

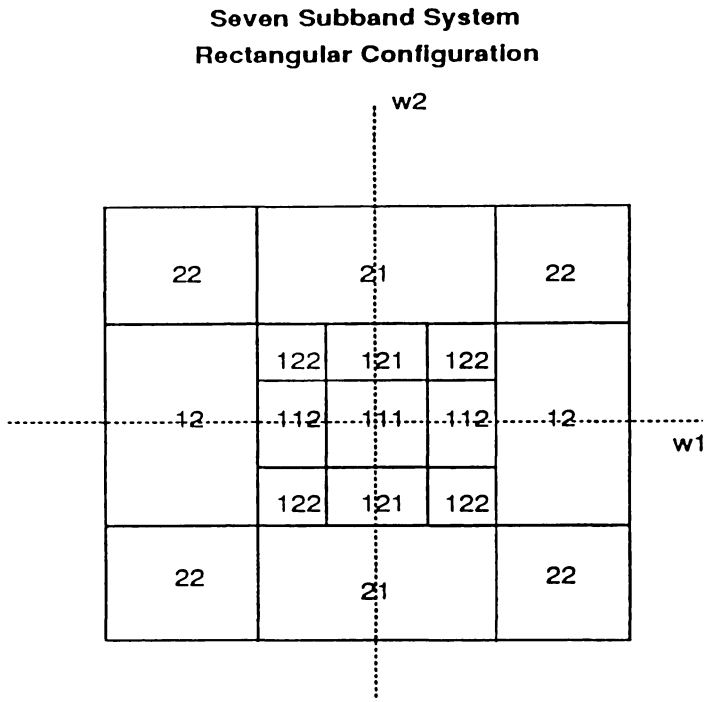


Figure 3.9: Frequency Domain Partition for the Seven-Band Decomposition.

The subband simulation was modified so that the filtering is now done in the time domain with constant extension at the boundaries to minimize edge effects. All filtering is done with the 32 tap filter 32D. The change to a time domain implementation reduces the filterbank distortion slightly, but more importantly speeds up the simulation run time. Figures 3.10 and 3.11 show the block diagrams of the analysis and synthesis filterbanks for each color component.

As discussed in Chapter 2, the input images are stored as C.I.E. XYZ tristimulus values. They are transformed to the desired color space before the transmitter

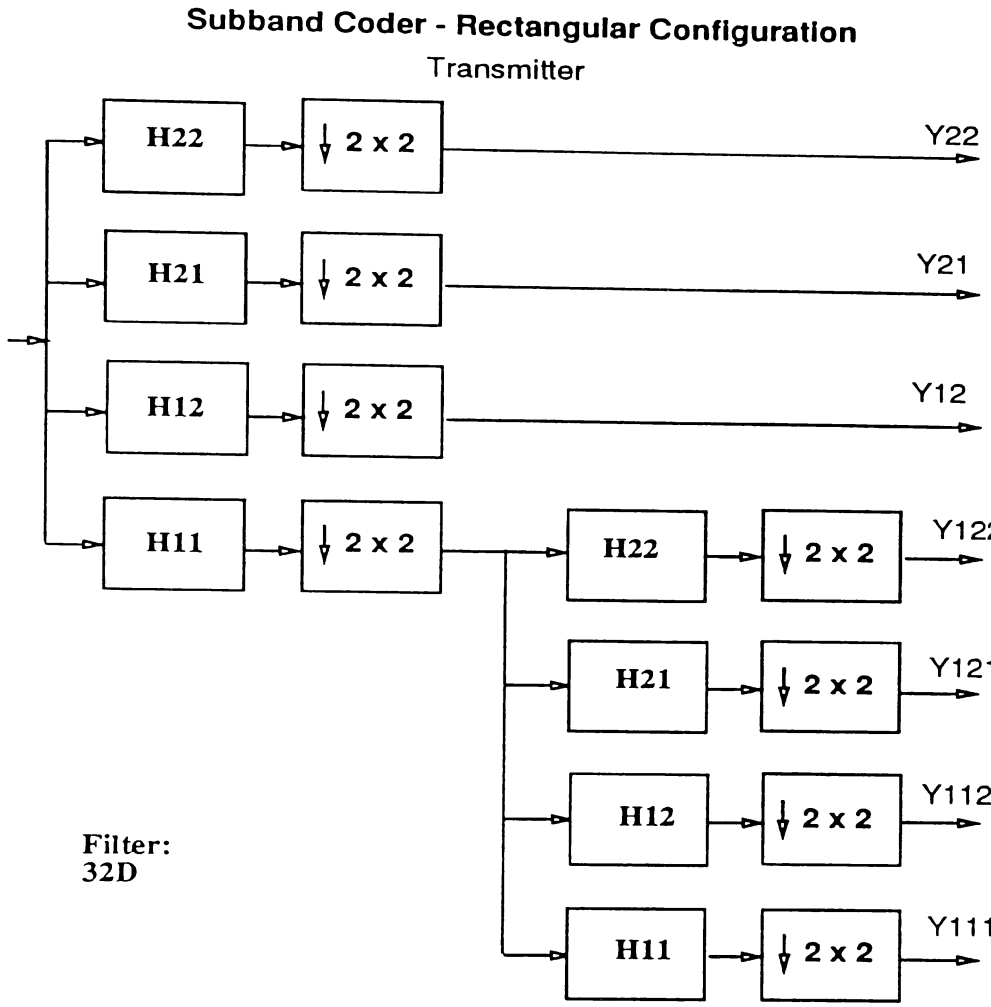


Figure 3.10: Subband Analysis Filterbank. Seven-Band Rectangular Configuration.

filterbank and transformed back to XYZ space after the receiver filterbank. All subband simulations in the rest of this work are run in this time domain configuration. Table 3.1 contains the ΔE distortion caused by the subband filtering in the three different color spaces. Notice that in the absence of coding, all three color spaces yield very good results. As we shall see later, the distortion caused by the quantization is ten to thirty times greater than this. However, a very close inspection of reconstructed images without any coding will show some slightly visible artifacts along the

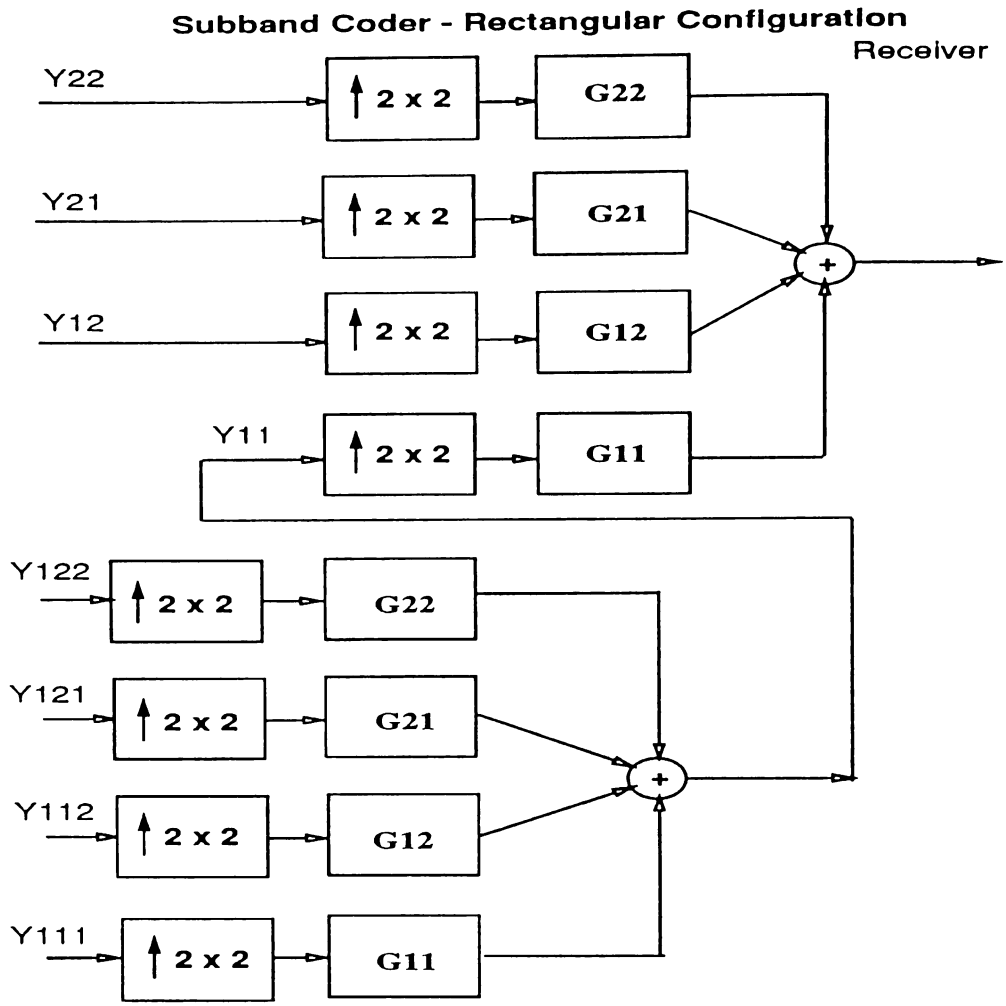


Figure 3.11: Subband Synthesis Filterbank. Seven-Band Rectangular Configuration.

borders of each image.

3.5.2 Diamond Subband Configuration

A five band system was implemented using the 32 tap quadrature mirror filter known as 32D [41]. Constant extension was first used on the image in both the vertical and horizontal directions. The extensions on each side had the length of 1/2 of the filter's length. This larger image was then diagonally interpolated. For the 32 tap case with

Color Space	Image	Δ E (ave)	$\sigma_{\Delta E}$
YIQ	GIRL	0.203	0.947
	DOLL	0.269	1.250
$L^*a^*b^*$	GIRL	0.187	0.930
	DOLL	0.246	1.235
AC_1C_2	GIRL	0.198	0.940
	DOLL	0.256	1.241

Table 3.1: Filterbank Distortion - No Coding. Rectangular Configuration.

a 256 × 256 image, the size of the extended image was 288 × 288, and the size of the interpolated image was of size 576 × 576. The center image of size 512 × 512 was then filtered and decimated. The above process is equivalent to doing constant extension along the diagonal boundaries between the image and the four corners filled with zeros. The latter process could be implemented in a real-time system since its computational complexity is significantly lower.

The subimages $\check{y}_{11}$ and $\check{y}_{12}$ were stored and the subimages $\check{y}_{21}$ and $\check{y}_{22}$ were discarded. The system filters the lowpass subimage, $\check{y}_{11}$, into four subimages in the conventional rectangular way. These subimages are decimated, and this yields a five band system with the frequency partition shown in Figure 3.12. Figures 3.13 and 3.14 show the block diagrams of the analysis and synthesis filterbanks for each color component. Only the valid pixels are transmitted. That is, the padded pixels and the zeros on the corners are not. In this way the total number of pixels remains constant. The receiver first does constant extension along the diagonals. It then interpolates and filters the subimages. These subimages are added together, and the reconstructed image is diagonally decimated to yield the final reconstructed image.

The ΔE distortion caused by the subband filtering is shown in Table 3.2 for the diamond configuration. The numbers are similar to the rectangular configuration.

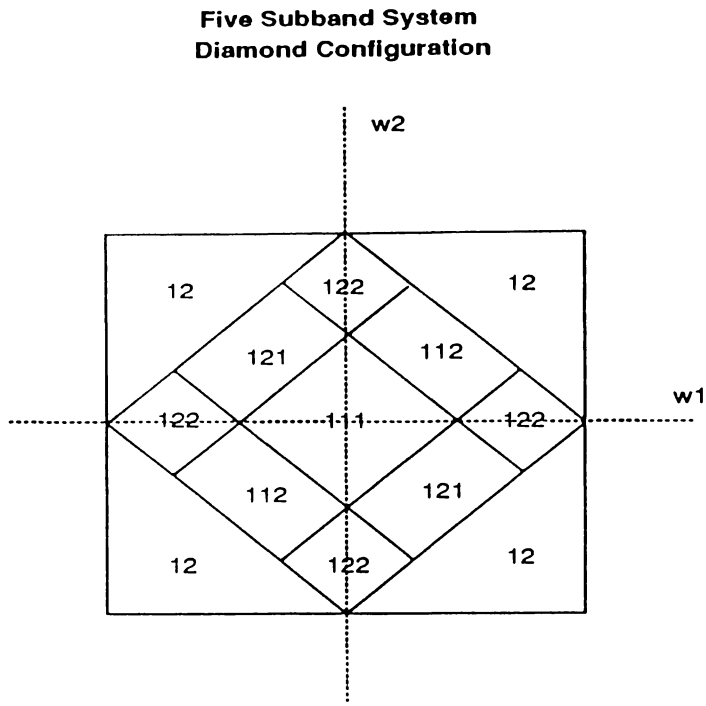


Figure 3.12: Frequency Domain Partition for the Five-Band Decomposition.

However, the diagonal interpolation makes the boundary extension more difficult and important because the discontinuity in the horizontal direction coincides with the one in the vertical direction. There is some visible distortion along the borders of the images, but they can be seen only by close inspection.

Color Space	Image	Δ E (ave)	$\sigma_{\Delta E}$
YIQ	GIRL	0.207	0.991
	DOLL	0.228	1.148
$L^*a^*b^*$	GIRL	0.200	0.963
	DOLL	0.210	1.008
AC_1C_2	GIRL	0.212	0.978
	DOLL	0.219	1.024

Table 3.2: Filterbank Distortion - No Coding. Diamond Configuration.

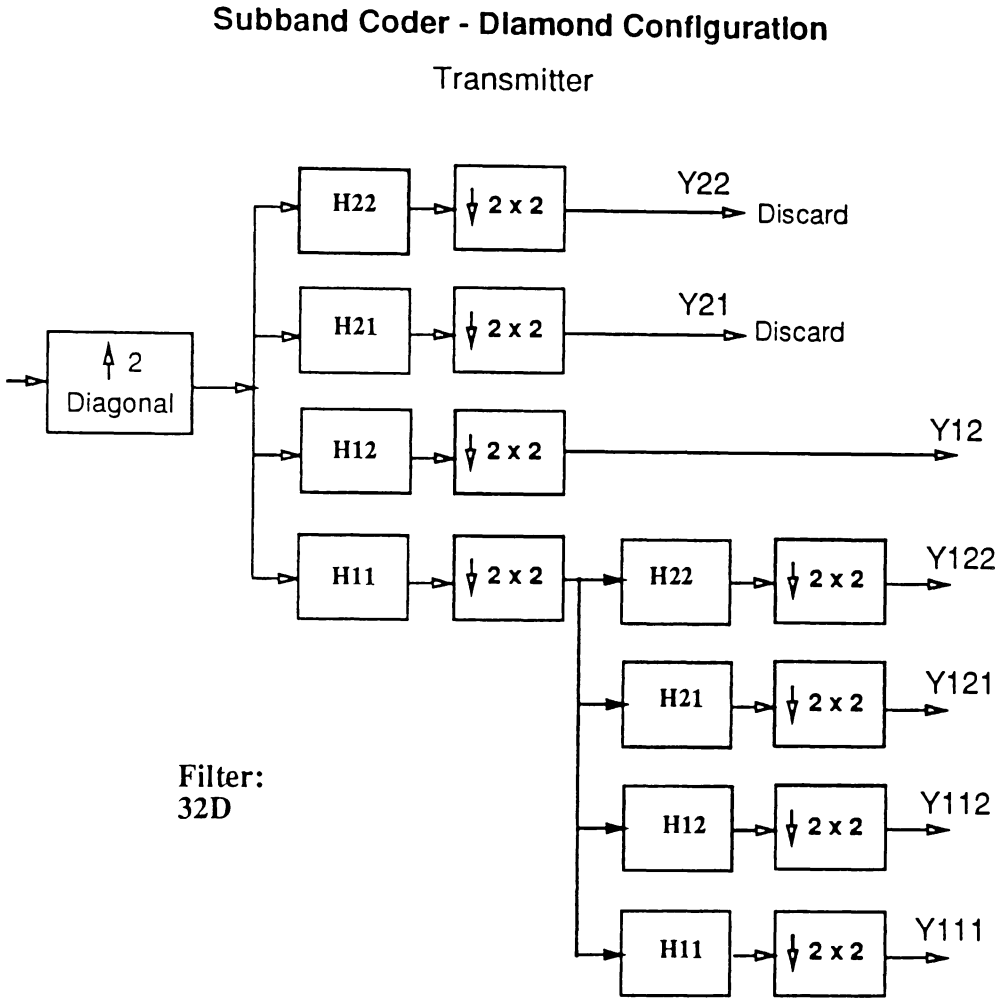


Figure 3.13: Subband Analysis Filterbank. Five-Band Diamond Configuration.

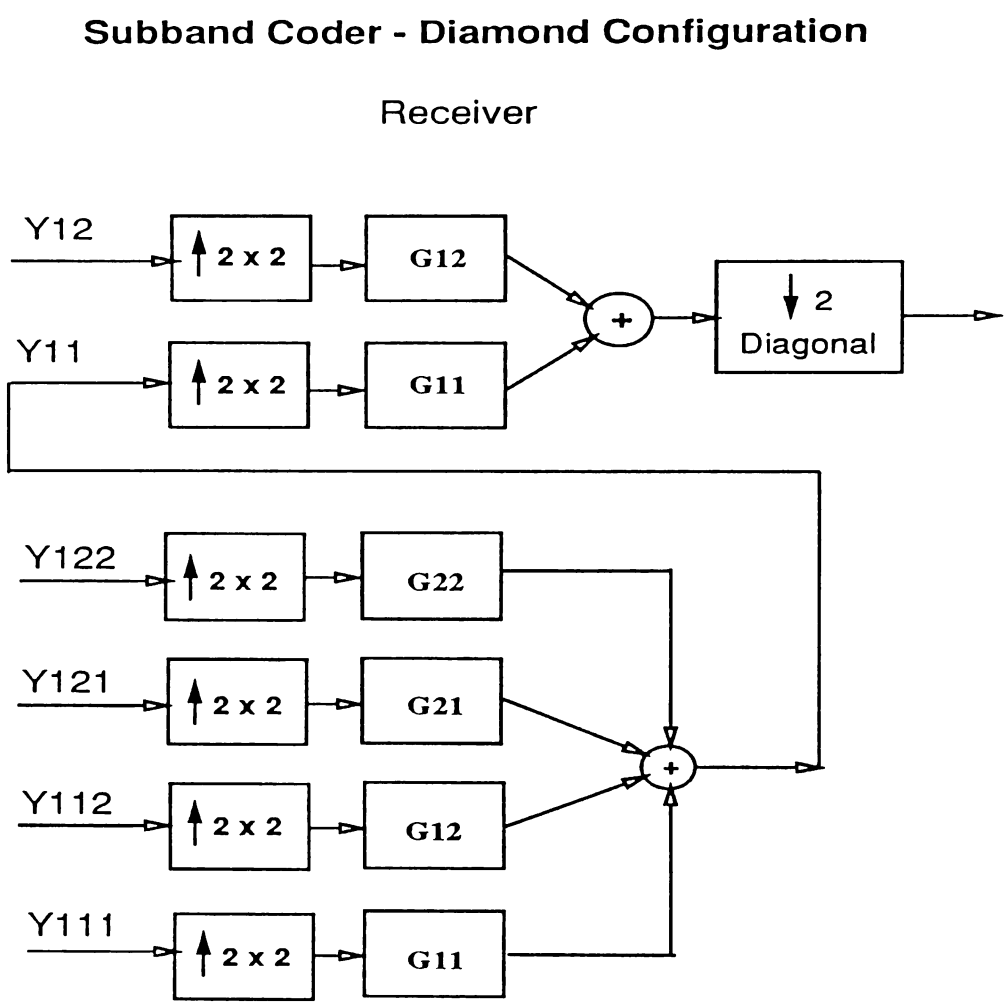


Figure 3.14: Subband Synthesis Filterbank. Five-Band Diamond Configuration.

CHAPTER 4

FULLBAND AND SUBBAND CODING WITH SCALAR QUANTIZATION

4.1 Introduction

As discussed in Chapter 2, each pixel of a color image is represented by three different color components; these components define a three-dimensional color space. Because there are three components instead of one component as in monochrome images, more degrees of freedom are introduced into the source coding problem. Linear or nonlinear invertible transformations can be used to map one color space to another. The actual compression is then achieved by quantization in some desired color space. For example, the color spaces that are considered perceptually uniform are nonlinear transformations of XYZ space. Quantization is itself a nonlinear process, so the choice of color space can have important and unpredictable results on the output image quality. In this chapter, scalar quantization in alternative color spaces is examined to determine and quantify these results. The knowledge gained will then be used in later chapters to design a color compression system.

Saying that a color image is scalar quantized means that each of the three color components are quantized separately on a pixel by pixel basis. Initially, fullband images are studied to determine what effect the choice of color space has on the amount and types of color errors in the reconstructed image. These results depend heavily on the quality of the quantizer used, so an optimal scalar quantizer is desired.

The design of such a quantizer requires that the probability density function of each component of the image be known, and that a set of nonlinear equations be solved.

Panter and Dite [65] derived an approximation to the optimal scalar quantizer that is valid when the number of quantization levels is large. Since the initial compression ratio of the simulations is only 2:1, this approximation is valid and their method can be used. We derive an iterative version of their algorithm, and this algorithm is used to design the scalar quantizers for the different color components. This algorithm was derived because the resulting distortion is less than that obtained using Panter and Dite's approximation; in some cases the resulting mean squared error is reduced by thirty-five percent. This new iterative method requires less computation than the Lloyd algorithm, although this is not a very important consideration for the scalar case. Quantizers are designed for C.I.E. XYZ, N.T.S.C. RGB, N.T.S.C. YIQ, and C.I.E. $L^*a^*b^*$ color spaces, and two test images are coded in each of these spaces.

To obtain further useful results at higher compression ratios, a subband coding system with scalar quantization is implemented. Here, each color component of the subbands is scalar quantized on a pixel by pixel basis. For a given compression ratio, a method is now required to determine how to allocate the bits among the different subband color components. This problem is formulated as an optimization problem where the objective function depends on the variances of the subband color components. The assumption is made that the bit allocations are continuous functions, and the problem is solved using the standard Lagrange multiplier method. The bit allocations are then rounded to the nearest integer. The number of bits allocated to a particular subband color component may be small. This violates the main assumption in Panter and Dite's derivation. To design the scalar quantizers for the subband case, the Lloyd algorithm [51] is used.

The format of the chapter is as follows. The next section derives the iterative version of the Panter and Dite scalar quantizer. The third section presents simulation results for images that have been scalar quantized in the different color spaces. Tables of the ΔE distortion are given, and photographs of some of the reconstructed images are included. The fourth section derives the variance-based bit allocation. The last section gives results of subband coding with optimal scalar quantization. The extension to vector quantization will be made in the following chapters.

4.2 Derivation of Scalar Quantizer

A method is needed to design quantizers for each color component of an image. Max showed [58] that to find the optimal quantizer for the one-dimensional signal given by the probability density function, $p_y(y)$, one had to solve the nonlinear equations

$$\begin{aligned} r_k &= \frac{\int_{t_k}^{t_{k+1}} u p_u(u)\, du}{\int_{t_k}^{t_{k+1}} p_u(u)\, du} \\ t_k &= \frac{r_k + r_{k-1}}{2}. \end{aligned} \tag{4.1}$$

In these equations, the $t_k's$ are the decision levels and the $r_k's$ are the reconstruction levels. His derivation uses minimum mean squared error as the distortion criterion. This optimal quantizer is also known as the Lloyd-Max quantizer.

Panter and Dite [65] derived a nearly optimal quantizer by assuming that the number of quantizer levels is large enough that the probability density function is constant for the region between two consecutive decision levels. This assumption leads to the following equations

$$t_{k+1} = A \frac{\int_{t_1}^{z_k + t_1} [p_u(u)]^{-1/3}\, du}{\int_{t_1}^{t_{L+1}} [p_u(u)]^{-1/3}\, du} + t_1 \tag{4.2}$$

and

$$r_k = \frac{t_{k+1} + t_k}{2} \tag{4.3}$$

where $A = t_{L+1} - t_1$, $z_k = kA/L$, L is the number of levels, and $k = 1 \ldots L$.

Eq. (4.2) is formed by taking the continuous approximation to the series

$$t_k = \frac{A}{K}[\frac{2}{p^{1/3}(r_1)} + \frac{2}{p^{1/3}(r_2)} + \cdots + \frac{2}{p^{1/3}(r_{k-1})}] + t_1 \tag{4.4}$$

where K is the normalization constant

$$K = \frac{2}{p^{1/3}(r_1)} + \frac{2}{p^{1/3}(r_2)} + \cdots + \frac{2}{p^{1/3}(r_L)}. \tag{4.5}$$

Therefore, Eq. (4.4) can be used instead of Eq. (4.2) to directly calculate the decision levels.

To obtain better performance, we modified the method of Panter and Dite by making it iterative [94]. A histogram was used as an approximation of the probability density function. The bin size was selected so that there were roughly 100 bins containing nonzero samples out of a total of 128 bins. This number was selected because it seemed to be more than sufficient. For example, the range of values of the luminance component in $L^*a^*b^*$ space was about zero to 106. If the maximum error is one-half of a bin, then the maximum ΔE error resulting from the use of the histogram is less than one. The minimum and maximum values for the quantizer range were selected. They were chosen to be the points where the tails of the distribution went to zero. The algorithm is as follows:

0) Chose a maximum number of iterations, Max. Create a uniform quantizer with the lowest decision level at the lowest bound and the highest decision level at the highest bound. Calculate the reconstruction levels using Eq. (4.3). For the given

probability density function (histogram), calculate the mean squared error of the uniform quantizer.

1) Average the histogram between each two adjacent decision levels, and use these values for the reconstruction levels r_1 through r_L.

2) Calculate the new decision levels using Eq. (4.4).

3) Calculate the reconstruction levels using Eq. (4.3).

4) Calculate the mean squared error of the quantizer.

5) If the number of iterations is less than Max, go to step 1).

In step 1) of the algorithm, the average of the histogram bins between the two decision levels is used. This averaging acts as a smoothing function and helps to induce convergence. Even with this modification, the algorithm is not absolutely stable. Using the midpoint value, as Panter and Dite originally did, leads to less stability in the algorithm. Since they only computed one iteration, the question of stability was not important.

The algorithm is run for 100 iterations. Because of the discrete nature of the histogram, the levels will usually oscillate between two sets. This generally occurs after less than twenty iterations so the limit of 100 iterations is very conservative. The iteration with the lowest mean squared error is chosen; this is why the mean squared error is calculated in step 4). The quantizer yielding the lowest mean squared error is not necessarily one of the last two, however this is sometimes the case. Often, after only two or three iterations the best result is obtained. The final values are slightly higher. While there is no guarantee that this method yields a global minimum, for smooth distributions the resulting distortion is lower than that of both the uniform quantizer and the quantizer designed using a single iteration (Panter and

Dite's method).

4.3 Quantization Results

The quantization experiments have three goals. The first one is to determine the types of color artifacts that occur in the different color spaces. To do this, each color component was allocated the same number of bits. Different compression ratios were chosen, and the locations and severities of the color errors were noted. The second goal is to compare the subjective ranking of the images with the calculated distortion numbers to obtain an initial assessment of the distortion measure's validity. It will be seen that although the average ΔE error is relatively convenient to use, this is not a particularly accurate indication of image quality for images that contain areas of large color errors. The ΔE error for each pixel is a more accurate indication, but it is not a single distortion measure. For this reason, the average ΔE error is used through out the rest of this work.

The third goal is to determine the relative importance of the three color components of $L^*a^*b^*$ space and to compare this with YIQ space. Both of these color spaces contain a luminance and two chrominance components, and are important because their use in color systems can allow compatibility with monochrome systems. It is well-known in conventional color television that the human visual system is less sensitive to the I and Q components than to the Y component. $L^*a^*b^*$ space is considered to be perceptually uniform in that the three color components are supposedly equally important. From a compression standpoint, it is important to determine how accurate this is.

One test consisted of quantizing the GIRL image in the four color spaces using 4

bits/pixel per component. This is only a compression ratio of 2:1, but it still shows differences among the color spaces. The ΔE value is the lowest for $L^*a^*b^*$ space. This is not surprising since, in this case, the quantizer design was run in the same color space that was used in computing the distortion measure. The question is whether the visual appearance matches these average numbers. As was shown in Figure 2.5 in Chapter 2, the MacAdam ellipses are not circles in $L^*a^*b^*$ space. Additionally, the calculation of ΔE is based on a comparison of fixed colors; it does not take into account spatial or frequency variations. A better distortion measure would use a more uniform color space and take into account the visibility of artifacts. The color space used in the distortion measure could use a transformation that is too complicated for real-time compression systems since the calculation of the distortion measure can be done off line.

Table 4.1 contains the ΔE distortion errors for scalar quantization in the four color spaces for a few different bit allocations. The XYZ space images have the highest ΔE errors. To see how well these numbers correlate with a subjective evaluation, we compare the coded images to the originals. The original GIRL and DOLL images are in Figure 2.10 in Chapter 2, while Figure 4.1 shows the GIRL image after coding in XYZ and RGB spaces with four bits/pixel per color component. Examining the XYZ image, one sees that these distortion numbers are justified. Parts of this image are not too bad, but the region on the girl's forehead has clearly visible color errors. The skin here appears to be green and red. The color errors in this region are high enough to raise the average and make the image visually unacceptable. The RGB image also shows similar color errors on the forehead, yet this space still appears to be better than XYZ space.

Figure 4.2 shows the images coded in YIQ and $L^*a^*b^*$ spaces with the same

bit allocation of four bits/pixel per color component. The YIQ image has some contouring on the girl's face, but smaller color errors. These artifacts are similar to those in the image coded in $L^*a^*b^*$ space. Although the average color errors are fairly small, this last image is still somewhat objectionable since the artifacts are easily noticed.

Color Space	bits/pix	Δ E (ave)
XYZ	5,5,5	8.260
	4,4,4	12.211
RGB	5,5,5	5.178
	4,4,4	9.653
YIQ	5,4,4	7.583
	4,4,4	7.762
$L^*a^*b^*$	5,4,4	1.801
	4,4,4	2.340
	5,3,3	3.517
	4,3,3	3.880

Table 4.1: Distortion Measure Results - GIRL Image.

The next set of tests consider only $L^*a^*b^*$ space. This color space was chosen since it is a perceptually uniform space. Also, since it consists of a luminance and two chrominance components, as does YIQ space, it can make use of the mean detection threshold data measured by Krishnakumar [46]; this will be discussed in later chapters. Three additional simulations were run. Table 4.1 shows how the bits were allocated to each color component. While the 4,4,4 case has a lower average distortion, the 5,3,3 case visually appears to be much better according to a panel of observers. Figure 4.3 shows that by allocating 5 bits/pixel to the luminance component, the contouring on the girl's face is practically eliminated. The chrominance errors are barely visible, even in the highly saturated yellow area. When subband

coding is evaluated, it will be seen that the number of bits/pixel allocated to the L^* component of the lowest frequency subband has a very significant effect on the reconstructed image quality. This experiment shows that 5 bits/pixel is a minimum requirement for $L^*a^*b^*$ space.

The GIRL images coded with the bit allocations of 4,4,4 and 4,3,3 do not look much different. They both show serious contouring and smaller color errors. This helps confirm that the three color components are not equally important; the L^* component dominates. The DOLL image contains many saturated colors and suffers from a higher distortion number than the GIRL image for the 5,3,3 bit allocation. However, the reconstructed image does not appear too desaturated. Since human observers are very sensitive to variations in skin color, a practical system must provide natural looking flesh tones. By allocating more bits to the luminance component, this is done and saturated images still look acceptable. Therefore, this bit allocation is superior.

The quantization experiments have shown that compression in XYZ and RGB space tends to cause green and red color errors if insufficient bits are allocated to the quantizers. This is true when directly examining the images on the monitor in a darkened room, and even when examining the photographs under cool fluorescent light. XYZ space suffers the most from these degradations and is, therefore, not a good color space to use for image compression. The errors in YIQ and $L^*a^*b^*$ spaces are similar. For an equal bit allocation, the luminance component will not have enough bits, and contouring will appear in areas of neutral color. This artifact can be removed by providing extra bits to the luminance component at the expense of the two chrominance components.

The spatial variation of the ΔE errors more closely matches the visual appear-

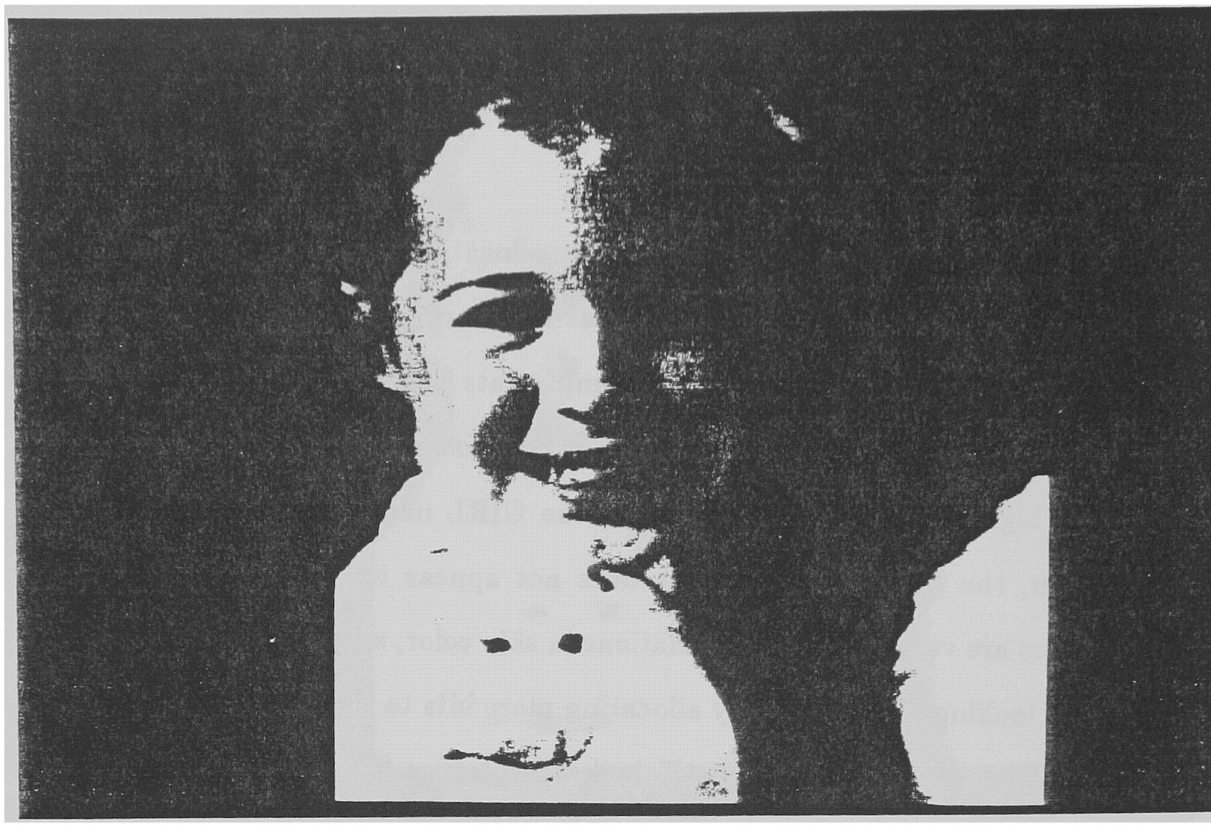


Figure 4.3: GIRL Image Coded in $L^*a^*b^*$ Space. 5 bits/pixel for L^*, 3 bits/pixel for a^*, and 3 bits/pixel for b^*.

ance of the reconstructed images than does the average ΔE error. For the scalar quantization of fullband images, the average ΔE error is not a very reliable measure of image quality. This is due to areas of large color errors and to the easily noticeable contouring. These artifacts may not be large enough to raise the average error, but they are high enough in localized areas to make the images objectionable. Subband coding provides better image quality at higher compression ratios. At moderate compression ratios of 8:1 through 16:1, the artifacts are much smaller and the distortion numbers are lower than in fullband coding. Here, it will be shown that the average ΔE error is a more useful indication of image quality for a given color space.

ance of the reconstructed images than does the average ΔE error. For the scalar quantization of fullband images, the average ΔE error is not a very reliable measure of image quality. This is due to areas of large color errors and to the easily noticeable contouring. These artifacts may not be large enough to raise the average error, but

4.4 Variance-Based Bit Allocation

The results of the last section show that there are noticeable color errors for compression ratios as low as 2:1 for coding done by fullband scalar quantization. The next step was to combine subband coding with scalar quantization. The seven subband system configuration described in Chapter 3 was used. Since there are three color components for each subband, there are a total of 21 subband components. A method was needed to determine how many bits to allocate to each subband color component.

The method derived by Panter and Dite is an approximation of the Lloyd-Max quantizer. If the number of bits allocated to a particular subband color component were small, the main assumption of Panter and Dite's derivation would no longer hold. To allow for this possibility, the derivation of the variance-based bit allocation will use the distortion model of the actual Lloyd-Max quantizer. This quantizer could have also been used for the fullband quantization of the color images. One can model the distortion, $D(b)$, of the Lloyd-Max quantizer as a function of the bit rate using the equation $D(b) = \sigma^2 K(b) 2^{-2b}$, where $K(b)$ is a slowly varying function of the bit rate and σ^2 is the variance [37]. For the case where the probability density function is Gaussian, one can plot this function using the results in [51, 58].

Since the $K(b)$ term does not change much with bit rate and its effect is less important than the exponential term, the assumption is made that it is a constant. Denote this constant by K. For the seven subband simulation with optimal scalar quantizers, the bit allocation problem becomes

$$min_{\mathbf{b}\in S} D(\mathbf{b}) = \sum_{k=1}^{4}\sum_{c=1}^{3} K\sigma_{k,c}^2 2^{-2b_{k,c}} + \sum_{m=5}^{7}\sum_{c=1}^{3} K\sigma_{m,c}^2 2^{-2b_{m,c}} \tag{4.6}$$

subject to

$$B(\mathbf{b}) = \frac{1}{16}\sum_{k=1}^{4}\sum_{c=1}^{3} b_{k,c} + \frac{1}{4}\sum_{m=5}^{7}\sum_{c=1}^{3} b_{m,c} = \beta. \tag{4.7}$$

$\mathbf{b}$ is the 21-dimensional vector giving the bit allocation for all 21 of the subband components, S is the set of all possible bit allocations among the subbands, D is the total distortion, β is the total bit rate, and $\sigma_{k,c}^2$ is the variance of the k, c component.

This bit allocation problem is a straight-forward extension to color images of the optimization problem solved by Woods and O'Neil [104]. Since they used DPCM coding, their model for the quantizer used the variance of the prediction error. Ma [52] formulates a similar optimization problem using the energy of each subband instead of the variance, and provides simulation results that show that this results in a subjectively better bit allocation. For the equally sized subband case that he considers, the difference between the two solutions is only in the use of signal variance versus signal energy. Ma then provides a sequential algorithm that implements an approximation to his analytical solution using an approach similar to [75].

The optimization problem described by Eqs. (4.6) and (4.7) is solved using the standard Lagrange multiplier method [78]. It should be noted that the resulting bit allocation is not constrained to yield non-negative integers for each subband component. In Chapter 6, a discrete Lagrange multiplier formulation is introduced that adds this constraint; the solution of this problem also allows the use of perceptual weights based on the mean detection threshold of the human visual system.

Define the Lagrangian function, $L(\mathbf{b})$, by

$$L(\mathbf{b}) = D(\mathbf{b}) + \lambda(\beta - B(\mathbf{b})), \tag{4.8}$$

where λ is the Lagrange multiplier. Differentiating gives

$$\nabla L(\mathbf{b}) = \mathbf{g}(\mathbf{b}) - \lambda \mathbf{A} \tag{4.9}$$

where $\mathbf{g}(\mathbf{b})$ is the gradient of the distortion function and $\mathbf{A}$ is the gradient of the constraint function, $B(\mathbf{b})$.

Since the constraint function is formulated as an equality, the first-order necessary condition is that

$$\nabla L(\mathbf{b}^*) = \mathbf{0} \tag{4.10}$$

where $\mathbf{b}^*$ is a stationary point of both the unconstrained problem given by Eq. (4.8) and of the constrained problem given by Eqs. (4.6) and (4.7) [78, pages 143-144]. Writing out Eq. (4.10) gives the following:

$$\nabla L(\mathbf{b}) = -2(\ln 2)K \begin{bmatrix} \sigma_{1,1}^2 2^{-2b_{1,1}} \\ \vdots \\ \sigma_{4,3}^2 2^{-2b_{4,3}} \\ \sigma_{5,1}^2 2^{-2b_{5,1}} \\ \vdots \\ \sigma_{7,3}^2 2^{-2b_{7,3}} \end{bmatrix} - \lambda \begin{bmatrix} 1/16 \\ \vdots \\ 1/16 \\ 1/4 \\ \vdots \\ 1/4 \end{bmatrix} = \mathbf{0}. \tag{4.11}$$

Eq. (4.11) contains twenty-one equations and twenty-two unknowns. Because there are only two different sizes of subbands, one can use symmetry to eliminate λ and reduce this system of equations down to three equations. The size of the subband is determined by the number of pixels in each subband. The first equation relates the bit rates of the components of the smaller subbands to each other, and the second relates the bit rates of the components of the larger subbands to the bit rates of the components of the smaller subbands. The third equation relates the bit rates of the components of the larger subbands to each other. Let the subscripts k and j refer

based bit allocation used the distortion model of the Lloyd-Max (optimal) quantizer, the quantizers were required to be of this type. They were designed using the Lloyd algorithm [51]. This algorithm is a one-dimensional version of the LBG algorithm [49] that will be used to design the vector quantizers. The algorithm requires a training sequence as input; the GIRL image was used as the training sequence. The GIRL and DOLL images were compressed and the distortion numbers for the reconstructed images are given in Table 4.4.

Color Space	$L^*a^*b^*$	YIQ	RGB
Subband	bits/pixel	bits/pixel	bits/pixel
111	7,7,7	7,7,7	7,7,7
112	3,5,4	5,3,4	3,4,4
121	3,3,3	5,4,3	4,5,4
122	1,3,2	3,2,2	2,3,2
12	2,4,3	2,3,2	3,2,2
21	0,2,0	2,1,1	1,2,1
22	0,1,0	0,0,0	0,0,0

Table 4.2: Subband-Scalar Quantization. Variance-Based Bit Allocation. 4:1 Compression Ratio.

Color Space	$L^*a^*b^*$	YIQ	RGB
Subband	bits/pixel	bits/pixel	bits/pixel
111	7,6,6	7,7,7	7,7,7
112	2,3,2	3,2,2	2,3,2
121	2,2,1	4,2,2	3,3,3
122	0,1,0	2,1,1	1,2,0
12	0,3,1	0,2,0	1,1,0
21	0,0,0	0,0,0	0,0,0
22	0,0,0	0,0,0	0,0,0

Table 4.3: Subband-Scalar Quantization. Variance-Based Bit Allocation. 8:1 Compression Ratio.

Color Space	Compression	Image	Δ E (ave)
$L^*a^*b^*$	4:1	GIRL	1.115
	4:1	DOLL	1.979
YIQ	4:1	GIRL	2.321
	4:1	DOLL	3.141
RGB	4:1	GIRL	1.909
	4:1	DOLL	2.321
$L^*a^*b^*$	8:1	GIRL	2.058
	8:1	DOLL	3.065
YIQ	8:1	GIRL	3.659
	8:1	DOLL	4.442
RGB	8:1	GIRL	3.079
	8:1	DOLL	3.649

Table 4.4: Subband SQ Distortion. Variance-Based Bit Allocation.

At a compression ratio of 4:1, the reconstructed GIRL images in all three color spaces look excellent. This is not too surprising since this image was used as the training sequence in the quantizer design. The results at the 8:1 compression ratio are very similar. It is very difficult to see the differences among the three color spaces. In all cases, there is a slight blurring of the image as some of the high frequency components are discarded. These images still look very good.

What is more interesting are the differences in the DOLL images in the three color spaces. Since the training sequence did not contain any highly saturated blue colors, these colors are desaturated in the reconstructed images. This is more visible in $L^*a^*b^*$ and YIQ then in RGB space. Figure 4.4 shows the DOLL image coded in RGB space at a compression ratio of 8:1 using the variance-based bit allocation. Figure 4.5 shows the DOLL image coded in YIQ and $L^*a^*b^*$ space with the same parameters. These last two color spaces each have a blue-yellow chrominance component. The quantizer design did not provide a sufficient dynamic range for this component. The choice of

a better training sequence will be addressed in a later chapter.

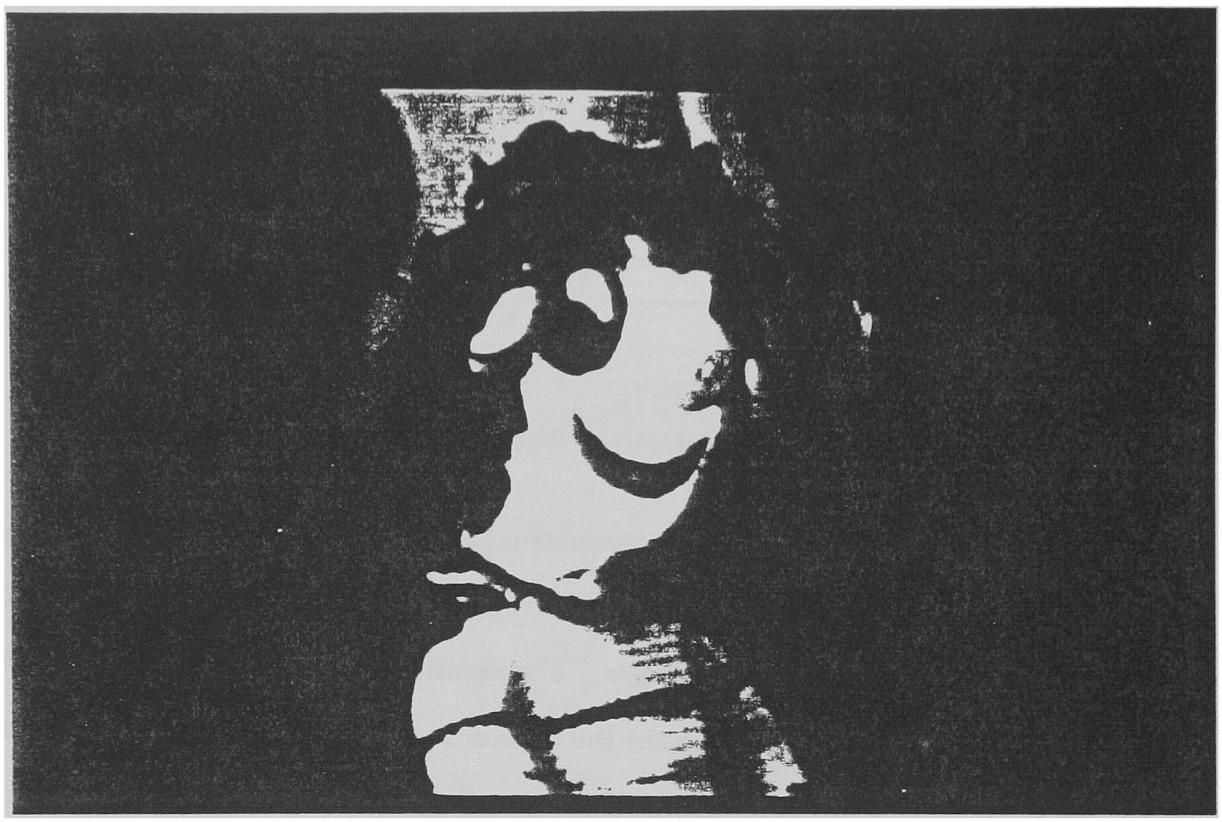


Figure 4.4: DOLL Image Coded in *RGB* Space. Variance-Based Bit Allocation. 8:1 Compression Ratio.

Figure 4.4: DOLL Image Coded in *RGB* Space. Variance-Based Bit Allocation. 8:1 Compression Ratio.

CHAPTER 5

VECTOR QUANTIZATION

5.1 Introduction

The distortion produced by a quantizer can be reduced by operating on a vector instead of a scalar. This is a consequence of Shannon's theory [81], and it is true even when the data source is memoryless. In the basic Full Search Vector Quantizer, each input vector is compared to a number of codevectors. The codevector that results in the minimum distortion is then selected. By ordering the codevectors, a label can be associated with each one. The label of the selected codevector is then used to choose the transmitted channel symbol. For a digital channel, the channel symbol is usually the binary representation of the label. The receiver uses the channel symbol as an index into a look-up table to determine the corresponding codevector. This codevector is used to approximate the original input vector.

The use of vector quantization for signal compression has been considered for a number of decades, but two problems had slowed its spread until the last decade. The first problem is due to the computational complexity of the vector quantizer. To achieve acceptable results, the codebook must usually contain a large number of codevectors. This requires enough memory to store the codevectors, and a processor fast enough to compare the incoming vectors with the codevectors. The speed of comparison becomes critical in a real-time system. The use of custom VLSI circuits and fast microprocessors has helped overcome this problem.

The second problem is the design of the codebook. For a given number of codevectors and a given distortion measure, these codevectors must be chosen so that the minimum possible distortion is achieved. This is a very difficult nonlinear optimization problem requiring knowledge of the multi-dimensional joint probability density function of the source. In 1980, an algorithm was derived by Linde, Buzo and Gray [49] that provides useful, although non-optimal, codebooks. This algorithm is known as the LBG or Generalized Lloyd algorithm. It formulates the codebook design as an optimization problem, and then uses an iterative method to locate a minimum. The method is guaranteed to converge, and does not require an exceptionally long run time; however, it may not converge to the global minimum. Newer methods that use simulated annealing [107] or the Kohonen learning algorithm [106] can now achieve better results, but often require more computer time.

There are a number of variations and improvements to the basic Full Search Vector Quantizer, and there are a numerous ways to employ vector quantizers to compress images. In the next section, the Full Search Vector Quantizer will be mathematically defined. This is the type of vector quantizer (VQ) used in our computer simulations. The distortion measure used is mean squared error. The codebooks are designed using the LBG algorithm which is described in the third section.

The number of calculations of the distortion between the input vector and the possible codevectors grows exponentially with the number of bits used to represent the codevectors. Although there are ways to reduce the total number of calculations [6], the growth is still exponential. For high bit rates, this gives the Full Search VQ a high computational complexity, and prevents it from being a practical choice for implementation. An alternative is the Tree Searched Vector Quantizer (TSVQ) described in the fourth section of this chapter.

The fifth section discusses improvements to the Full Search Vector Quantizer, including Classified Vector Quantizers, Finite-State Vector Quantizers, and Entropy-Constrained Vector Quantizers. The sixth section considers the application of vector quantization to subband coding. The mapping of the pixels to the vectors is described for the cases that are simulated. The final section discusses the choice of training sequences.

5.2 Full Search Vector Quantization

Given some distortion measure $d(\mathbf{x}, \hat{\mathbf{x}})$, a vector quantizer is a set of two mappings. The first takes a k-dimensional vector $\mathbf{x}$ and maps it into a set of channel symbols M. The second maps the channel symbols into a set of reproduction vectors $\hat{\mathbf{x}}$. This can be written as

$$U = \alpha(\mathbf{x}) \tag{5.1}$$

$$\hat{\mathbf{x}} = \beta(U) \tag{5.2}$$

where U is an element of set M [33]. The channel symbols are often binary vectors so that they can be easily transmitted over a digital channel. The set of reproduction vectors constitutes the codebook, C. The elements of this codebook are called the codevectors; they are denoted by $\mathbf{y}_i$, where i is used to index them.

The vector quantizer is optimal if it minimizes the average distortion between the input and reproduction vectors:

$$E\{\, d(\mathbf{x}, \beta[\alpha(\mathbf{x})])\,\}. \tag{5.3}$$

Since the multi-dimensional distribution for $\mathbf{x}$ is generally not known, one usually uses a long training sequence of data in the design of the vector quantizer. If the

source is stationary and ergodic then a vector quantizer designed from a sufficiently long training sequence will provide good performance on future samples taken from the same source. As long as the source is asymptotically mean stationary, this design method will still lead to good performance. The choice of the training sequence to use will be discussed later in this chapter.

The distortion measure used for image coding is usually that of mean squared error. In this case, the distortion can be written as

$$d(\mathbf{x}, \hat{\mathbf{x}}) = \sum_{j=0}^{k-1} (x_j - \hat{x}_j)^2 \tag{5.4}$$

[61]. A weighted mean square error has also been used [74]. While the first one is not ideal from a visually subjective viewpoint, it is simple to implement. Our approach is to do the quantization in a color space where the mean squared error is perceptually more meaningful.

5.3 LBG Algorithm

The vector quantizer described above is known as a Full Search Vector Quantizer because the encoder compares the input vector, $\mathbf{x}$, to all of the possible reproduction vectors, $\hat{\mathbf{x}}$, to determine which one will minimize the distortion. The codeword corresponding to this reproduction vector is then transmitted over the channel. At the receiver, a read-only-memory (ROM) is used to perform a table look-up to get the reproduction vector. The simplicity of the decoder makes this method attractive to broadcast systems where there must be a number of inexpensive receivers and few transmitters.

The design of the Full Search Vector Quantizer attempts to find the set of re-

production vectors that will yield the minimum average distortion. Finding a global minimum for this problem is extremely difficult so most methods try to find a local minimum that yields an acceptably low distortion. The most useful method is the LBG algorithm which is a clustering algorithm similar to the k-means algorithm [49]. It can be used with either an initial probability distribution or with a training sequence. This algorithm has no differentiability requirements so that purely discrete distributions can be used. The training sequence version was implemented and used to design the quantizers for the various subbands.

For a given training set, the algorithm depends on three parameters. They are the number of levels, N; a distortion threshold, $\epsilon \geq 0$; and an N level reproduction alphabet, $\hat{A}_0$. The number of levels is the most important of these three parameters; the average distortion generally decreases exponentially with increasing number of levels. The number of levels is either based on the required distortion, or determined by constraints on the transmission rate of the system. The LBG algorithm is relatively insensitive to the distortion threshold. A typical value is $\epsilon = 0.001$. Many of the simulations in this work use a value of 0.0008. If $\epsilon = 0$, the algorithm will run until there is no change in distortion from one iteration to the next. A small non-zero value significantly reduces the number of iterations required while keeping the final distortion small. There are two main ways of choosing the the initial reproduction alphabet. Both of these methods will be discussed after the algorithm is given. The LBG algorithm is:

0) The first step is an initialization. Start with a given number of levels, N, a fixed value of ϵ, and an initial alphabet $\hat{A}_0$. The training sequence is denoted by $\{\mathbf{x}_j; j = 0, \cdots, n-1\}$, where n is the number of vectors in the sequence. Set the distortion, D_{-1}, to ∞ and $m = 0$.

1) Given $\hat{A}_m = \{\mathbf{y}_i; i = 1, \cdots, N\}$, find the minimum distortion partition $\mathcal{P}(\hat{A}_m) = \{S_i; i = 1, \cdots, N\}$ of the training sequence: $\mathbf{x}_j \in S_i$ if $d(\mathbf{x}_j, \mathbf{y}_i) \leq d(\mathbf{x}_j, \mathbf{y}_l)$, for all l. Compute the average distortion

$$D_m = D(\{\hat{A}_m, \mathcal{P}(\hat{A}_m)\}) = \frac{1}{n} \sum_{j=0}^{n-1} \min_{\mathbf{y} \in \hat{A}_m} d(\mathbf{x}_j, \mathbf{y}). \tag{5.5}$$

2) If $(D_{m-1} - D_m)/D_m \leq \epsilon$, stop and use $\hat{A}_m$ as the reproduction alphabet; otherwise continue.

3) Find the optimal reproduction alphabet $\hat{\mathbf{x}}(\mathcal{P}(\hat{A}_m)) = \{\hat{\mathbf{x}}(S_i); i = 1, \cdots, N\}$ for the partition $\mathcal{P}(\hat{A}_m)$. Set $\hat{A}_{m+1} \doteq \hat{\mathbf{x}}(\mathcal{P}(\hat{A}_m))$. Replace m by $m + 1$ and go to 1. For the mean squared error distortion

$$\hat{\mathbf{x}}(S_i) = \frac{1}{\|S_i\|} \sum_{j: \mathbf{x}_j \in S_i} \mathbf{x}_j. \tag{5.6}$$

The above algorithm is guaranteed to converge because the distortion, as a function of iteration, is a monotonically decreasing sequence. Since the distortion must be non-negative, this sequence is bounded below by zero. Therefore, the sequence has a limit and the algorithm converges. "For finite alphabet distributions such as sample distributions, the algorithm always converges to a fixed-point quantizer in a finite number of steps" [49, page 89]. To see that the sequence is monotonically decreasing we examine both step 1) and step 3). In step 1), the reproduction vectors are given and the partition must be found. The optimal partition must map each vector in the training sequence into the reproduction vector that yields the minimum distortion; this is what is done.

In step 3), the partition is fixed and the optimal reproduction alphabet must be found. Denote the fixed partition as $S = \{S_i; i = 1, \cdots, N\}$. The optimal

reproduction alphabet, $\hat{\mathbf{x}}(S)$, must satisfy:

$$E(\ d(\mathbf{x}, \hat{\mathbf{x}}(S))\ | x \in S) = \min_{\mathbf{u}} E(\ d(\mathbf{x}, \mathbf{u})\ | x \in S) \tag{5.7}$$

where $E(\cdot)$ is expectation, $d(\cdot, \cdot)$ is the distortion between two vectors, and $\mathbf{x}$ is a real random vector with a given probability distribution. When the distortion function is mean squared error and when the training sequence is used as the discrete distribution then the reproduction alphabet given by Eq. (5.6) solves Eq. (5.7).

Two ways are commonly used to find the initial codebook. The first consists of placing the initial N reproduction vectors on a k-dimensional lattice. The spacing of the lattice is chosen so that it covers the region of k-dimensional space where the probability density function is non-zero. This method is simple and requires only one iteration of the LBG algorithm. The problem with this method is that one does not often know what the multi-dimensional density function looks like. The lattice, that is chosen, often does not use all of the reproduction vectors. While there are a number of ways to reuse these vectors, some trial and error is involved in choosing the dynamic range of the lattice so that the local minimum is a good one.

The second method of placing the initial vectors is known as the splitting algorithm. Let N be the actual number of reproduction levels and let M be the desired number of reproduction levels. One starts by finding the single optimal codevector for the entire training sequence. For the mean squared error distortion criterion, this is equivalent to finding the centroid of the training sequence, $\hat{A}_0(1) = \hat{\mathbf{x}}(A)$, for $N = 1$. Starting with the reproduction alphabet for size N, split each codevector $\mathbf{y}_i$ by adding and subtracting a small perturbation vector from them to yield $\mathbf{y}_i + \delta$ and $\mathbf{y}_i - \delta$. There are now $2N$ codevectors so replace N by $2N$ and run the LBG algorithm using the training sequence and these new codevectors. If $N = M$, then

the design is complete. If not, then split the codevectors again and continue. This method gives a sequence of codebooks of size $1, 2, 4, \cdots, M$.

The perturbation vector, δ, is fixed at the beginning of the algorithm. It is chosen so that most of the training vectors that have the given reproduction vector as the closest vector now have one of the two new vectors as their closest vector. If δ is too large, the first iteration of the algorithm may result in a small increase in distortion relative to the zeroth iteration. If this were to happen, then the value of δ should be decreased and the algorithm rerun. The resulting codebook is not particularly sensitive to the choice of the perturbation vector. A typical perturbation vector used in designing a four-dimensional vector quantizer is $(0.005\ 0.005\ 0.005\ 0.005)^T$.

5.4 Tree Searched Vector Quantization

When the codebook is large, the required number of distortion calculations and comparisons may make the Full Search method too computationally intensive to implement, since the number of both of these operations grows exponentially with the number of bits/vector. This is a large problem for real-time systems such as video that must process a given frame in a small amount of time. The Tree Searched Vector Quantizer (TSVQ) significantly reduces the number of calculations required at the cost of a larger memory size and a loss of optimality. Since the required amount of memory is not particularly large, this factor is usually of no major importance. Simulations show that even though the TSVQ is non-optimal, the loss in performance is also small [34, 35].

The structure of a TSVQ is that of a tree with $m + 1$ levels where level 0 is the root of the tree and level m is that of the leaves. Each node at level $l - 1$ has $N_l = 2^{R_l}$

branches leading to the next level. When $R_l = 1$ for all l then the tree is a binary tree. In any case, the tree can be described by an m-dimensional vector $\mathbf{R} = (R_1 \cdots R_m)^T$.

The vector to be quantized, $\mathbf{x}$, is first compared to N_1 codeword vectors, $\mathbf{y}_b$, at level $l = 1$. Each vector is associated with one of the branches leading to that level, and the branch that results in the minimum distortion between the input vector and the codeword vector is taken. The process is repeated starting at level $l = 1$ by comparing the input vector to the new codeword vectors, $\mathbf{y}_{x,b}$ of level 2 where the subscript x, b refers to the branch x that was taken in going from level 0 to level 1. The branch that is represented by the codevector yielding the minimum distortion is again chosen. This process is repeated until a leaf node is reached. At this time there is an m-dimensional vector $\mathbf{b} = (b_1 \cdots b_m)^T$ that is the path map taken through the tree. This path map is what is transmitted to the receiver where it is used to determine the reproduction vector, $\hat{\mathbf{x}}$, that was at the leaf of the encoding tree.

The design of the TSVQ is based on the LBG algorithm with splitting. The complete flow chart is given in [34]. Since a binary tree is the most common type of tree used in an implementation of a TSVQ, the discussion of the design will be limited to this case. Initially, a Full Search Vector Quantizer is designed with $N_1 = 2$ codevectors using the entire training sequence. The resulting levels are designed one level at a time. Starting at a node in level $l = 1, \cdots, m-1$ the codevectors at node $l + 1$ are designed as follows:

1) Split each codevector, $\mathbf{y}_{i,l}$, from level l into two vectors $\mathbf{y}_{i,l}$ and $\mathbf{y}_{i,l} + \delta$. This differs from the original splitting algorithm because one of the new vectors is the same as the old one. This is done to ensure that the distortion does not increase. The value of δ can be the same as in the Full Search LBG algorithm.

2) Run the LBG algorithm using these two vectors as the codevectors and only

the members of the original training sequence that reached the node corresponding to $\mathbf{y}_{i,l}$ in level l. The resulting two codevectors designated, $\mathbf{y}_{i,l+1}$ and $\mathbf{y}_{j,l+1}$, will be the nodes at the next level that are accessible from $\mathbf{y}_{i,l}$.

3) After completing steps 1) and 2) for all of the codevectors at level l, repeat these two steps for level $l+1$. Stop when the codevectors for level m have been designed.

The bit rate of an $(R_1 \cdots R_m)^T$ TSVQ is

$$R = \sum_{i=1}^{m} R_l \quad bits/vector \tag{5.8}$$

or $r = R/k$ bits/sample.

5.5 Improvements to Full Search VQ

5.5.1 Classified Vector Quantization

There are a number of ways to create the input vectors needed for vector quantization of images. Perhaps the simplest is to take a contiguous rectangular block of pixels in an image and map it into a k-dimensional vector. The image is divided into non-overlapping blocks and each block is quantized separately. Typical block sizes range from 2×2 through 5×5. While larger block sizes are more efficient from a compression standpoint, they have high coding complexity because of the large number of codevectors. Blocks of size 4×4 and 5×5 also suffer from edge degradation. This happens because the mean squared error distortion criterion is not a perceptually optimal one. While blocks with edges make up only a small percentage of the codevectors and the training sequence, they are perceptually very important. Vectors

containing these edges must be quantized with a lower mean squared error than those that do not contain edges; failure to do this results in visually objectionable artifacts.

The classified vector quantization method of Ramamurthi and Gersho [73, 74] is an attempt to correct both of these problems. They want to achieve edge integrity even if they must sacrifice some precision on both sides of the edge. The visual masking effect will allow this to look good if the edge is well preserved. The method assumes a composite model for the source, in which a number of vector subsources contribute to the image. Each subsource produces vectors belonging to a certain perceptual class. They used seven classes including four edge classes. The edge classes contained edges at orientations of 0, 45, 90, and 135 degrees. The other three classes were shade, midrange, and mixed. The shade class contains no significant gradient and the midrange class has a gradient but no edges. The mixed class contains edges with multiple orientations.

The encoding process first uses a classifier to determine which class the input vector $\mathbf{x}$ belongs to. If the input vector is classified as being a member of class i, then it is compared with all of the codevectors of this class, $\mathbf{y}_{i,j}$, and the codevector that yields the minimum distortion is chosen. A different distortion measure, $d_i(\mathbf{x}, \mathbf{y}_{i,j})$ can be used for each class. The label of the codevector is transmitted over the channel. At the receiver, the label is used as an entry in a look-up table to determine the reproduction vector, $\hat{\mathbf{x}}$.

5.5.2 Finite-State Vector Quantization

Another way to reduce the size of the codebook that must be searched, and to also improve performance is to use Finite-State Vector Quantization (FSVQ). This method is another generalization of the basic vector quantizer, and it is discussed in detail in

[16, 24]. Like Classified Vector Quantization, a set of vector quantizers is designed and a particular vector quantizer is chosen to encode each input vector. Instead of choosing the quantizer based on a classification of the input vector, a finite-state machine is used. The state of this machine determines the choice of vector quantizer.

Starting with a set of k-dimensional vectors $\{\mathbf{x}_n, n = 0, 1, \cdots\}$, a finite state space, S, and an initial state S_0, a FSVQ consists of three mappings. The first one is the encoder that maps the input vectors into the channel sequence $\{U_n, n = 0, 1, \cdots\}$. If the input vectors come from an alphabet A, and the channel sequence has a finite alphabet M, then the encoder mapping is: $\alpha : A \times \mathrm{S} \rightarrow \mathrm{M}$. The second mapping is the decoder that maps the channel sequence to a reproduction vector, $\hat{\mathbf{x}}$, in the reproduction alphabet $\hat{A}$. This is represented by $\beta : \mathrm{S} \times \mathrm{M} \rightarrow \hat{A}$. The last mapping is the transition function of the finite state machine. It is given by $f : \mathrm{S} \times \mathrm{M} \rightarrow \mathrm{S}$. All of the channel symbols, states, and reproduction vectors are determined by the input sequence and the initial state. The relations are

$$U_n = \alpha(\mathbf{x}_n, S_n), \quad \hat{\mathbf{x}}_n = \beta(S_n, U_n) \tag{5.9}$$

$$S_{n+1} = f(S_n, U_n), \quad n = 0, 1, \cdots \tag{5.10}$$

Again, the receiver can use table look-up to determine the reproduction vector. Because the next state function depends only on the present state and the channel symbol, the decoder does not need any additional information to determine the present state. Starting with the initial state, it can track the state transitions. The reproduction vectors for each state, s, comprise the codebook $C_S = \{\beta(s, u); u \in \mathrm{M}\}$. The total codebook for the system is the union of the codebooks for each state.

5.5.3 Entropy-Constrained Vector Quantization

The Full Search Vector Quantizer is designed to have a specified number of reproduction codevectors, $\hat{\mathbf{x}}$, in the codebook. These codevectors are placed in some order, and the index number of the codevector is what becomes the channel symbol, U. For a digital channel, the channel symbols are binary strings that represent the index number. If these binary strings are direct representations of the index number, they will all be the same length. If the codevectors are not all equally probable, then a further reduction in bit rate can be achieved by entropy coding the channel symbols. This can be done in a number of ways including Huffmann coding, arithmetic coding, or Ziv-Lempel coding.

Entropy-Constrained Vector Quantization [12] (ECVQ) is a source coding method where the vector quantizer is designed subject to an entropy constraint. Designing the vector quantizer with this constraint can lead to an improvement in signal-to-noise ratio when compared to designing the vector quantizer subject to a constraint on the number of reproduction levels and then adding entropy coding to the system. The ECVQ system consists of four mappings. The first one maps each input vector into the label for the closest codevector. This is written as

$$\mathcal{J} = \alpha(\mathbf{x}) \tag{5.11}$$

where $\mathcal{J}$ is the index for the closest codevector. The second mapping is lossless and transforms the index into a variable length channel symbol. It is given by

$$U = \gamma(\mathcal{J}) \tag{5.12}$$

where U is the channel symbol. Eq. (5.12) must be both invertible and uniquely decodable.

The receiver first recovers the index from the channel symbol according to

$$\mathcal{J} = \gamma^{-1}(U) \tag{5.13}$$

and then uses the index to obtain the reproduction vector

$$\hat{\mathbf{x}} = \beta(\mathcal{J}). \tag{5.14}$$

Eqs. (5.11) through (5.14) can also be used to represent any vector quantizer followed by entropy coding. What makes ECVQ different is that these equations are all used in the design of the codebook. This is done by first expressing the coder as a Lagrangian functional given by

$$J_\lambda(\alpha, \gamma, \beta) = E[\, d(\mathbf{x}, \beta(\alpha(\mathbf{x}))) + \lambda |\gamma(\alpha(\mathbf{x}))| \,] \tag{5.15}$$

where $|\gamma(\cdot)|$ is the length of $\gamma(\cdot)$. This functional is then minimized using an iterative descent algorithm similar to the LBG algorithm. More details can be found in [12].

5.6 Color Subband Coding with Vector Quantization

In Chapter 4, we have seen some results for color subband coding with scalar quantization. The bit rate can be further reduced by exploiting the correlation among the color components of a subband, among the subbands, and among the pixels within a color component and subband. The common way to implement vector quantization on monochrome images is to create a vector from contiguous blocks of pixels [30]. Westerink *et al.* [102] applied vector quantization to subband coding by filtering the image into sixteen subbands and using a sixteen-dimensional vector quantizer where each vector contained information from the same pixel location of all sixteen subbands.

A subband/VQ scheme where the vectors are made from rectangular blocks has been studied for both monochrome [2] and color images [43, 44]. Subband coding with entropy-constrained vector quantization has also been recently studied for monochrome images [45]. Combining the three color components of a pixel into a vector and then implementing a vector quantizer has been used to design color look-up tables [11]. We have extended this last approach to subband coding and presented some initial results in [95]. Nasrabadi and King presented a good review of the use of vector quantization for image coding in [61].

In Chapters 7 and 8, simulation results will be presented for three cases. Case 1, mentioned in the previous paragraph, scalar quantizes each color component of the lowest frequency subband and vector quantizes the six higher frequency subbands. The vector quantization uses three-dimensional vectors created by combining the three color components of each subband pixel. Case 2 is the straightforward extension of subband/VQ coding to color images where each subband color component is coded separately with four-dimensional vectors created from 2×2 blocks in each color component. The three color components of the lowest frequency subband are scalar quantized to lessen the computational complexity. Case 3 is the same as Case 2, except that the chrominance components of the lowest frequency subband are also vector quantized using the same four-dimensional vectors. Table 5.1 shows the size of the vectors and how they are created for Cases 2 and 3.

Subband	Case 2			Case 3		
	Lum	R-G	B-Y	Lum	R-G	B-Y
111	1×1	1×1	1×1	1×1	2×2	2×2
112	2×2	2×2	2×2	2×2	2×2	2×2
121	2×2	2×2	2×2	2×2	2×2	2×2
122	2×2	2×2	2×2	2×2	2×2	2×2
12	2×2	2×2	2×2	2×2	2×2	2×2
21	2×2	2×2	2×2	2×2	2×2	2×2
22	2×2	2×2	2×2	2×2	2×2	2×2

Table 5.1: Composition of Vectors for Cases 2 and 3. Rectangular Configuration. Lum refers to the luminance component, R-G refers to the red-green component, and B-Y refers to the blue-yellow component of the color spaces, $L^*a^*b^*$, YIQ, or AC_1C_2.

5.7 Vector Quantizer Implementation

5.7.1 Choice of Training Sequence

The rectangular and diamond subband configurations studied in this work were given in Chapter 3. The former configuration has one low frequency subband and six high frequency subbands, while the latter configuration also has one low frequency subband but only has four high frequency ones. Cases 1, 2, and 3 described above have the same meaning for both the rectangular and diamond configurations regarding which subbands are scalar or vector quantized. This means that for Case 1, the lowest frequency subband is scalar quantized and the higher frequency subbands are vector quantized with three-dimensional vectors created from the three color components. For Cases 2 and 3, the diamond configuration creates the four-dimensional vectors from 2×2 diagonally oriented blocks.

The LBG algorithm is used to design the codebooks in all cases [49]. A Full Search VQ is used since the maximum number of levels needed for each higher frequency subband is not large. Case 3 was run only for high compression ratios so that the

number of levels for the lowest frequency chrominance components were also not too large. The one-dimensional version of the LBG algorithm, also known as Lloyd's algorithm [51], is used to design the scalar quantizers used in the lowest frequency subband. The initial codevectors are obtained by using the splitting method. This requires that codebooks for all rates smaller than the desired rate be found first. By doing this, one obtains a distortion-rate curve for each particular vector quantizer. These curves will be used in the bit allocation algorithm derived in the next chapter.

Ideally, the statistics of the training sequence should match those of the ensemble of images to be coded. If these images were stationary and ergodic, then this condition could be more easily achieved. However, this is not the case, so in practice the training sequence must be chosen so that a few criteria are satisfied. First, the dynamic range of the images in the training sequence should be as large as those in the image ensemble. Failure to met this condition will result in the inability to reproduce the extreme values. This criterion is less difficult to achieve for gray-scale images than for color images. For the latter, the dynamic range of each color component of the training sequence must be large enough. Since the training sequence is a discrete set of vectors, the second criterion is that there are vectors in most regions of the n-dimensional vector space.

To ensure that the above conditions are satisfied, the training sequence is usually created from a number of images that are similar to those to be coded. The use of one to ten images is typical. Chan and Chow [11] use the input image as the training set. This achieves good results for that image, but requires that the codebook design be run for each input image. Budge *et al.* [8] use three 512×512 pixel color images as the training set and then code images of size 256×256 pixels. If the images are of a special type that requires high precision, such as MRI images, then more images

should be used. Riskin and Gray [76] use twenty magnetic resonance brain scans in their design of a TSVQ; more images are usually required to design a TSVQ so that there are training vectors for each leaf of the tree.

Two sets of training images are used. The first set of codebooks was designed using the GIRL image as the training image. This image is designated Training Set 1 (TS 1). Both the GIRL and the DOLL images are coded, so the first image is inside the training set and the second is outside of it. The second set of training images, designated Training Set 2 (TS 2), consists of four 256×256 color images also taken from the original toy store image. All codebooks were designed using these four images so both the GIRL and DOLL images are outside the training set. TS 1 does not provide enough saturated colors to properly encode the DOLL image. TS 2 was created so that it had a large range of colors including various shades of skin. This training sequence met the above conditions for being a good training sequence. Simulation results for both training sets will be discussed in later chapters.

5.7.2 Number of Codebooks Required

For the rectangular configuration, each image in the training set is filtered into seven subbands. For Case 1, the three color components of the pixels in each higher frequency subband are combined into vectors, and these vectors become the training vectors for that subband. Nine codebooks must be designed, three for the color components of the lowest subband, and one codebook for each of the other six subbands. For Case 2, each block of four pixels in a color component of a higher frequency subband is placed in a vector. The vectors for that component and subband are then used as the training sequence. Now, a total of 21 codebooks must be designed, one for each component of the seven subbands. When four images are used as the training

set, each image is filtered separately into the subbands and then mapped into the vectors. After this, the union of the vectors from the four images becomes the set of training vectors for that subband (and color component).

The procedure for the diamond configuration is the same as for the rectangular configuration, except that the image is filtered into five subbands. Seven codebooks must be designed for Case 1, and 15 codebooks must be designed for Case 2. Note that in both configurations, the lowest subband is scalar quantized in both Case 1 and Case 2. Therefore, these codebooks only have to be designed once for each configuration. These procedures are repeated for the color spaces YIQ, $L^*a^*b^*$, and AC_1C_2 in both configurations.

CHAPTER 6

PERCEPTUALLY OPTIMAL BIT ALLOCATION

6.1 Introduction

As in all subband coding schemes, one must determine how to allocate bits among the subbands. In color subband coding, one faces the additional problem of allocating bits among the color components of each subband if the components are treated separately. A number of algorithms have been proposed to solve this problem. They can generally be classified into two separate categories. The first method, discussed in Chapter 4, uses a model of the optimal scalar quantizer and the variances of the subband components to provide a mathematically suboptimal, yet useful, bit allocation. Because the problem is solved by using a continuous Lagrange multiplier method, the results are not constrained to non-negative integers. The non-negativity constraint can, however, be added [80], but the results are still real numbers. These can be rounded to the nearest integer and adjusted, if necessary, to yield the required bit allocation.

Fox [25] introduced the second method. It uses a generalized Lagrange multiplier method derived by Everett [18]. This method, known as Marginal Analysis, was originally used to solve military allocation problems, and was used by Trushkin [89, 90] for bit allocation of vector quantizers. Shoham and Gersho [83] proposed an extension of this method for arbitrary quantizers where the distortion-rate curves were neither convex nor monotonic. Westerink *et al.* [101] generalized Trushkin's work to allow

non-integer bit allocations, and then used their version for the monochrome subband case. Because Marginal Analysis uses the actual distortion-rate curves for the quantizers, any types of quantizers can be implemented. Scalar and vector quantizers can be used for different subband components in the same system. This is important since it is desirable to scalar quantize each color component of the lowest subband because of the large number of levels required, and vector quantize the higher subbands to achieve a greater compression ratio.

The latter method will be described in the following section. It has been used to obtain the bit allocation when the higher frequency subbands were vector quantized. We propose an extension of the method for subband coding of color images by adding weights to the cost functions based on properties of the human visual system. This generalization is important since the human visual system not only has different responses for the color components, but these responses are also frequency dependent. The new bit allocation scheme will attempt to provide the perceptually optimal solution.

6.2 Marginal Analysis Method

In this section, we show how the method of Marginal Analysis can be used to provide an optimal solution to the bit allocation problem. The discussion of this method will illustrate Case 1. The definitions of the cases were given in Section 5.6 of Chapter 5. The composition of the vectors for Cases 2 and 3 is given in Table 5.1 in the same section. In Case 1, each color component of the lowest frequency subband is scalar quantized, and the other six subbands are vector quantized by combining the three color components of each pixel into a three-dimensional vector. The formulation of

the bit allocation problem for cases 0, 2, and 3 is treated similarly.

The bit allocation problem can be formulated as follows:

$$min_{\mathbf{b}\in S} D(\mathbf{b}) \tag{6.1}$$

subject to

$$C(\mathbf{b}) \leq B, \tag{6.2}$$

where $D(\mathbf{b})$ is the distortion function, B is the total bit rate, and $C(\mathbf{b})$ is the number of bits required by strategy $\mathbf{b}$. The set S is the set of all possible bit allocations among the subbands.

The bit allocation problem reduces to what is known in the operations research literature as the cell problem [18], because the mean squared errors for each vector quantizer are summed to obtain the total distortion. The optimization problem is now:

$$min_{b_i\in S_i} \sum_{c=1}^{3} D_{1,c}(b_{1,c})/w_{1,c} + \sum_{i=2}^{7} D_i(b_i)/w_i \tag{6.3}$$

subject to

$$\sum_{c=1}^{3} C_{1,c}(b_{1,c}) + \sum_{i=2}^{7} C_i(b_i) \leq B. \tag{6.4}$$

The function $D_i(b_i)$ is the distortion-rate curve for subband i. The second subscript for the first subband is used to designate the three color components, since these components are scalar quantized separately. The functions $C_i(b_i) = c_i b_i$, where the cost factors c_i are needed because of the different sizes of the subbands. The cost functions are given in Tables 6.1 and 6.2 for the four cases. The w_i are perceptual weighting factors that are determined by the visual importance of the subband and color component. They will be discussed in more detail later; they were

placed in the denominator so that they can be treated like the costs. Since there are seven subbands and the lowest has three components, $\mathbf{b}$ is a nine-dimensional vector, $\mathbf{b} = (b_{1,1}\ b_{1,2}\ b_{1,3}\ b_2 \cdots b_7)^T$. The components of the vector can be relabeled to give $\mathbf{b} = (b_1 \cdots b_9)^T$.

In Tables 6.1 and 6.2, the numbers of the subbands correspond to those in the rectangular configuration whose spatial frequency decomposition is shown in Figure 3.9 in Chapter 3. A different set of costs is used for the diamond configuration. The color space used is either $L^*a^*b^*$, YIQ, or AC_1C_2 space. All three of these spaces have a luminance component, a red-green component, and a blue-yellow component. The labels, Lum, R-G, and B-Y, are used to denote these three color components in the space used.

	Case 0			Case 1		
Subband	Lum	R-G	B-Y	Lum	R-G	B-Y
111	1/16	1/16	1/16	1/16	1/16	1/16
112	1/16	1/16	1/16	1/16		
121	1/16	1/16	1/16	1/16		
122	1/16	1/16	1/16	1/16		
12	1/4	1/4	1/4	1/4		
21	1/4	1/4	1/4	1/4		
22	1/4	1/4	1/4	1/4		

Table 6.1: Subband Costs - Cases 0 and 1. The costs depend on the size in pixels of the subbands. For the scalar quantizers these are the costs of adding one bit/pixel. For the vector quantizers these are the costs of adding one bit/vector. Lum refers to the luminance component, R-G refers to the red-green component, and B-Y refers to the blue-yellow component. In Case 1, the vectors include the three color components so there is only one weight per subband.

Figure 6.1 shows the mean squared error versus bit rate for subbands 112, 121, and 122 in $L^*a^*b^*$ space for Case 1. One can see that these functions are convex. The definition of convex for functions defined only on the integers is that the first

	Case 2			Case 3		
Subband	Lum	R-G	B-Y	Lum	R-G	B-Y
111	1/16	1/16	1/16	1/16	1/64	1/64
112	1/64	1/64	1/64	1/64	1/64	1/64
121	1/64	1/64	1/64	1/64	1/64	1/64
122	1/64	1/64	1/64	1/64	1/64	1/64
12	1/16	1/16	1/16	1/16	1/16	1/16
21	1/16	1/16	1/16	1/16	1/16	1/16
22	1/16	1/16	1/16	1/16	1/16	1/16

Table 6.2: Subband Costs - Cases 2 and 3. The costs depend on the size in pixels of the subbands. For the scalar quantizers these are the costs of adding one bit/pixel. For the vector quantizers these are the costs of adding one bit/vector. Lum refers to the luminance component, R-G refers to the red-green component, and B-Y refers to the blue-yellow component.

differences are all negative and increasing as the bit rate increases. When all of the distortion-rate curves are convex, the problem of finding the optimal bit allocation is considerably simplified. The Marginal Analysis algorithm will give the optimal solution.

6.2.1 Marginal Analysis Algorithm

The allocation algorithm is iterative, and is given as follows:

1. Initially set $\mathbf{b}^0 = (0 \cdots 0)^T$. Set $k = 1$.

2. $\mathbf{b}^k = \mathbf{b}^{k-1} + \mathbf{e}_i$, where $\mathbf{e}_i$ is the ith unit vector and i is the index which has the maximum

$$[D_j(b_j^{k-1}) - D_j(b_j^{k-1} + 1)]/(w_j c_j).$$

3. If $C(\mathbf{b}^k) > B$ then stop, else $k = k + 1$ and go to 2.

When the algorithm terminates, one may have to adjust the last few of allocations so the total bit rate matches B and does not exceed it. If all of the subbands had the

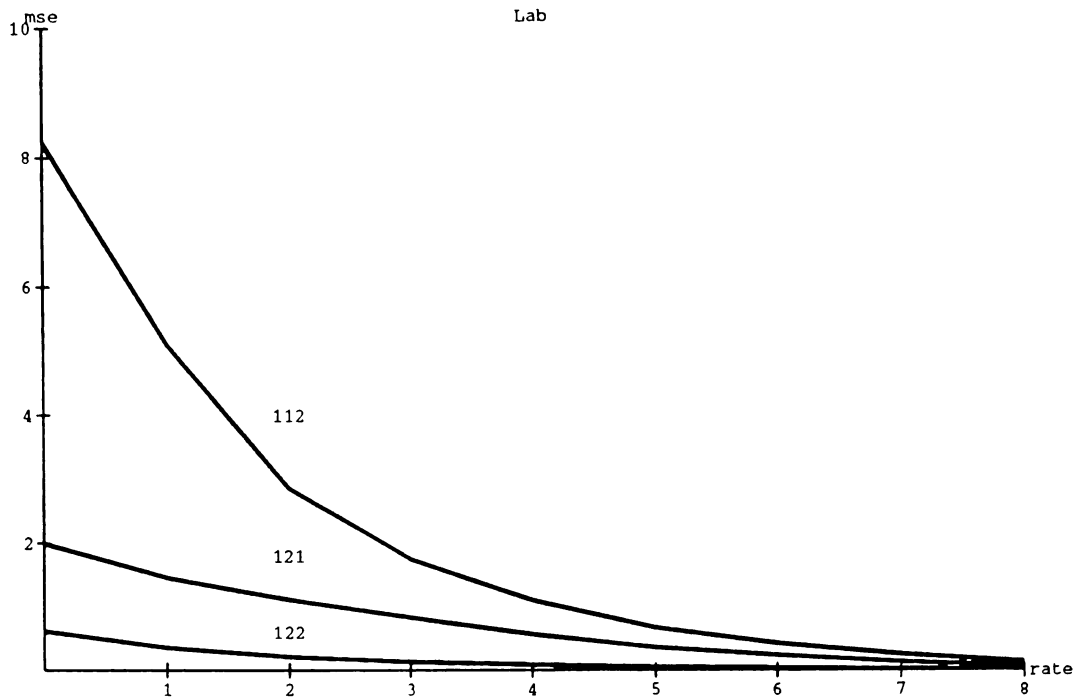


Figure 6.1: Distortion-Rate Curves of Subbands 112, 121, and 122 in $L^*a^*b^*$ Space.

same number of pixels, this would not be necessary.

6.2.2 Quasiconvexity Condition

The Marginal Analysis algorithm is guaranteed to provide the optimal solution only if all of the distortion-rate curves are convex. Unfortunately, this is not true, especially for Case 2 and Case 3. The LBG algorithm is guaranteed to converge to only a local minimum. The change in distortion between bit rate $i-1$ and i is not always greater than the change between bit rate i and $i+1$. This non-convexity most often occurs for the higher frequency subbands where the decrease in distortion from a bit rate of zero bits/pixel to one bit/pixel is less than the decrease from a bit rate of one bit/pixel to two bits/pixel.

The quasiconvexity condition is a sufficient condition for the bit allocation to be

optimal [90]. This condition is:

$$s_i(\mathbf{b}^*) \leq S_j(\mathbf{b}^*); \;\; i,j = 1,\cdots,9; \; i \neq j \tag{6.5}$$

where

$$s_i(\mathbf{b}^*) = max_{b_i^* \leq b_i < B}(D_i(b_i) - D_i(b_i+1)/(w_i c_i)), \tag{6.6}$$

$$S_j(\mathbf{b}^*) = min_{0 \leq b_j < b_j^*}(D_j(b_j) - D_j(b_j+1)/(w_j c_j)), \tag{6.7}$$

and $\mathbf{b}^* = (b_1^* \cdots b_9^*)^T$ is the bit allocation from the Marginal Analysis algorithm.

The bit allocation algorithm first uses Marginal Analysis to find an initial allocation. If the quasiconvexity condition is satisfied, the allocation is optimal and the procedure is finished. If not, the subbands or subband components that cause the nonconvexity are determined and the locations on the distortion-rate curves where convexity is lost are found. This information is provided by the bit allocation program. Since the program lists the order that bits are allocated to the various subbands, a procedure of pair-wise changes is used. The total distortion is first calculated and then a pair-wise change is made in the bit allocation. The new distortion is calculated and compared to the previous one. If the distortion is lower, the new allocation is kept. In this procedure, input is required at each iteration to determine which pair-wise changes are to be used in the allocation.

6.3 Mean Detection Threshold

In some initial simulations, the perceptual weights, w_j, were all set to unity. However, the human visual system is both color and frequency dependent. Subjective experiments have shown [46] that even in a perceptually uniform color space such as

$L^*a^*b^*$ space, the mean detection threshold is a function of spatial frequency, orientation, luminance, background color, and direction of the color transition. Using this research, we have derived sets of weighting factors. These sets give a numerical weight for each subband color component, and are based on a specified viewing distance from the image. The next few subsections will describe how the mean detection threshold data was measured, and how it can be transformed into the different color spaces. The following section will discuss the calculation of the perceptual weights from the mean detection threshold data.

6.3.1 Experimental Measurements

The experiments [46] measured the smallest change in a color necessary for a human observer to notice that change. Sinusoidal variations in color at different spatial frequencies were displayed on a monitor, and the amplitude of each variation was increased until the pattern was visible or an upper limit was reached. The pattern was then made clearly visible and the amplitude was decreased until the pattern was not visible or a lower limit was reached. The average of the ascending threshold and the descending threshold gives the detection threshold for one subject. The average of the detection thresholds for all six subjects gives the mean detection threshold. The details of these experiments can be found in [46, 72].

The initial data was measured in xyY space where x and y are the C.I.E. chromaticity coordinates and Y is the luminance. Four representative chromaticities were chosen, one each in the white, red, green, and blue regions of the monitor's gamut. The chromaticity coordinates for these points are given in Table 6.3. About these points, the mean detection thresholds were measured for transitions along the luminance, red-green, and blue-yellow directions for six different spatial frequencies and

for Y tristimulus values of 5, 10, and 20 cd/m^2. Figure 6.2 shows the red-green and blue-yellow directions on the xy chromaticity diagram for transitions about the white point, $(x, y) = (0.33, 0.35)$. Included on the plot is the spectral locus and the monitor's gamut for $Y = 5\ cd/m^2$.

Hue	Chromaticity Coordinates	
	x	y
White	0.33	0.35
Red	0.42	0.39
Green	0.34	0.46
Blue	0.29	0.29

Table 6.3: Representative Chromaticities. These are the four background colors used in the visual sensitivity experiments.

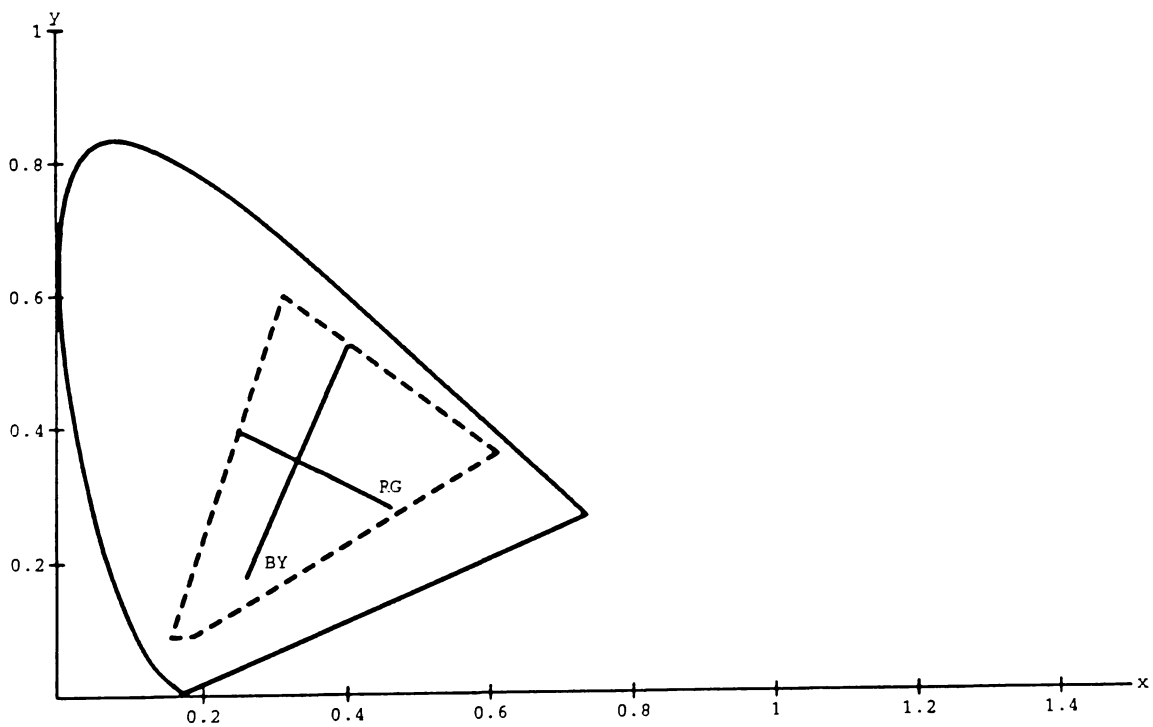


Figure 6.2: Directions of Variation about the White Point. The white point's chromaticity is (0.33,0.35). The monitor's gamut for $Y = 5\ cd/m^2$ is also plotted.

For transitions along the red-green and blue-yellow directions, each experimentally determined value of the mean detection threshold provides two chromaticity coordinates. These coordinates correspond to the maximum and minimum values of the sinusoidal variation about the representative chromaticity. The two chromaticity coordinates are given parametrically by:

$$x = x_0 \pm \Delta x \times t \tag{6.8}$$

$$y = y_0 \pm \Delta y \times t$$

where t is the mean detection threshold, and the point (x_0, y_0) is one of the four representative chromaticities. The numbers Δx and Δy are the changes in the x and y chromaticity coordinates. They determine the transition direction on the chromaticity diagram, and are different for the red-green and blue-yellow directions. Table 6.4 gives the values of Δx and Δy for all three directions of color transition.

Direction	ΔY	Δx	Δy
Luminance	0.0124	0.0	0.0
Red-Green	0.0	0.000655	-0.000357
Blue-Yellow	0.0	0.000283	0.000689

Table 6.4: Amounts of Change for Each Direction.

For a constant luminance, the xy contrast is given by $\Delta xy = ((\Delta x)^2 + (\Delta y)^2)^{1/2}$. The data was measured in contrast units corresponding to an absolute difference of $\Delta xy = 0.000745$. Since the variations in color were sinusoidal about the center point, the contrast is equivalent to twice the variation from the center to either end point. For the red-green and blue-yellow transitions, the mean detection threshold, t, in Eq. (6.8) is the amplitude from the center, and the contrast is

$$((0.000655)^2 + (0.000357)^2)^{1/2} = ((0.000283)^2 + (0.000689)^2)^{1/2} = 0.000745. \tag{6.9}$$

For the luminance direction, the chromaticity coordinates remain unchanged and only the luminance value is varied as will be seen in the next subsection.

6.3.2 Transformation to XYZ Space

To convert the mean detection threshold data to the color spaces YIQ, $L^*a^*b^*$, and AC_1C_2, the data must first be transformed from xyY space to XYZ space. The procedure is different for the luminance and chrominance changes, so each one will be discussed. For both the red-green and blue-yellow directions, Eq. (6.8) provides two chromaticity coordinates for each mean detection threshold value. Call these two points (x_1, y_1, z_1) and (x_2, y_2, z_2), where $z_i = 1 - x_i - y_i$ for $i = 1, 2$. These two points are then converted to XYZ space for the given value of Y. That is,

$$Y_i = Y \quad X_i = x_i Y / y_i \quad Z_i = z_i Y / y_i \tag{6.10}$$

for $i = 1, 2$.

For the luminance direction, each mean detection threshold value provides two XYZ tristimulus values. The equations are:

$$\begin{aligned} X_i &= X_0 \pm \Delta Y \times (\frac{X_0}{Y_0}) \times t \\ Y_i &= Y_0 \pm \Delta Y \times t \\ Z_i &= Z_0 \pm \Delta Y \times (\frac{Z_0}{Y_0}) \times t \end{aligned} \tag{6.11}$$

where ΔY is given in Table 6.4, t is the mean detection threshold, and $i = 1, 2$. The tristimulus vector $(X_0\ Y_0\ Z_0)^T$ contains the tristimulus values of the center point at the desired luminance value, Y. It is computed in the same way as Eq. (6.10).

Once the data has been converted to XYZ space, the mean detection threshold in this color space can be computed. It is the Euclidean norm of the difference of the two tristimulus vectors given by Eq. (6.10) for either the red-green or blue-yellow direction, or by Eq. (6.11) for the luminance direction. The value is given by:

$$t_{XYZ} = \|\mathbf{T}_1 - \mathbf{T}_2\| \tag{6.12}$$

where $\mathbf{T}_1 = (X_1\ Y_1\ Z_1)^T$ and $\mathbf{T}_2 = (X_2\ Y_2\ Z_2)^T$. The mean detection thresholds in the color spaces $L^*a^*b^*$, AC_1C_2, and YIQ are the Euclidean norms of the differences of the corresponding vectors obtained by transforming the two XYZ tristimulus vectors to the desired color space.

6.3.3 Transformation to $L^*a^*b^*$ Space

The transformation from XYZ space to $L^*a^*b^*$ is defined in terms of the tristimulus values of a white point. In Chapter 2, it was mentioned that the monitor's white point was used. The transformation of the mean detection data to $L^*a^*b^*$ space done by Krishnakumar [46] used an arbitrary white point with chromaticity of (0.33,0.33). The luminance was set to the experimental value of either $Y = 5$, 10, or 20 cd/m^2. For the luminance value of $Y = 5$ cd/m^2, this results in a white point of:

$$\begin{bmatrix} X_0 \\ Y_0 \\ Z_0 \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \\ 5 \end{bmatrix}. \tag{6.13}$$

For this white point, the plot of the red-green and blue-yellow directions on the a^*b^* plane is shown in Figure 6.3. Since the luminance value used to transform the data from xyY to XYZ space is also used for the white point, $L^* = 100$ and is a constant.

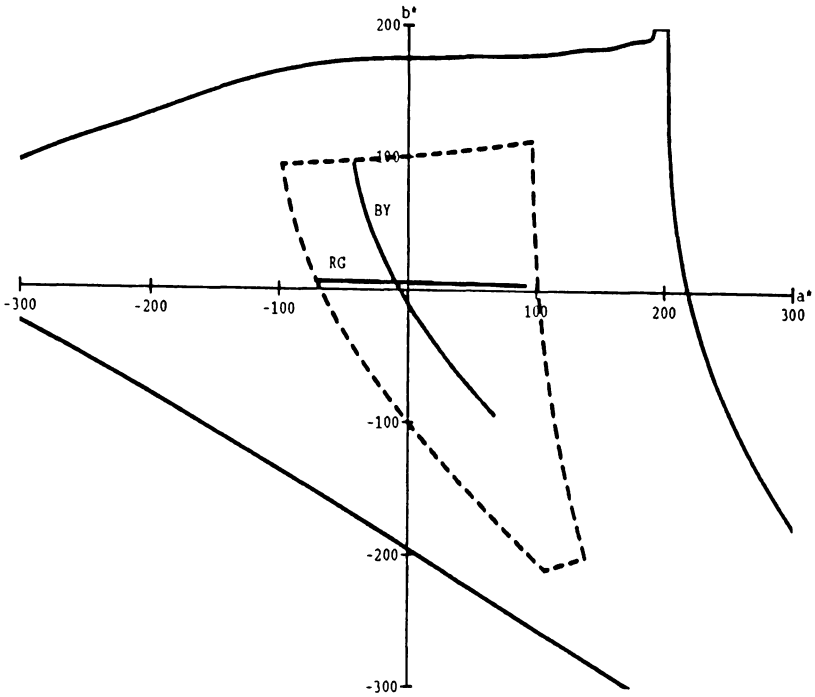


Figure 6.3: Directions of Variation in $L^*a^*b^*$ Space. White Point Chromaticity is (0.33,0.33). $Y = 5\ cd/m^2$.

Using the monitor's white point of $(34.51\ 37.36\ 37.16)^T$, the red-green and blue-yellow directions in $L^*a^*b^*$ space are shown in Figure 6.4. The data was still measured with $Y = 5\ cd/m^2$, but this now maps to $L^* = 43.34$. In both plots, the red-green direction is essentially along the a^* axis as is desired. However, the blue-yellow direction is not along the b^* axis. Since both transformations use a white point that is not the same as the white point in Table 6.3, there are slight offsets as seen in the plots. The data obtained from the first transformation was used to derive the perceptual weights for $L^*a^*b^*$ space for the initial set of simulation results. The final set used the data obtained from the second transformation.

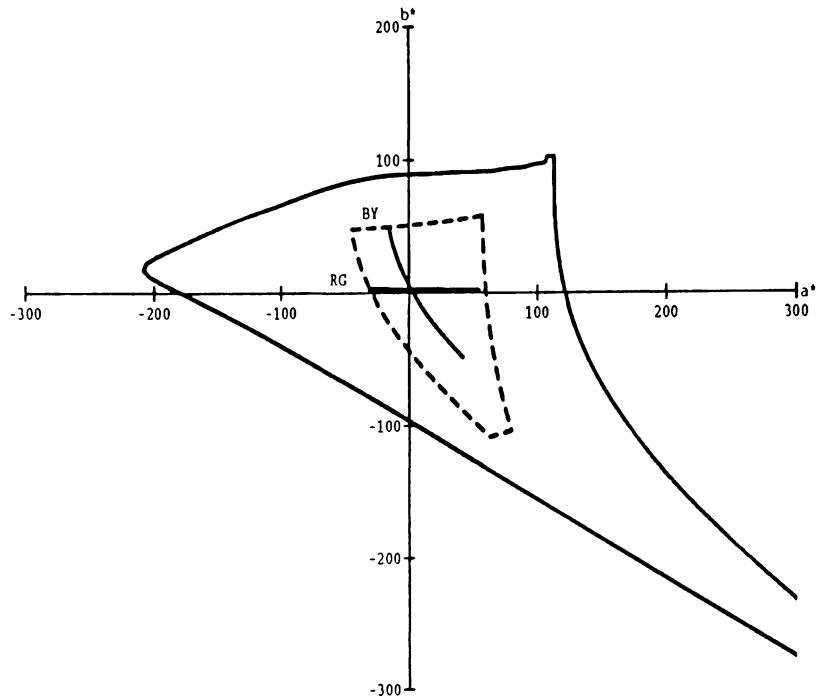


Figure 6.4: Directions of Variation in $L^*a^*b^*$ Space. White Point Chromaticity is (0.3165,0.3426). $Y = 5\ cd/m^2$.

6.3.4 Transformation to AC_1C_2 Space

The transformation from XYZ to AC_1C_2 space consists of three parts as was discussed in Chapter 2. The first part is a linear transformation to the cone space, LMS. The mapping used by Faugeras [21] uses the transformation from [105]. He uses a white point mapping such that either standard illuminant C or illuminant D_{65} with unity luminance maps to the origin.

It is simple enough to derive a transformation to AC_1C_2 space such that any desired white point maps to the origin. While standard illuminant C is used as the white point in the N.T.S.C. XYZ to YIQ transformation, the use of the monitor's white point would be more consistent with its use in the transformation to $L^*a^*b^*$ space. This transformation was implemented, and the red-green and blue-yellow directions

are shown in Figure 6.5. Again, one notices that there is an offset between the center of the color transitions and the origin. More importantly, these two directions of color transition are almost orthogonal and are very close to the axes.

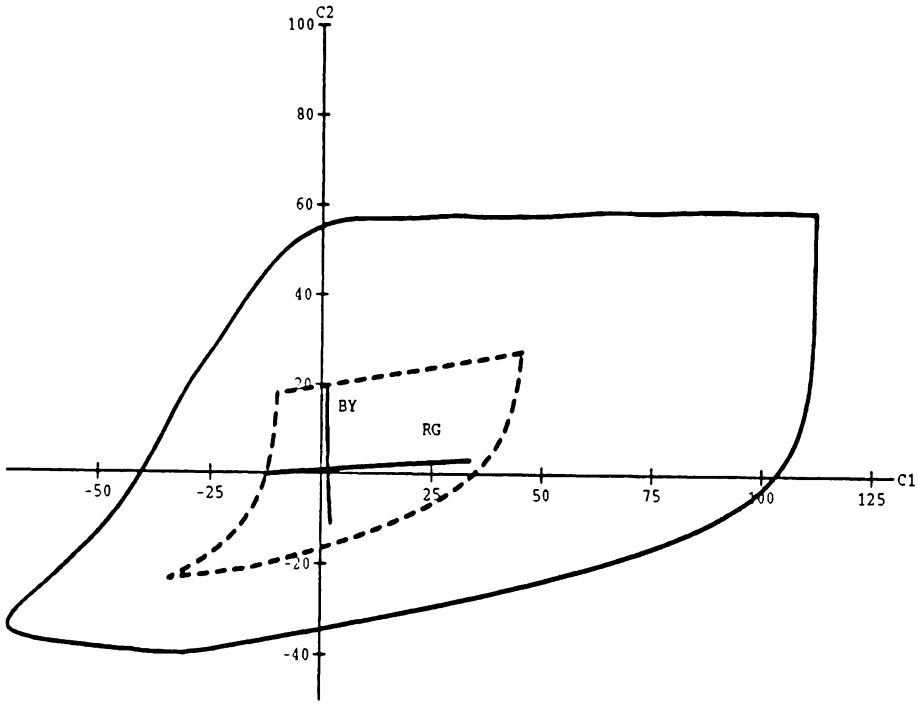


Figure 6.5: Directions of Variation in AC_1C_2 Space. White Point Chromaticity is (0.3165,0.3426). $Y = 5\ cd/m^2$.

6.3.5 Transformation to YIQ Space

The transformation to YIQ space is a linear one, so the red-green and blue-yellow lines map to straight lines as seen in Figure 6.6. The red-green line lies mostly in the inphase, I, direction, and the blue-yellow line lies mostly in the quadrature, Q, direction. The angles that these lines have been rotated through, relative to the I axis, depend on the white point about which the lines were constructed. For the

white point of (0.33,0.35), the angles for the red-green and blue-yellow directions are 34.47 and -73.26 degrees. Notice that these lines are not orthogonal to each other. Using standard illuminant C as the center changes these rotation angles by less than 2 degrees.

A new color space, designated YI^*Q^* was derived using the N.T.S.C. phosphors and the monitor's white point; the details are discussed in Chapter 2. The I^* axis was constructed from $(R-Y)/1.14$ and the Q^* axis from $(B-Y)/2.03$. There was no rotation of 33 degrees as is done in the conventional N.T.S.C. YIQ space. The directions of the red-green and blue-yellow lines can be seen in Figure 6.7. The red-green line is now along the I^* axis. The blue-yellow line is at an angle of approximately 14.4 degrees from the Q^* axis.

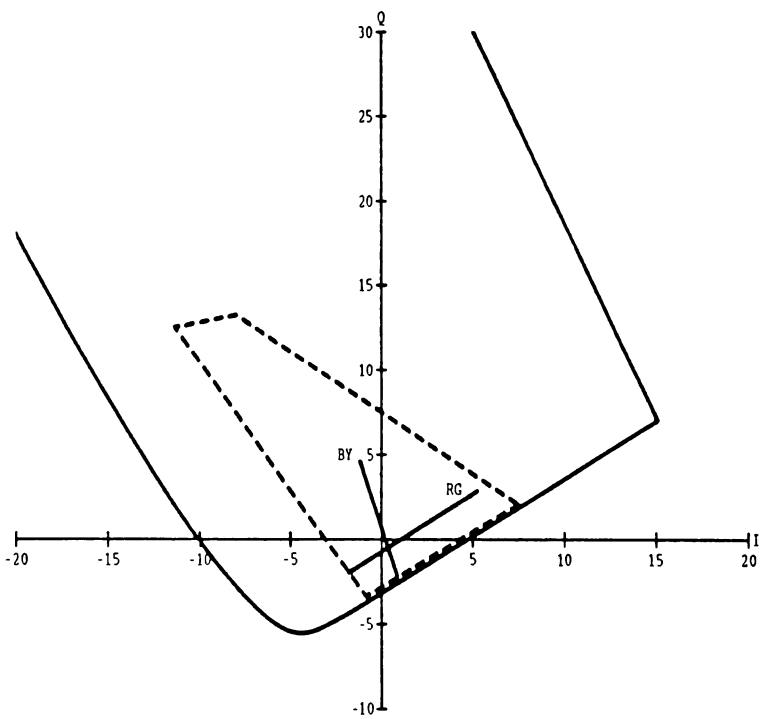


Figure 6.6: Directions of Variation in YIQ Space. $Y = 5\ cd/m^2$.

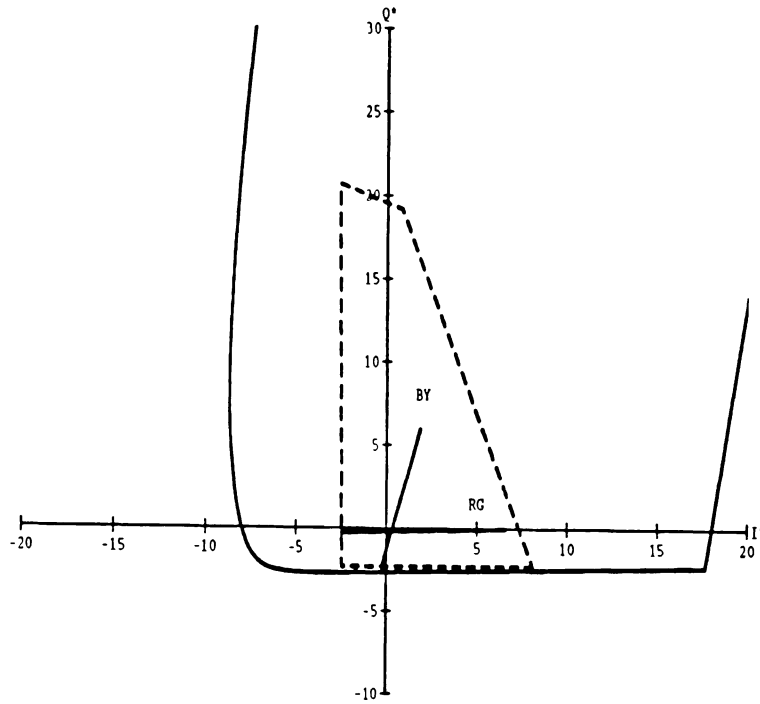


Figure 6.7: Directions of Variation in YI^*Q^* Space. $Y = 5\ cd/m^2$.

6.3.6 Example Transformation of Mean Threshold Data

The following example will illustrate how the mean threshold data was transformed from xyY to $L^*a^*b^*$ space using the monitor's white point in the transformation. Table 6.5 contains the mean threshold data in xyY space. The entries with a single asterisk by them indicate that at least one of the experimental subjects was unable to detect the pattern. A double asterisk indicates that no subjects were able to detect the pattern; the value of 150 was used. The value of 150 was selected since it was higher than any of the measured data, but low enough to allow the possibility of a non-zero bit allocation. Even though none of the subjects were able to see the pattern, other subjects possibly could. For low compression ratios, some bits should be allocated to these subband color components. This data is taken from the more

complete tables in Appendix A of this work.

Spatial Dir.	Color Dir.	Spatial Frequency cycles/deg					
		0.5	1.0	2.0	5.0	10.0	20.0
Hor.	Lum	8.167	6.750	6.333	7.250	13.500	65.083
	R-G	4.667	4.750	4.750	7.167	17.417	77.417
	B-Y	11.667	6.000	6.833	32.667	70.167	**150.000
Vert.	Lum	7.000	6.833	6.250	6.833	22.500	*77.800
	R-G	5.083	5.583	7.083	9.250	23.000	*90.375
	B-Y	11.000	6.667	9.417	31.833	*65.700	**150.000
Left	Lum	8.250	7.667	6.917	11.167	37.083	*49.000
	R-G	7.167	7.917	7.167	16.083	37.500	*100.750
	B-Y	16.000	12.417	18.500	*45.500	*86.500	**150.000
Right	Lum	8.583	8.083	7.583	9.167	42.583	*85.750
	R-G	7.250	7.750	6.333	13.833	35.417	*103.500
	B-Y	16.417	13.750	19.750	*47.750	*83.000	*114.000

Table 6.5: Mean Detection Thresholds in xyY Space. $Y = 5\ cd/m^2$. Representative chromaticity is white with chromaticity coordinates (0.33,0.35).

Consider the transitions with a horizontal orientation and a spatial frequency of 0.5 cycles/degree. The luminance transition has a mean detection threshold of 8.167 as seen in Table 6.5. The $(X_0\ Y_0\ Z_0)^T$ value for the white representative chromaticity at a luminance value of $Y = 5\ cd/m^2$ is $(4.714\ 5.000\ 4.571)^T$. Using Eq. (6.11) gives the XYZ tristimulus values of $(4.810\ 5.101\ 4.664)^T$ and $(4.619\ 4.899\ 4.479)^T$. Converting these vectors to $L^*a^*b^*$ space gives the vectors $(43.733\ 1.766\ 2.852)^T$ and $(42.932\ 1.743\ 2.814)^T$. Taking the Euclidean norm of the difference of these two vectors gives a mean detection threshold value of 0.803.

Since the two chrominance components are calculated in the same manner, we shall show how the mean detection threshold for the red-green direction is computed. The xyY threshold value is 4.667 for the horizontal orientation and spatial frequency

of 0.5 cycles/degree. The two vectors in this color space are found by using Eq. (6.8). They are $(0.333\ 0.348\ 0.319)^T$ and $(0.327\ 0.352\ 0.321)^T$. Using Eq. (6.10), one gets the XYZ tristimulus values of $(4.781\ 5.000\ 4.573)^T$ and $(4.649\ 5.000\ 4.570)^T$. Transforming these vectors to $L^*a^*b^*$ space yields the vectors $(43.335\ 2.958\ 2.819)^T$ and $(43.335\ 0.551\ 2.847)^T$. Notice that since the transition was only in chrominance, the L^* values are the same. The Euclidean norm of the difference gives a mean detection value of 2.407. Continuing this process with the other spatial frequencies and spatial directions gives Table 6.6.

Spatial Dir.	Color Dir.	Spatial Frequency cycles/deg					
		0.5	1.0	2.0	5.0	10.0	20.0
Hor.	Lum	0.803	0.663	0.622	0.712	1.327	6.426
	R-G	2.407	2.450	2.450	3.697	8.986	40.093
	B-Y	4.471	2.299	2.618	12.544	27.169	60.447
Vert.	Lum	0.688	0.671	0.614	0.671	2.212	7.698
	R-G	2.622	2.880	3.654	4.771	11.868	46.873
	B-Y	4.216	2.555	3.609	12.223	25.406	60.447
Left	Lum	0.811	0.753	0.680	1.097	3.649	4.828
	R-G	3.697	4.084	3.697	8.297	19.361	52.325
	B-Y	6.134	4.759	7.093	17.510	33.681	60.447
Right	Lum	0.843	0.794	0.745	0.901	4.193	8.498
	R-G	3.740	3.998	3.267	7.136	18.283	53.774
	B-Y	6.294	5.270	7.573	18.384	32.277	44.939

Table 6.6: Mean Detection Thresholds in $L^*a^*b^*$ Space. $Y = 5\ cd/m^2$. Representative chromaticity is white with chromaticity coordinates (0.33,0.35).

6.4 Calculation of Perceptual Weights

Once one has obtained the mean detection thresholds in the desired color space, the weights for the given subband configuration must be calculated. The visual experiments provide data at 0.5, 1.0, 2.0, 5.0, 10.0 and 20.0 cycles/degree. The experiment was designed with the subjects at seven feet from the display so that the images subtended two degrees at the eye. The mean detection threshold data must be converted from cycles/degree to cycles/inch for some standard viewing distance. A 256×256 pixel image has physical dimensions of 3×3 inches on our specific monitor. This corresponds to a sample spacing $T_s = 0.01172$ inches/pixel in both the vertical and horizontal directions, or a sampling frequency $F_s = 1/T_s = 85.33$ cycles/inch. Assuming that there was no aliasing in the original digital image implies that the image is bandlimited to a single-sided bandwidth of 42.66 cycles/inch in each direction.

A schematic of the experimental set-up is shown in Figure 6.8. Define the visual axis as the line that is perpendicular to the image and runs through its center. For a given viewing distance, let θ represent the angle between the point on the visual axis at this distance and the edge of the image. Twice this angle is the angle subtended at the eye by the entire image. Assuming that $tan(\theta) = \theta$ allows one to convert the data to cycles/inch. For example, consider a viewing distance of five times the picture height. $\theta = 5.711$ degrees, and this angle covers 1.5 inches of the image. Therefore, one degree equals 0.2627 inches.

The perceptual weights are defined to be the mean detection thresholds at the desired spatial frequencies, orientations, and color transition directions. The data for the white background at a luminance value of $Y = 5\ cd/m^2$ is used. The white

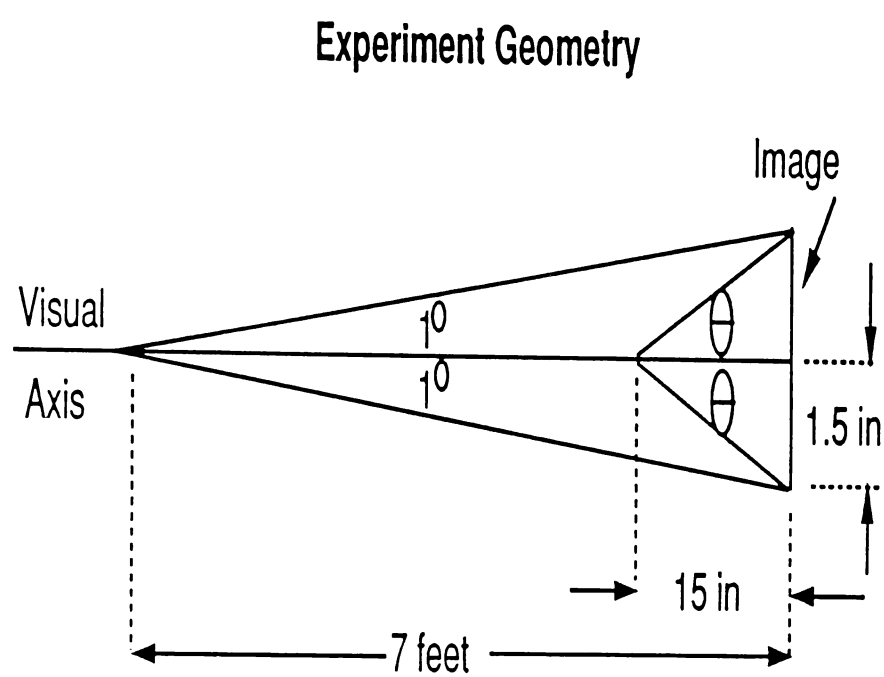


Figure 6.8: Schematic of the Mean Detection Threshold Experiment.

background was chosen since without any *a priori* knowledge of the input, one can not assume that the average background color is red, green, or blue. A neutral color such as white is a better assumption. The luminance value was chosen to be 5 cd/m^2 since the histogram of the toy store image has a peak at the lower luminance levels. The choices of background color and background luminance do not have much effect on the bit allocation. Sets of perceptual weights were derived for luminance values of $Y = 10$, and $Y = 20$ cd/m^2 for the white background. The resulting bit allocations were often the same as for the weights derived with $Y = 5$ cd/m^2. Similar results occurred using weights derived from the other background colors. See the discussion of a color adaptive system in Chapter 9 for more details.

The use of perceptual weights is an attempt to correct for the non-uniform response of the HVS. Since the mean detection threshold is the smallest change along a color

direction that can be detected, it is reasonable to directly equate the perceptual weights with this threshold. Using some function of the mean detection threshold is possible, but this is better done by calculating the weights in uniform color spaces. Higher values of the weights imply a decreased sensitivity of the human visual system. In Eq. (6.3), the perceptual weights are in the denominator. This means that the higher the value of a weight, the more the distortion for that subband is reduced. Since the total distortion is to be minimized, this implies that fewer bits will be allocated to a subband if its perceptual weight is high. This corresponds to the subband not being perceptually important.

6.4.1 Rectangular Subband Configuration

Table 6.7 contains the center frequencies of the subbands measured radially from the origin and shows their orientations. These frequencies will be the ones used to determine the perceptual weights, and they are shown in Figure 6.9. These center frequencies were converted to cycles/degree for the given viewing distance, five picture heights, and linear interpolation in the desired color space was used to obtain the mean threshold values. Since the lowest frequency subimage contains the baseband information, an average of the horizontal and vertical orientation data was used. These two orientations provided very similar thresholds for this spatial frequency, justifying the use of the average. By using this average of horizontal and vertical orientations instead of a diagonal orientation at the vector sum of them, the lowest frequency subband is given slightly more importance. Even though subbands 122 and 22 are rectangular in shape, they contain the diagonal high-frequency information. These two subbands do not make a distinction between the left and right diagonal orientations so an average of the two was used.

Subband	Center Freq. cycles/inch	Center Freq. cycles/deg.	Orientation
111	5.333	1.401	1/2 H, 1/2 V
112	16.000	4.203	V
121	16.000	4.203	H
122	22.627	5.944	1/2 L, 1/2 R
12	32.000	8.406	V
21	32.000	8.406	H
22	45.255	11.888	1/2 L, 1/2 R

Table 6.7: Subband Orientation - Rectangular Configuration. Center Frequencies in cycles/degree are for a viewing distance of 5 picture heights.

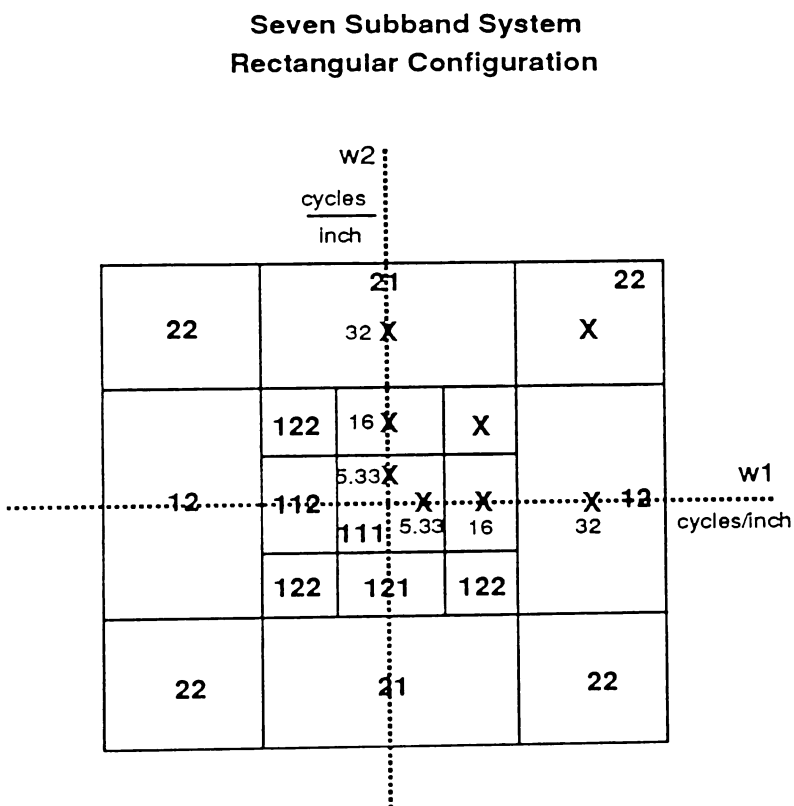


Figure 6.9: Spatial Frequency Locations of the Perceptual Weights. Rectangular Configuration. The bold numerals are the numbers of the subbands. The smaller numerals are the spatial frequencies in cycles/inch.

In some preliminary simulation results, the three transition directions were directly used to provide weights for the three color components. That is, the luminance direction was used to provide weights for the Y, L^*, and A components. The red-green direction was used for the I, a^*, and C_1 color components, and the blue-yellow direction was used for the Q, b^*, and C_2 components. This assignment method provides weights that are valid for AC_1C_2 space since, as was shown before, the red-green and blue-yellow directions are essentially along the C_1 and C_2 axes. The assumption that this method can be used for $L^*a^*b^*$ space is also reasonable since only the blue-yellow direction is not along a coordinate axis.

Two sets of perceptual weights in $L^*a^*b^*$ space, based on a viewing distance of five times the picture height, are given in Table 6.8. The first set was calculated by transforming from XYZ to $L^*a^*b^*$ space using the white point with tristimulus values $(5\ 5\ 5)^T$. The second set was transformed using the monitor's white point of $(34.51\ 37.36\ 37.16)^T$. Simulation results using these two sets will be discussed in Chapter 7. Preliminary simulations used the first set, and the final simulations used the second set. The weights for the b^* direction were also modified in the second set. Since the blue-yellow direction is not a straight line in $L^*a^*b^*$ space, it was approximated by a straight line at an angle of $-59.62°$ from the a^* axis. The weights for the transitions in the b^* direction were obtained by assuming the blue-yellow weights lie on this line, and projecting this line onto the b^* axis. The result is that these weights were multiplied by $cos(30.38°)$.

Subband	Weights			Weights		
	L^*	a^*	b^*	L^*	a^*	b^*
111	1.271	5.370	5.242	0.648	2.820	2.331
112	1.287	8.520	19.271	0.656	4.474	8.570
121	1.350	6.408	19.218	0.689	3.366	8.547
122	3.042	18.685	40.318	1.551	9.813	17.931
12	3.375	18.289	41.129	1.721	9.605	18.292
21	2.218	13.898	43.657	1.131	7.300	19.417
22	8.705	48.141	67.243	4.439	25.284	31.662

Table 6.8: $L^*a^*b^*$ Perceptual Weights - Rectangular Configuration. 5 Times Picture Height. First set uses white point $(5\ 5\ 5)^T$. Second set uses white point $(34.51\ 37.36\ 37.16)^T$.

The set of perceptual weights in AC_1C_2 space is given in Table 6.9. Table 6.10 contains the sets of weights in YIQ and YI^*Q^* spaces. The reason that the weights for the luminance components are slightly different is that the former color space uses standard illuminant C as the white point while the latter uses the monitor's white point. The weights for the Q^* direction were obtained by projecting the weights for the blue-yellow direction in YI^*Q^* space onto the Q^* axis. Since the angle between the blue-yellow direction and the Q^* axis was only 14.4°, the effect of this change was negligible, but it was done to be consistent with processing in $L^*a^*b^*$ space.

These sets of weighting factors have been used to obtain simulation results for Case 1, Case 2, and Case 3. To apply these weighting factors to Case 1, a single value is needed for the higher subbands. We have used the luminance component for the weight of each of these subbands since the luminance is more important. This gives slightly too little weight to the chrominance components of the lowest frequency subband, but the results are visually very good. Further discussion of this point will occur in the section on experimental results.

Subband	Weights		
	A	C_1	C_2
111	0.739	1.391	0.694
112	0.749	2.207	2.552
121	0.786	1.660	2.545
122	1.771	4.841	5.343
12	1.964	4.738	5.450
21	1.290	3.600	5.786
22	5.077	12.505	9.452

Table 6.9: AC_1C_2 Perceptual Weights - Rectangular Configuration. 5 Times Picture Height.

Subband	Weights			Weights		
	Y	I	Q	Y	I^*	Q^*
111	0.165	0.259	0.151	0.164	0.274	0.175
112	0.167	0.410	0.556	0.166	0.434	0.644
121	0.175	0.309	0.555	0.174	0.327	0.643
122	0.394	0.900	1.170	0.392	0.952	1.355
12	0.437	0.881	1.193	0.435	0.932	1.382
21	0.287	0.669	1.268	0.286	0.708	1.468
22	1.125	2.322	2.090	1.120	2.457	2.421

Table 6.10: YIQ Perceptual Weights - Rectangular Configuration. 5 Times Picture Height.

6.4.2 Diamond Subband Configuration

The derivation of the perceptual weights for the diamond configuration was done in a similar manner to the rectangular configuration. The radial spatial frequencies of the diamond subbands are listed in Table 6.11 and are shown in Figure 6.10. Comparison with the rectangular configuration shows that the centers of the lower subbands are at higher spatial frequencies for the diamond configuration. The differences in the mean detection thresholds at the lowest frequency are not very large, so the weights are similar. This is especially true for the luminance weights.

Subband	Center Freq. cycles/inch	Center Freq. cycles/deg.	Orientation
111	10.667	2.802	1/2 H, 1/2 V
112	22.627	5.944	L
121	22.627	5.944	R
122	32.000	8.406	1/2 H, 1/2 V
12	45.255	11.888	1/2 L, 1/2 R

Table 6.11: Subband Orientation - Diamond Configuration. Center Frequencies in cycles/degree are for a viewing distance of 5 picture heights.

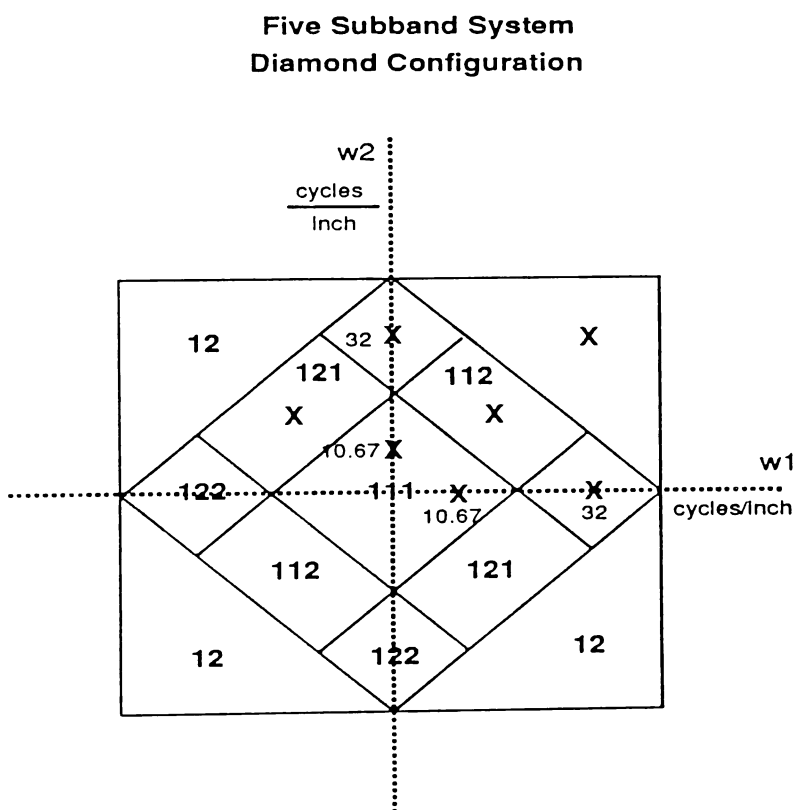


Figure 6.10: Spatial Frequency Locations of the Perceptual Weights. Diamond Configuration. The bold numerals are the numbers of the subbands. The smaller numerals are the spatial frequencies in cycles/inch.

Tables 6.12, 6.13, and 6.14 list the perceptual weights in the color spaces $L^*a^*b^*$, AC_1C_2, and YIQ, respectively. Two sets of weights for $L^*a^*b^*$ space were computed as in the rectangular configuration. The first set use the arbitrary white point with tristimulus values $(5\ 5\ 5)^T$ and the second use the monitor's white point of $(34.51\ 37.36\ 37.16)^T$. The weights for the b^* component were also reduced by $\cos(30.38°)$. Notice that the weights for subband 12 are the same as those for subband 22 in the rectangular configuration. This is true for all three color spaces.

Subband	Weights			Weights		
	L^*	a^*	b^*	L^*	a^*	b^*
111	1.251	6.413	10.847	0.638	3.368	4.824
112	3.097	19.775	39.887	1.579	10.386	17.739
121	2.986	17.594	40.748	1.522	9.241	18.123
122	2.797	16.094	42.393	1.426	8.452	18.854
12	8.705	48.141	67.243	4.439	25.284	31.662

Table 6.12: $L^*a^*b^*$ Perceptual Weights - Diamond Configuration. 5 Times Picture Height. First set uses white point $(5\ 5\ 5)^T$. Second set uses white point $(34.51\ 37.36\ 37.16)^T$.

Subband	Weights		
	A	C_1	C_2
111	0.728	1.661	1.436
112	1.803	5.124	5.286
121	1.738	4.558	5.400
122	1.627	4.169	5.618
12	5.077	12.505	9.452

Table 6.13: AC_1C_2 Perceptual Weights Diamond Configuration. 5 Times Picture Height.

Subband	Weights			Weights		
	Y	I	Q	Y	I^*	Q^*
111	0.162	0.309	0.313	0.161	0.327	0.362
112	0.401	0.952	1.157	0.399	1.008	1.341
121	0.387	0.847	1.182	0.385	0.897	1.369
122	0.362	0.775	1.231	0.361	0.820	1.425
12	1.125	2.322	2.090	1.120	2.457	2.420

Table 6.14: YIQ Perceptual Weights - Diamond Configuration. 5 Times Picture Height.

CHAPTER 7

SIMULATION RESULTS SBC/VQ RECTANGULAR CONFIGURATION

7.1 Introduction

In Chapter 4, we presented results for color subband coding with scalar quantization. The bit rate can be further reduced by exploiting the correlations among the color components of a subband, and among the pixels within a color component and subband. In this chapter, simulation results are presented for the case where three-dimensional vectors are created by combining the three color components of each subband (Case 1), and for the case where four-dimensional vectors are created from 2×2 blocks of pixels in each subband color component (Cases 2 and 3). All of the simulations use the seven-band rectangular configuration; simulation results using the five-band diamond configuration are presented in Chapter 8. A few notes on the photographic processing of the color images are given in Appendix B.

The results of the computer simulations presented in this chapter attempt to answer three questions. The first one is how much improvement in reconstructed image quality can be obtained by the use of a perceptually optimal bit allocation. The use of perceptual weights in the Marginal Analysis bit allocation method is compared to the use of uniform weights. For conciseness, we refer to these as the perceptual method and the uniform method, respectively. The second question is to determine how well the three different vector quantizer cases compare for a number

of different compression ratios. To answer this, the results will emphasize coding in $L^*a^*b^*$ space. The third question addresses the advantages and disadvantages of coding in the three different color spaces.

To answer the above three questions, it is assumed that the codebooks used for the quantizers are good in some sense. They must be designed so that at low compression ratios there are no large, noticeable color errors. The choice of training sequence plays a very important role in the quality of these quantizers. Initial simulations used codebooks that were originally designed using only the GIRL image as the training sequence. This image did not contain saturated blues and reds so the resulting codebooks could not reproduce these colors well. Subsequent simulations used new sets of codebooks designed using four different images as the training sequence. The effects of these different training sequences will be discussed.

The outline of this chapter is as follows. The next section provides some statistics on the test images such as the means, variances, and correlations among the color components and pixels for the different color spaces used. The third section compares results obtained using the two different training sequences. The sizes and types of errors in the different color spaces, resulting from the choice of training sequence, are discussed. The fourth section compares and examines the effectiveness of using the perceptual weights in the bit allocation algorithm. The bit allocation is run for different compression ratios using both the uniform weights and the perceptual weights, and the results are compared subjectively and in terms of average ΔE error. The fifth section compares the three different vector quantizer cases as a function of compression ratio. Finally, the sixth section addresses the question of which color space should be used for image compression.

7.2 Image Statistics

The use of vector quantization instead of scalar quantization leads to either an increase in compression ratio for a given image quality or to an increase in image quality for a given compression ratio. The amount of this gain is related to the amount of correlation present among the components of the vectors. Tables 7.1 and 7.2 contain some basic statistics about the GIRL and DOLL images, respectively. The most important statistic is the matrix of correlation coefficients for each color space.

The color components in XYZ and RGB spaces are highly correlated, indicating that vector quantization across the color components would yield a significant gain over scalar quantization. This also explains why the quantization images in Chapter 4 suffered so much in these color spaces for a bit allocation of four bits/pixel per component. In both YIQ and $L^*a^*b^*$ spaces, most of the energy is in the luminance component. As was seen in the quantization experiments, by allocating five bits/pixel to the luminance component and three bits/pixel to each chrominance component, the quality of the reconstructed image was greatly improved.

The advantage of Case 2 over Case 1 is also explained by the relatively low correlations among the color components in YI^*Q^*, $L^*a^*b^*$, and AC_1C_2 spaces compared to the correlations among the pixels in each spatial block. The correlation matrices for the latter two color spaces are given in Tables 7.3 and 7.4 where each table contains the correlation matrices for the 2×2 blocks of pixels in each color component. Many of the correlations are greater than 0.99. The correlations among the blocks of pixels are equally high in the other color spaces that were used.

Space	Mean	Variance	Energy	Correlation Coeffs.		
X	13.0641	179.0264	349.6974	1.0000	0.9964	0.9231
Y	13.6111	220.1268	405.3886	0.9964	1.0000	0.9347
Z	10.6652	188.7731	302.5193	0.9231	0.9347	1.0000
R	14.6296	200.4663	414.4910	1.0000	0.9657	0.8513
G	14.0452	262.6255	459.8927	0.9657	1.0000	0.9256
B	8.7274	130.4415	206.6094	0.8513	0.9256	1.0000
Y	13.6111	220.1268	405.3886	1.0000	-0.0534	-0.3700
I^*	1.6448	7.1902	9.8955	-0.0534	1.0000	-0.6155
Q^*	-1.5158	7.8127	10.1104	-0.3700	-0.6155	1.0000
L^*	51.3489	1262.6721	3899.3817	1.0000	-0.1751	0.3834
a^*	8.0883	148.5274	213.9476	-0.1751	1.0000	0.3070
b^*	10.9521	241.9961	361.9436	0.3834	0.3070	1.0000
A	33.9927	1613.4020	2768.9083	1.0000	-0.0123	0.3569
C_1	6.7264	88.5819	133.8264	-0.0123	1.0000	0.6624
C_2	3.0985	21.0861	30.6869	0.3569	0.6624	1.0000

Table 7.1: Statistics of the GIRL Image: means, variances, energies, and matrices of correlation coefficients.

Space	Mean	Variance	Energy	Correlation Coeffs.		
X	9.3652	98.8145	186.5207	1.0000	0.9899	0.8823
Y	9.3071	117.1305	203.7516	0.9899	1.0000	0.9121
Z	7.8427	99.6675	161.1747	0.8823	0.9121	1.0000
R	10.6703	119.1768	233.0315	1.0000	0.9153	0.7730
G	9.1628	138.4620	222.4196	0.9153	1.0000	0.9056
B	6.4867	69.0927	111.1703	0.7730	0.9056	1.0000
Y	9.3071	117.7516	203.7516	1.0000	0.0553	-0.3798
I^*	1.7438	9.4066	12.4475	0.0553	1.0000	-0.6032
Q^*	-0.7279	5.4981	6.0278	-0.3798	-0.6032	1.0000
L^*	45.3967	816.3034	2877.1619	1.0000	0.0879	0.4009
a^*	9.7785	352.1341	447.7530	0.0879	1.0000	0.5904
b^*	5.4472	306.7299	336.4021	0.4009	0.5904	1.0000
A	30.4338	1221.0602	2147.2735	1.0000	0.3070	0.5014
C_1	4.9609	223.3470	247.9576	0.3070	1.0000	0.8617
C_2	1.2462	38.7103	40.2632	0.5014	0.8617	1.0000

Table 7.2: Statistics of the DOLL Image: means, variances, energies, and matrices of correlation coefficients.

		Correlation Coefficients							
		GIRL				DOLL			
Pixel		1	2	3	4	1	2	3	4
L^*	1	1.0000	0.9973	0.9980	0.9952	1.0000	0.9944	0.9953	0.9900
	2	0.9973	1.0000	0.9957	0.9980	0.9944	1.0000	0.9900	0.9953
	3	0.9980	0.9957	1.0000	0.9973	0.9953	0.9900	1.0000	0.9944
	4	0.9952	0.9980	0.9973	1.0000	0.9900	0.9953	0.9944	1.0000
a^*	1	1.0000	0.8664	0.9812	0.8542	1.0000	0.9063	0.9844	0.8978
	2	0.8664	1.0000	0.8541	0.9783	0.9063	1.0000	0.8993	0.9883
	3	0.9812	0.8541	1.0000	0.8663	0.9844	0.8993	1.0000	0.9061
	4	0.8542	0.9783	0.8663	1.0000	0.8978	0.9883	0.9061	1.0000
b^*	1	1.0000	0.9741	0.9947	0.9707	1.0000	0.9848	0.9909	0.9761
	2	0.9741	1.0000	0.9696	0.9947	0.9848	1.0000	0.9776	0.9910
	3	0.9947	0.9696	1.0000	0.9744	0.9909	0.9776	1.0000	0.9848
	4	0.9707	0.9947	0.9744	1.0000	0.9761	0.9910	0.9848	1.0000

Table 7.3: Pixel Correlations in $L^*a^*b^*$ Space. For each block of four pixels, pixel 1 is the upper left, pixel 2 is the upper right, pixel 3 is the lower left, and pixel 4 is the lower right.

		Correlation Coefficients							
		GIRL				DOLL			
Pixel		1	2	3	4	1	2	3	4
A	1	1.0000	0.9956	0.9978	0.9935	1.0000	0.9935	0.9964	0.9903
	2	0.9956	1.0000	0.9938	0.9979	0.9935	1.0000	0.9901	0.9963
	3	0.9978	0.9938	1.0000	0.9956	0.9964	0.9901	1.0000	0.9935
	4	0.9935	0.9979	0.9956	1.0000	0.9903	0.9963	0.0035	1.0000
C_1	1	1.0000	0.7827	0.9797	0.7717	1.0000	0.8947	0.9876	0.8883
	2	0.7827	1.0000	0.7696	0.9760	0.8947	1.0000	0.8896	0.9907
	3	0.9797	0.7696	1.0000	0.7817	0.9876	0.8896	1.0000	0.8942
	4	0.7717	0.9760	0.7817	1.0000	0.8883	0.9907	0.8942	1.0000
C_2	1	1.0000	0.9142	0.9913	0.9088	1.0000	0.9706	0.9928	0.9650
	2	0.9142	1.0000	0.9077	0.9919	0.9706	1.0000	0.9647	0.9929
	3	0.9913	0.9077	1.0000	0.9146	0.9928	0.9647	1.0000	0.9706
	4	0.9088	0.9910	0.9146	1.0000	0.9650	0.9929	0.9706	1.0000

Table 7.4: Pixel Correlations in AC_1C_2 Space. For each block of four pixels, pixel 1 is the upper left, pixel 2 is the upper right, pixel 3 is the lower left, and pixel 4 is the lower right.

7.3 Effects of the Training Sequences

7.3.1 Color Errors

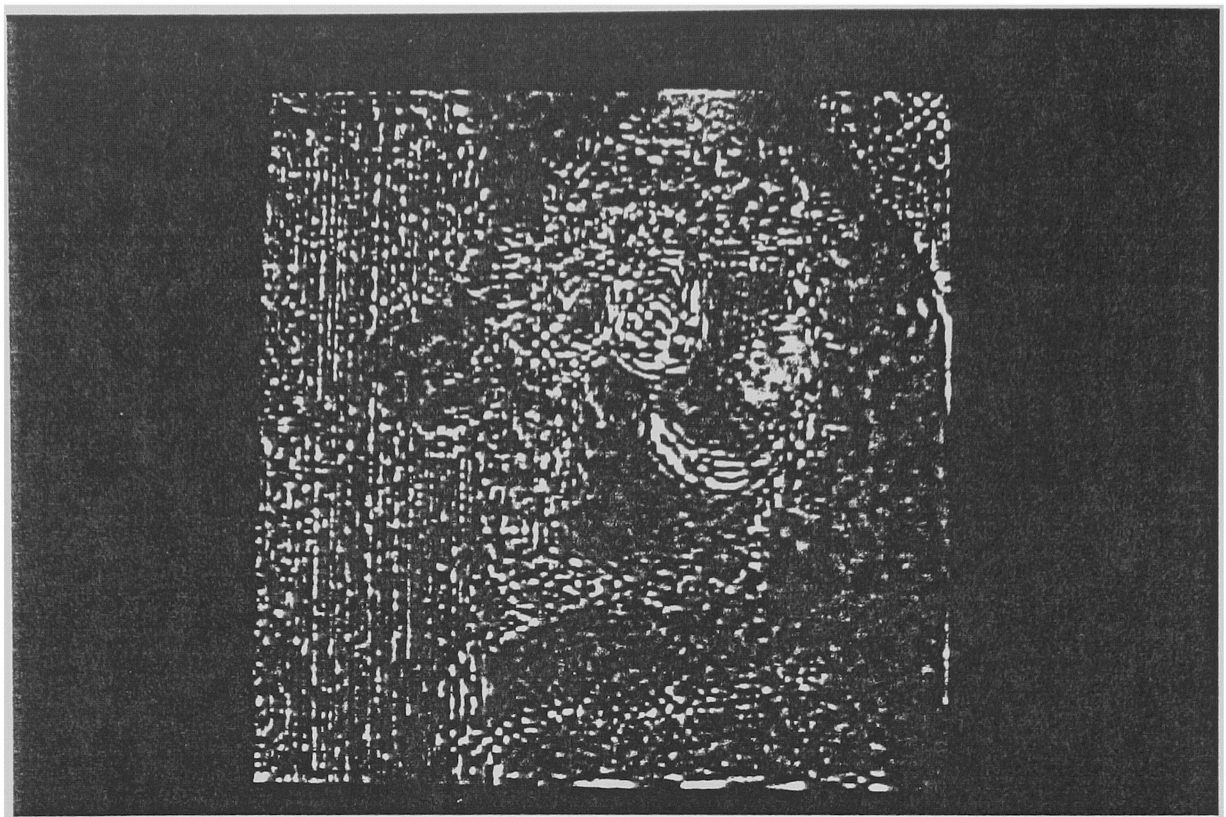
An initial set of simulations of Cases 1 and 2 were run using Training Sequence 1 (TS 1) in the design of the codebooks. This training sequence consists of only the GIRL image. Coding results for the GIRL image were quite good since the codebooks were optimized for this specific image. The DOLL image was outside the training set, and suffered from noticeable color errors. This is due to the fact that the codebooks were unable to reproduce the vivid blues and reds in the image. A final set of simulations were run using a different training sequence, Training Sequence 2 (TS 2). This consisted of four 256×256 images taken from the toy store image. Now, both the GIRL and DOLL images are outside of the training sequence. Even though this could lead to noticeably worse results for the GIRL image, the training images were chosen so that both flesh tones and saturated colors are represented.

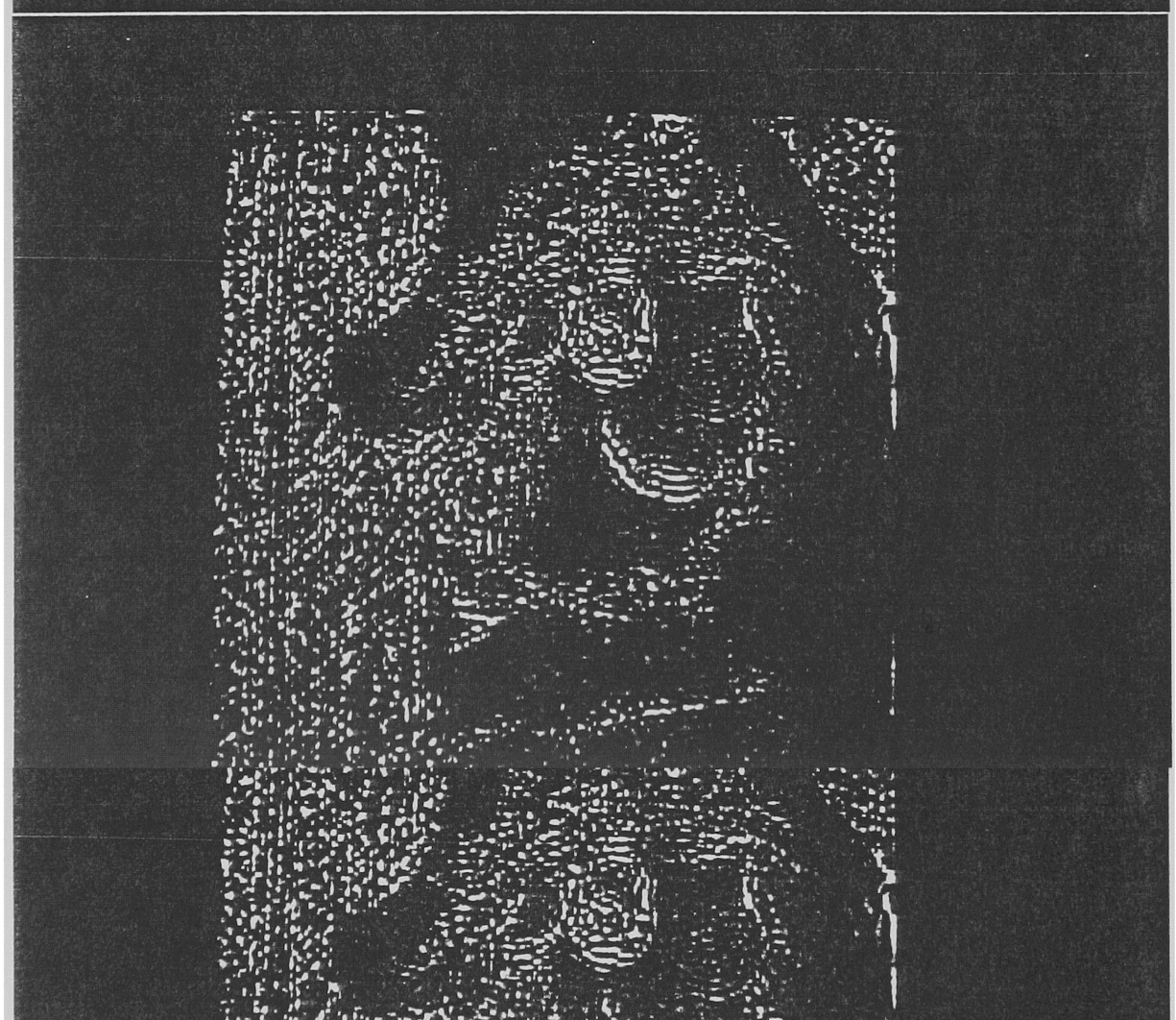
Figures 4.4 and 4.5 at the end of Chapter 4 show photographs of the DOLL image in RGB, YIQ, and $L^*a^*b^*$ spaces. The compression ratio is 8:1, and scalar quantization was used for each subband component. Training Sequence 1 was used to design the quantizers. In all three of these images, the bright blues are desaturated and the reds are somewhat darker. This is especially true for the YIQ and $L^*a^*b^*$ space images. Using vector quantizers instead of scalar quantizers does not change these results. The images coded in YIQ and $L^*a^*b^*$ spaces suffer from the same types of color errors. The reconstructed image in AC_1C_2 space, however, did not suffer from as much desaturation. Figure 7.1 shows the GIRL and DOLL images after coding in this space. There is a slight bit of contouring in the constant gray background in all three color spaces. This contouring is practically eliminated in the Case 2 results.

It is useful to examine error images to determine where in the image these errors occur, and to see how large they are. To do this, the output image is subtracted from the original image, and the results are multiplied by a factor of ten. This operation is done in the display RGB space. In this space, each color component is stored with 8 bits/pixel, so the colors can take on values from zero to 255. A value of 128 is added to each color component of every pixel, so that both positive and negative errors can be displayed. An error of zero will now be seen as a mid-level gray. Figure 7.2 shows error images for the DOLL image after coding in YIQ and YI^*Q^* spaces. The YIQ image uses TS 1 while the YI^*Q^* image uses TS 2. One can see that the former image has much larger color errors. Notice also, that the directions of the color errors are different. This is due to the fact that the chrominance axes have been rotated by 33° for YIQ space, while this was not done for YI^*Q^* space.

7.3.2 Bit Allocation

In the preliminary set of simulations using TS 1, the maximum number of bits allowed to the lowest frequency subband was limited to seven bits/pixel. The reasoning behind this is that the original image in Display RGB space has only eight bits/pixel per component, so seven bits was a reasonable number to allow good results. However, the bit allocation often allocated all seven bits. The second set of codebooks, designed using TS 2, allowed a maximum of eights bits/pixel per component for this subband. Tables 7.5 and 7.6 show the bit allocations in the different color spaces for the two different training sequences. The compression ratio is 8:1, and both Case 1 and Case 2 are included. Notice that in AC_1C_2 space, the C_2 component has fewer bits than either the Q or b^* components.





	Case 1			Case 2		
Subband	YIQ bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix	YIQ bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix
111	7,7,7	7,5,5	7,5,5	7,7,7	7,7,7	7,7,6
112	8	8	7	9,9,9	9,9,8	9,9,6
121	8	6	4	9,9,9	9,9,0	9,9,0
122	3	1	0	8,2,0	0,0,0	2,0,0
12	2	4	5	2,5,0	5,8,3	6,8,3
21	0	0	0	4,0,0	0,0,0	0,0,0
22	0	0	0	0,0,0	0,0,0	0,0,0

Table 7.5: Bit Allocation - Perceptual Weighting. 8:1 Compression Ratio. Five times Picture Height. TS 1.

	Case 1			Case 2		
Subband	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix
111	8,6,7	7,5,5	7,6,5	8,7,8	8,7,7	8,7,6
112	8	8	8	9,9,8	9,9,6	9,9,5
121	8	7	6	9,9,8	9,9,3	9,9,0
122	3	0	0	8,0,0	7,0,0	7,4,0
12	2	4	4	4,6,0	4,7,2	6,8,0
21	0	0	0	0,0,0	0,0,0	0,0,0
22	0	0	0	0,0,0	0,0,0	0,0,0

Table 7.6: Bit Allocation - Perceptual Weighting. 8:1 Compression Ratio. Five times Picture Height. TS 2.

Tables 7.7 and 7.8 contain the distortion numbers for the two training sequences. Notice that the values for the GIRL image are lower using TS 1, while the values for the DOLL image are lower using TS 2. More importantly, both images are of very good quality for the TS 2 simulations. The use of perceptual weighting increases the number of bits allocated to the lower frequency subbands and decreases the number allocated to the higher frequency ones. For Case 2, the luminance component is allocated more bits at the expense of the chrominance components. This does not

lead to very significant visual changes at this compression ratio, but becomes more important at higher compression ratios. This topic will be discussed in the next section.

Color Space	Case	Image	Δ E (ave)	$\sigma_{\Delta E}$
YIQ	1	GIRL	2.467	2.368
	1	DOLL	3.355	3.274
$L^*a^*b^*$	1	GIRL	1.795	1.382
	1	DOLL	3.297	3.327
AC_1C_2	1	GIRL	2.237	1.507
	1	DOLL	3.644	2.967
YIQ	2	GIRL	3.142	3.345
	2	DOLL	4.115	4.126
$L^*a^*b^*$	2	GIRL	1.649	1.222
	2	DOLL	2.809	2.454
AC_1C_2	2	GIRL	1.871	1.363
	2	DOLL	2.959	2.243

Table 7.7: Distortion - Perceptual Weighting. 8:1 Compression Ratio. Five times Picture Height. TS 1.

7.4 Effect of the Perceptual Weights

The first set of experiments examine the differences between coding with the uniform weights and the perceptual weights. At compression ratios of 4:1 and 8:1, the visual differences between the two methods is very small. Both methods provide enough bits to the lowest frequency subband; the allocation of enough bits to this subband is essential for high quality reproduction. The luminance component receives at least six bits/pixel, and this is enough to prevent contouring. The chrominance components each receive at least five bits/pixel. At a compression ratio of 8:1, the differences in bit allocations between the two weighting methods is apparent for Case 1. In

Color Space	Case	Image	Δ E (ave)	$\sigma_{\Delta E}$
YI^*Q^*	1	GIRL	2.949	2.780
	1	DOLL	3.197	2.817
$L^*a^*b^*$	1	GIRL	2.201	2.027
	1	DOLL	2.287	1.716
AC_1C_2	1	GIRL	2.420	1.906
	1	DOLL	2.717	1.998
YI^*Q^*	2	GIRL	2.548	2.403
	2	DOLL	2.877	2.601
$L^*a^*b^*$	2	GIRL	1.992	1.574
	2	DOLL	2.194	1.659
AC_1C_2	2	GIRL	2.355	1.834
	2	DOLL	2.799	2.120

Table 7.8: Distortion - Perceptual Weighting. 8:1 Compression Ratio. Five times Picture Height. TS 2.

$L^*a^*b^*$ space, the uniform weighting gives the lowest frequency luminance component six bits/pixel compared to seven for the perceptual weighting. There is a slight bit of contouring in the GIRL image for the former method. Both methods yield high quality results for Case 2.

The differences between the two methods can be more easily seen at compression ratios of 12:1 and higher. For the 12:1 compression ratio, the uniform weighting bit allocations are given in Table 7.9, and the corresponding bit allocations using perceptual weighting are in Table 7.10. The Case 1 allocations for the two methods are similar. The perceptual method adds slightly more bits to the lower frequency subbands at the expense of the higher frequency subbands. Because of the extra degrees of freedom in Case 2, the bit allocations are more different. By reducing the number of bits given to the three highest frequency subbands, the perceptual method can allocate more bits to all three color components of the four lower frequency subbands. Tables 7.11 and 7.12 contain the uniform and perceptual bit allocations

	Case 1			Case 2		
Subband	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix
111	7,6,6	6,5,5	6,5,4	7,6,6	6,5,6	7,6,4
112	6	5	5	9,9,5	5,8,4	9,9,3
121	6	3	4	9,0,0	6,5,0	8,3,0
122	1	0	0	0,0,0	0,0,0	0,0,0
12	0	2	2	0,5,0	0,6,2	0,7,0
21	0	0	0	0,0,0	0,0,0	0,0,0
22	0	0	0	0,0,0	0,0,0	0,0,0

Table 7.9: Bit Allocation Uniform Weighting. 12:1 Compression Ratio. TS 2.

	Case 1			Case 2		
Subband	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix
111	7,6,6	6,5,5	7,5,5	8,6,7	7,6,6	7,6,5
112	7	7	7	9,7,3	9,9,3	9,9,3
121	6	5	4	9,8,0	9,6,0	9,2,0
122	0	0	0	0,0,0	0,0,0	0,0,0
12	0	1	1	0,2,0	0,4,0	0,6,0
21	0	0	0	0,0,0	0,0,0	0,0,0
22	0	0	0	0,0,0	0,0,0	0,0,0

Table 7.10: Bit Allocation - Perceptual Weighting. 12:1 Compression Ratio. 5 Times Picture Height. TS 2.

for a compression ratio of 16:1. Notice that for Case 2, two extra bits/pixel are given to the luminance component of the lowest frequency subband in $L^*a^*b^*$ space.

A close examination of the Case 2 images at a compression ratio of 12:1 shows a noticeable improvement in the perceptually weighted images. This is especially true for the GIRL image, where the mottled appearance of the forehead is significantly decreased. Table 7.13 contains the average ΔE errors for the uniform method and Table 7.14 contains the errors for the perceptual method. While the use of this

	Case 1			Case 2		
Subband	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix
111	6,5,5	5,5,5	6,5,4	7,5,6	5,5,5	6,5,4
112	4	5	5	9,6,0	0,9,3	5,6,0
121	4	0	0	9,0,0	4,0,0	5,0,0
122	0	0	0	0,0,0	0,0,0	0,0,0
12	0	1	1	0,0,0	0,5,0	0,5,0
21	0	0	0	0,0,0	0,0,0	0,0,0
22	0	0	0	0,0,0	0,0,0	0,0,0

Table 7.11: Bit Allocation - Uniform Weighting. 16:1 Compression Ratio. TS 2.

	Case 1			Case 2		
Subband	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix
111	6,5,5	6,4,5	6,5,4	7,5,6	7,5,5	7,5,5
112	4	6	6	9,6,0	9,7,3	9,7,3
121	4	3	3	9,0,0	9,0,0	9,0,0
122	0	0	0	0,0,0	0,0,0	0,0,0
12	0	0	0	0,0,0	0,0,0	0,0,0
21	0	0	0	0,0,0	0,0,0	0,0,0
22	0	0	0	0,0,0	0,0,0	0,0,0

Table 7.12: Bit Allocation - Perceptual Weighting. 16:1 Compression Ratio. TS 2.

statistic only indicates the accuracy of the color reproduction, it still shows that the errors using the perceptual method are less.

These differences are accentuated at a compression ratio of 16:1. Figures 7.3 and 7.4 show the reconstructed GIRL and DOLL images after coding in $L^*a^*b^*$ space. Each figure contains photographs of the images resulting from the two different methods of bit allocation. The uniform method images are clearly degraded, while the perceptual ones do not look much different from those at compression ratios of 8:1 and 12:1. There is a slight loss of high frequency detail noticeable when compared to

Color Space	Case	Image	Δ E (ave)	$\sigma_{\Delta E}$
YI^*Q^*	1	GIRL	4.083	4.169
	1	DOLL	4.412	3.964
$L^*a^*b^*$	1	GIRL	2.920	2.448
	1	DOLL	3.022	2.159
AC_1C_2	1	GIRL	3.518	2.428
	1	DOLL	3.798	2.580
YI^*Q^*	2	GIRL	3.319	2.957
	2	DOLL	3.783	3.285
$L^*a^*b^*$	2	GIRL	2.450	1.776
	2	DOLL	2.723	1.860
AC_1C_2	2	GIRL	3.052	2.002
	2	DOLL	3.488	2.361

Table 7.13: Distortion - Uniform Weighting. 12:1 Compression Ratio. TS 2.

Color Space	Case	Image	Δ E (ave)	$\sigma_{\Delta E}$
YI^*Q^*	1	GIRL	3.820	3.651
	1	DOLL	4.190	3.599
$L^*a^*b^*$	1	GIRL	3.206	2.720
	1	DOLL	3.239	2.469
AC_1C_2	1	GIRL	4.183	2.806
	1	DOLL	4.083	2.816
YI^*Q^*	2	GIRL	3.579	3.440
	2	DOLL	4.188	3.747
$L^*a^*b^*$	2	GIRL	2.633	2.147
	2	DOLL	2.857	2.042
AC_1C_2	2	GIRL	2.840	1.974
	2	DOLL	3.276	2.284

Table 7.14: Distortion - Perceptual Weighting. 12:1 Compression Ratio. Five times Picture Height. TS 2.

the original.

7.5 Comparison of the Different VQ Cases

The second set of experiments determine the relative quality of the reconstructed images as a function of compression ratio. The ranges where each case provides good quality results are noted. Perceptual weighting was used in all three cases. Figures 7.5 and 7.6 show the plots of the average ΔE distortion as a function of compression ratio for the GIRL and DOLL images coded in $L^*a^*b^*$ space, respectively. One can see that the image quality slowly degrades as the compression ratio is increased. This change in quality is fairly constant over a wide range of compression ratios. Cases 1 and 2 yield similar quality images for the lower compression ratios, but Case 2 provides better results at and above a compression ratio of 12:1. At still higher compression ratios, i.e. 20:1 and greater, almost all of the bits are allocated to the lowest frequency subband. Cases 1 and 2 become the same, since both cases scalar quantize the lowest frequency subband.

Figure 7.7 shows the GIRL and DOLL images coded in $L^*a^*b^*$ space using Case 2 at a compression ratio of 12:1. These images are very good. There is a slight loss of high frequency detail, but there are no annoying color artifacts or contouring. Figure 7.8 shows the GIRL and DOLL images in the same color space for Case 1 at a compression ratio of 16:1. Comparing these two images to those in Figures 7.3 and 7.4, one sees that the latter images are better. Even though the Case 1 images look good, there is more contouring in them. Since it is possible in Case 2 to allocate bits only to the high frequency luminance components without allocating any to the chrominance, there is more high frequency detail in Case 2.

In Case 3, the chrominance components of the lowest frequency subband are vector quantized. The maximum number of bits allowed for the chrominance components

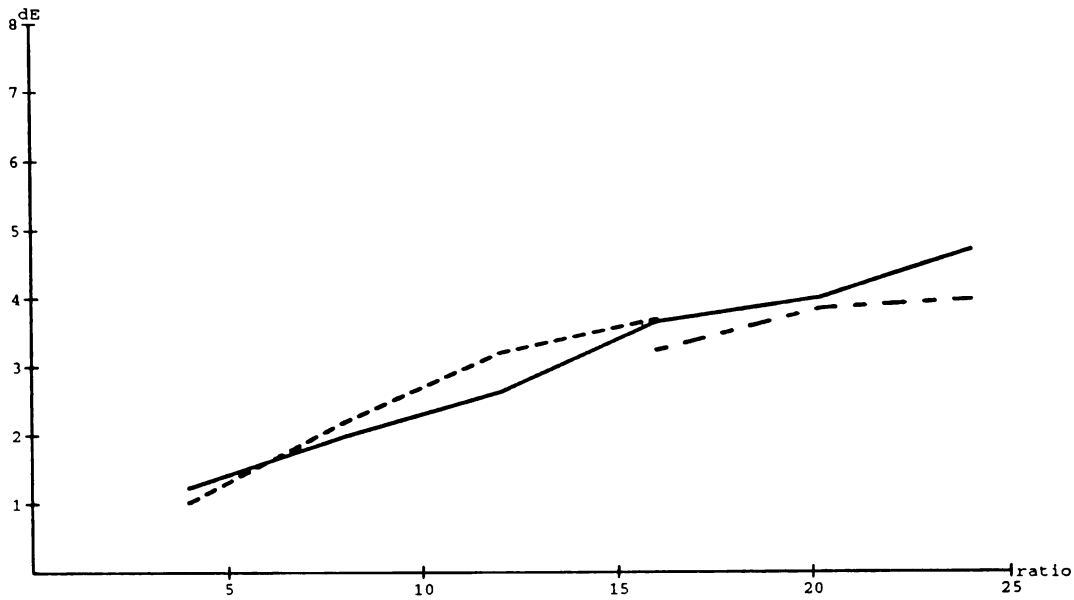


Figure 7.5: Average ΔE Distortion of the GIRL Image in $L^*a^*b^*$ Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.

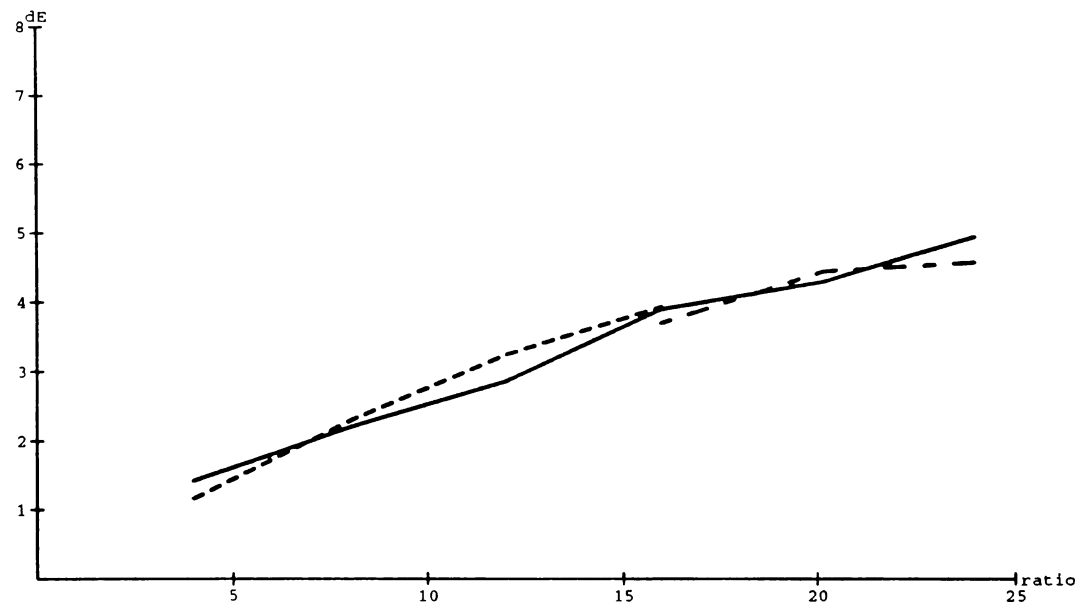
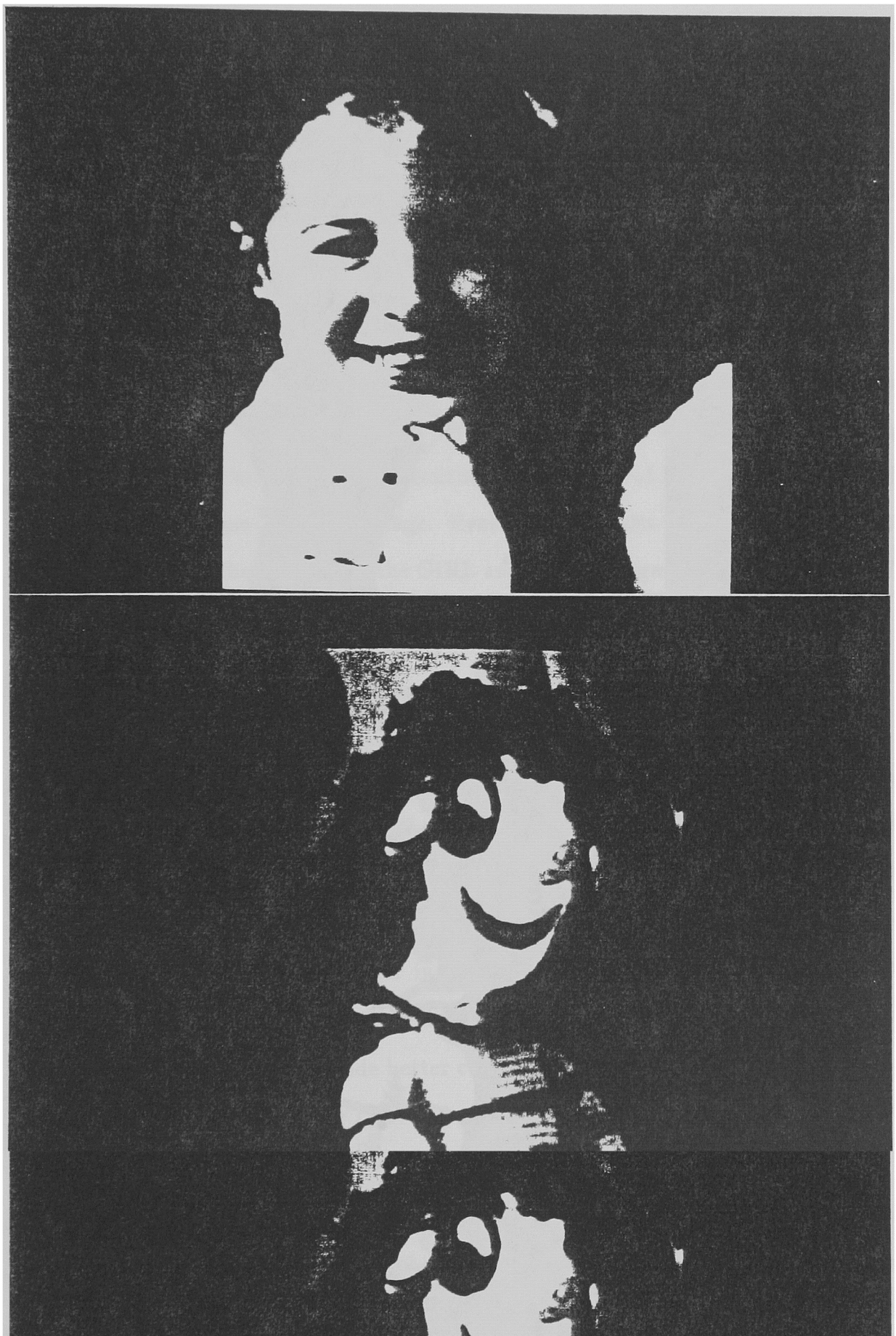


Figure 7.6: Average ΔE Distortion of the DOLL Image in $L^*a^*b^*$ Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.



of this subband is set at 12. This is done for two reasons. The first is to limit the computational complexity of the coding system. The second reason was that at 11 and 12 bits/vector, the LBG algorithm did not use the entire number of codevectors allowed. This is mainly due to the size of the training sequence, and to the value of the perturbation used in the splitting. Therefore, Case 3 was only run for compression ratios of 16:1 and greater. Even at 16:1, the bit allocation algorithm gave all 12 bits/vector to each chrominance component.

For the GIRL image, Figure 7.5 shows that Case 3 provides better results than Case 2. This was most visible at a compression ratio of 20:1. The DOLL image contains many saturated colors. Limiting the number of bits/vector lessened the effectiveness of Case 3 for this image. Even so, the results were still slightly better than for Case 2. Photographs of the GIRL and DOLL images are shown in Figure 7.9 for Case 3 at a compression ratio of 20:1.

7.6 Choice of Color Space

The last question to be answered is which color space should be used for processing. YI^*Q^* packs most of the energy in the luminance component. The perceptual weights ensure that sufficient bits are given to this component to eliminate contouring. In this respect, this color space is better than the other two. However, for highly saturated colors, the color errors are higher in this space. This is seen both in the higher average ΔE errors for this space, and by examining the reconstructed images. Both $L^*a^*b^*$ and AC_1C_2 space provide better color fidelity. Figures 7.14 and 7.15 show the GIRL and DOLL images in YI^*Q^* and AC_1C_2 spaces at a compression ratio of 24:1 using Case 3. The yellow shirt of the clown is slightly desaturated in YI^*Q^* space.

However, the forehead of the girl is quite smooth.

The relationship between image quality and compression ratio is similar for both AC_1C_2 and YI^*Q^* spaces. Plots of the average ΔE error as a function of compression ratio are given in Figures 7.10 and 7.11 for the GIRL and DOLL images coded in AC_1C_2 space. The plots for the images coded in YI^*Q^* space are given in Figures 7.12 and 7.13. The average ΔE errors are higher in both of these color spaces as compared to $L^*a^*b^*$ space, but the trends are the same. Notice that in AC_1C_2 space, the average ΔE error actually decreases for the GIRL image when changing from a compression ratio of 12:1 to 16:1 for Case 1. Visually, the two images are indistinguishable. What is interesting here, is that a significant improvement can be obtained by using Case 2 instead of Case 1. Of the three color spaces examined, AC_1C_2 usually needed the fewest bits for the blue-yellow component. Case 2 allows one to reduce the number of bits given to this component and use them elsewhere.

The choice of the best color space to use depends upon the compression ratio desired and on the amount of color errors that can be tolerated. Similarly to conventional television, if some color error is acceptable in saturated colors, then YI^*Q^* (or YIQ) space should be used. At compression ratios over 20:1, the inverse transformation from AC_1C_2 space to XYZ space can accentuate the quantization errors and cause visible artifacts. These artifacts are seen as areas of color errors, especially in white areas of the image. For this reason, AC_1C_2 is probably not the best space to use in order to achieve high compression. $L^*a^*b^*$ space is a good compromise. It has better color fidelity than YI^*Q^* space, but it does not suffer as much from visible artifacts as does AC_1C_2 space.

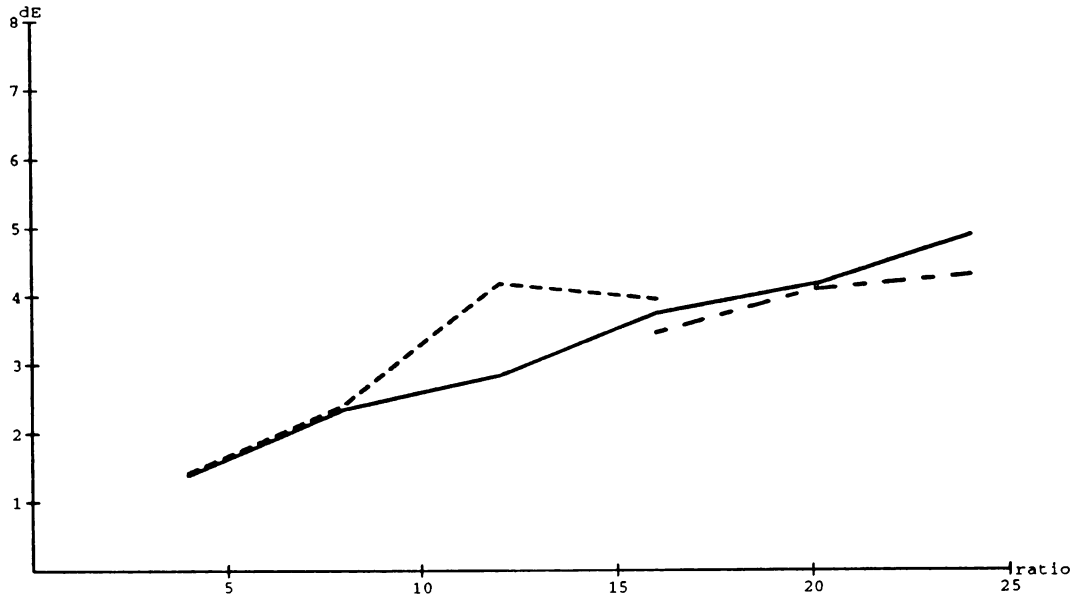


Figure 7.10: Average ΔE Distortion of the GIRL Image in AC_1C_2 Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.

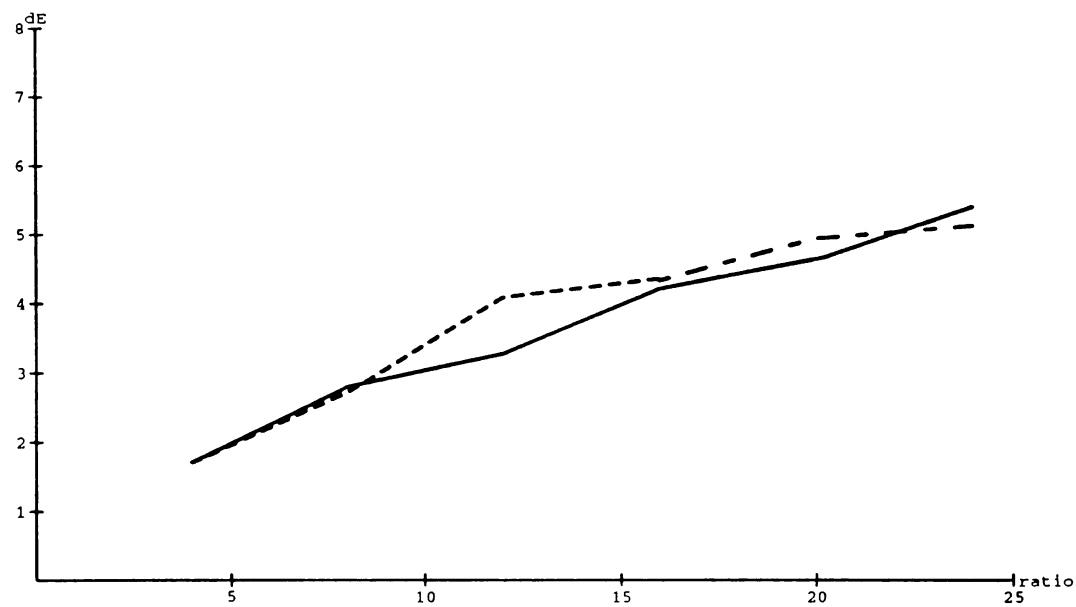


Figure 7.11: Average ΔE Distortion of the DOLL Image in AC_1C_2 Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.

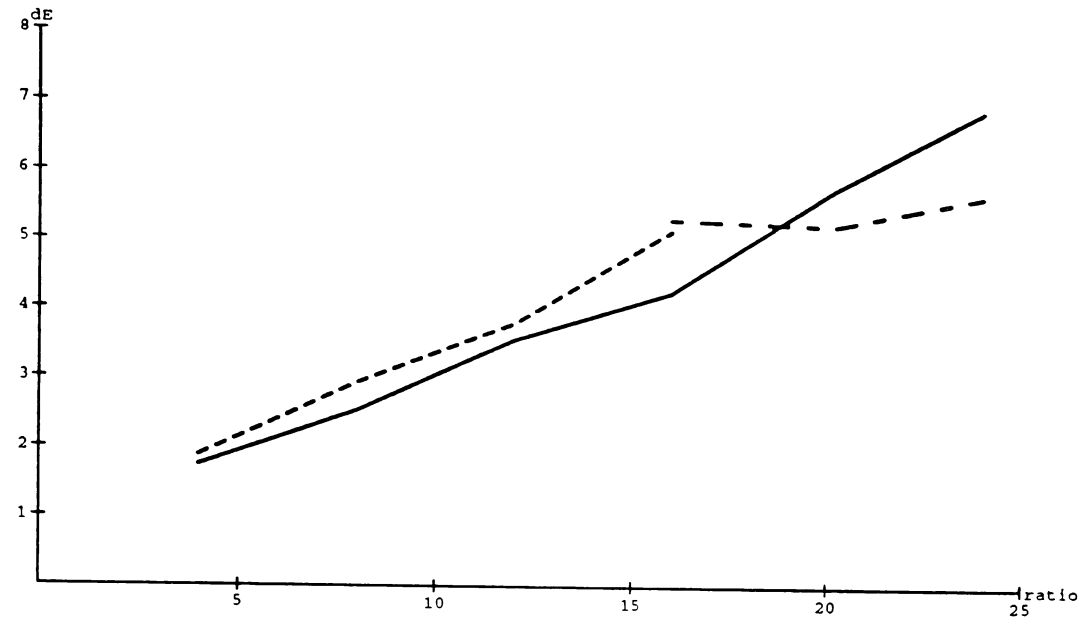


Figure 7.12: Average ΔE Distortion of the GIRL Image in YI^*Q^* Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.

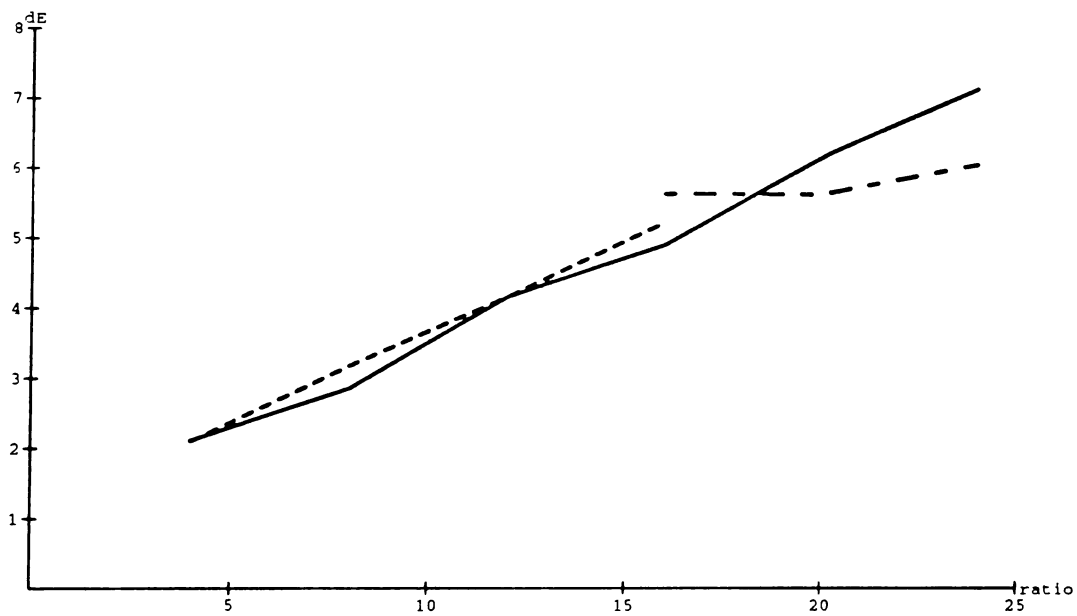


Figure 7.13: Average ΔE Distortion of the DOLL Image in YI^*Q^* Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.

CHAPTER 8

SIMULATION RESULTS SBC/VQ DIAMOND CONFIGURATION

8.1 Introduction

In Chapter 7, simulation results were presented for a seven-band rectangular configuration. That system is capable of achieving high quality images at compression ratios of 12:1 and 16:1. Additionally, the implementation of that configuration is separable, using only one-dimensional quadrature mirror filters for the subband decomposition. Since the HVS is more sensitive to horizontal or vertical patterns than to diagonal ones, it should be possible to exploit this property to improve the quality of the reconstructed images. To provide subbands that are diamond-shaped while still allowing a separable implementation, a new diamond subband coder was derived in Chapter 3. Simulation results of this configuration are presented in this chapter and were also given in [97]. All simulations use TS 2 as the training sequence for the quantizer design.

Since the diamond subband configuration splits the input image into five subbands instead of seven, there are fewer degrees of freedom available to the bit allocation algorithm. The orientation of the subbands in a manner that more closely matches the orientation of the human visual system should compensate this to some extent. The goal of this set of experiments it to determine if this improvement is sufficiently great to justify the use of this configuration. The use of perceptual weighting plays a

role in determining the acceptability of the diamond configuration.

The lowest frequency subband has $(128 \times 128)/2$ pixels in it compared to 64×64 in the rectangular configuration. Consequently, the coding of the lowest subband is even more important. The use of scalar quantization for this subband is one of the main reasons that the diamond coder does not perform as well as the rectangular coder for Cases 1 and 2. While the Case 3 results are still inferior to those of the rectangular configuration, a substantial improvement is achieved over Cases 1 and 2. The results at compression ratios of 12:1 and 16:1 become acceptable, and the results at 20:1 are not as visibly annoying, as will be seen below.

There are four sections in the rest of the chapter. The next section examines the effect of the perceptual weights, and determines their utility for the diamond configuration. The third section compares the three different vector quantizer cases as a function of compression ratio. The fourth section then compares the coding in the three different color spaces, and makes suggestions as to which color space should be used. The last section provides a short summary of the chapter and compares the diamond configuration to the rectangular configuration.

8.2 Effect of the Perceptual Weights

Since there are fewer subbands, the increase in performance due to the perceptual weights is smaller than for the rectangular configuration. At a compression ratio of 8:1, the bit allocations for the two different sets of weights are essentially the same for the color spaces YI^*Q^* and $L^*a^*b^*$ for Case 1. This is seen by comparing Tables 8.1 and 8.2 which contain the uniformly and perceptually weighted bit allocations, respectively. The Case 1 results for AC_1C_2 space are more different. The use of the

perceptual weights reduces the contouring and leads to a visible improvement.

The use of the perceptual weights improves the subjective quality of the reconstructed images for Case 2. In all three color spaces, there are now sufficient bits allocated to the lowest frequency luminance component to prevent contouring. There are also more bits allocated to the higher frequency luminance components. This makes the images slightly "crisper". The results are all subjectively pretty good at this compression ratio, so the use of the perceptual weights can not improve the quality by a large amount.

Subband	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix
111	7,5,6	6,5,5	6,5,4	7,6,6	6,5,6	7,6,4
112	1	1	1	8,0,0	0,7,0	0,3,0
121	3	2	2	0,3,0	0,6,0	0,4,0
122	2	5	6	0,9,0	0,9,6	0,9,0
12	0	0	0	0,0,0	0,0,0	0,3,0

Table 8.1: Uniform Bit Allocation. 8:1 Compression Ratio. First Set-Case 1. Second Set-Case 2.

Subband	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix
111	7,6,6	6,5,5	7,5,4	7,7,6	7,6,6	8,6,6
112	0	1	0	8,0,0	5,0,0	8,0,0
121	3	2	2	8,0,0	3,3,0	8,0,0
122	2	5	6	0,0,0	0,9,0	0,0,0
12	0	0	0	0,0,0	0,0,0	0,0,0

Table 8.2: Perceptual Bit Allocation. 8:1 Compression Ratio. Five times Picture Height. First Set-Case 1. Second Set-Case 2.

Tables 8.3 and 8.4 contain the bit allocations resulting from the uniform and perceptual methods for a compression ratio of 12:1. The GIRL image has a mottled appearance in the uniformly weighted results, but this artifact is essentially eliminated in Case 2 by using the perceptual weights. In $L^*a^*b^*$ space, there is not much improvement gained by using the perceptual weights for Case 1; the quality of the reconstructed image is not acceptable for either method. The other two color spaces provide better quality results for the GIRL image.

Subband	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix
111	6,5,5	5,4,5	5,4,3	6,5,5	5,4,5	6,5,3
112	0	0	0	0,0,0	0,0,0	0,0,0
121	0	0	0	0,0,0	0,0,0	0,0,0
122	0	2	4	0,0,0	0,8,0	0,8,0
12	0	0	0	0,0,0	0,0,0	0,0,0

Table 8.3: Uniform Bit Allocation. 12:1 Compression Ratio. First Set-Case 1. Second Set-Case 2.

Subband	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix	YI^*Q^* bits/pix	$L^*a^*b^*$ bits/pix	AC_1C_2 bits/pix
111	7,5,4	5,4,4	6,4,3	7,5,4	6,5,4	6,5,4
112	0	0	0	0,0,0	0,0,0	0,0,0
121	0	0	0	0,0,0	0,0,0	0,0,0
122	0	3	3	0,0,0	0,4,0	0,4,0
12	0	0	0	0,0,0	0,0,0	0,0,0

Table 8.4: Perceptual Bit Allocation. 12:1 Compression Ratio. Five times Picture Height. First Set-Case 1. Second Set-Case 2.

For the DOLL image, there is a trade-off between contouring and desaturation of the red and blue regions. The uniform method provides slightly less desaturation

for the vivid colors, while the perceptual method reduces the contouring in the background and other single-colored areas. The reduction in the coding artifacts seems to be more important, so the perceptual method should be used. For the rest of this chapter, the bit allocations for all simulations use the perceptual weights.

8.3 Comparison of the VQ Cases

When the compression ratio is increased sufficiently, all of the bits are allocated to the lowest frequency subband, and Cases 1 and 2 become the same. In $L^*a^*b^*$ and AC_1C_2 spaces, this happens at a compression ratio of 16:1. The bit allocation in YI^*Q^* space tends to place a higher percentage of the bits in the lowest frequency subband. This results in the merging of Case 1 and Case 2 at a 12:1 compression ratio. The average ΔE error as a function of compression ratio is given in Figures 8.1 and 8.2 for the GIRL and DOLL images coded in $L^*a^*b^*$ space. Notice that the numbers are slightly higher for Case 2 than for Case 1. This is the opposite of the rectangular configuration. However, the image quality is still subjectively better for Case 2.

The use of vector quantization to code the chrominance components of the lowest subband has an even larger effect than it does in the rectangular configuration because of the larger size of this subband. The Case 3 results are better than the other two cases for compression ratios as low as 12:1, and this is true for both the GIRL and the DOLL images. Figure 8.3 shows the GIRL and DOLL images coded in $L^*a^*b^*$ space at a compression ratio of 12:1 for Case 3.

Comparing Figure 8.3 to Figure 7.7 in the previous chapter, one sees that the rectangular configuration yields better results. Although the former image does not

look too bad, it is not as sharp and suffers from more contouring and color errors. Figure 8.4 shows the images at a compression ratio of 16:1 for Case 3. These images suffer from ringing along the diagonal edges, and have greater color errors. However, the Case 1 and Case 2 results at this compression ratio contain much more visible distortion and are not acceptable.

8.4 Choice of Color Space

Figures 8.5 and 8.6 show the average ΔE distortion as a function of compression ratio for the GIRL and DOLL images coded in AC_1C_2 space. The same functions for YI^*Q^* space are shown in Figures 8.7 and 8.8. Comparing these four figures to Figures 8.1 and 8.2, one sees that the distortion is always lowest for $L^*a^*b^*$ space. Again, since the distortion measure is based on this space, this result should not be surprising. Examining the reconstructed images shows that subjectively this holds as well for AC_1C_2 space.

There are numerous color errors in the images coded in AC_1C_2 space. There is also visible ringing along diagonal edges such as the edge of the white animal in the GIRL image. The Case 1 image at a compression ratio of 12:1 has easily noticeable patches of color errors on the girl's face. These errors are even more annoying for the 16:1 compression ratio. For this color space, the Case 2 results have a lower average ΔE distortion than the Case 1 results. The Case 3 results are better yet, but they still contain more areas of visible color error than do the images coded in $L^*a^*b^*$ space.

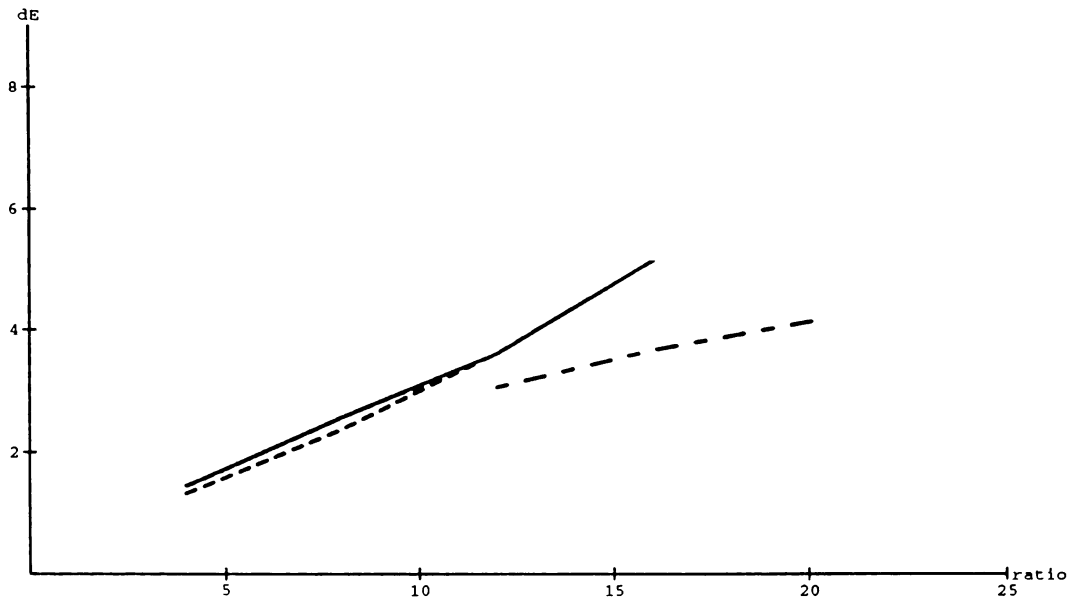


Figure 8.1: Average ΔE Distortion of the GIRL Image in $L^*a^*b^*$ Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.

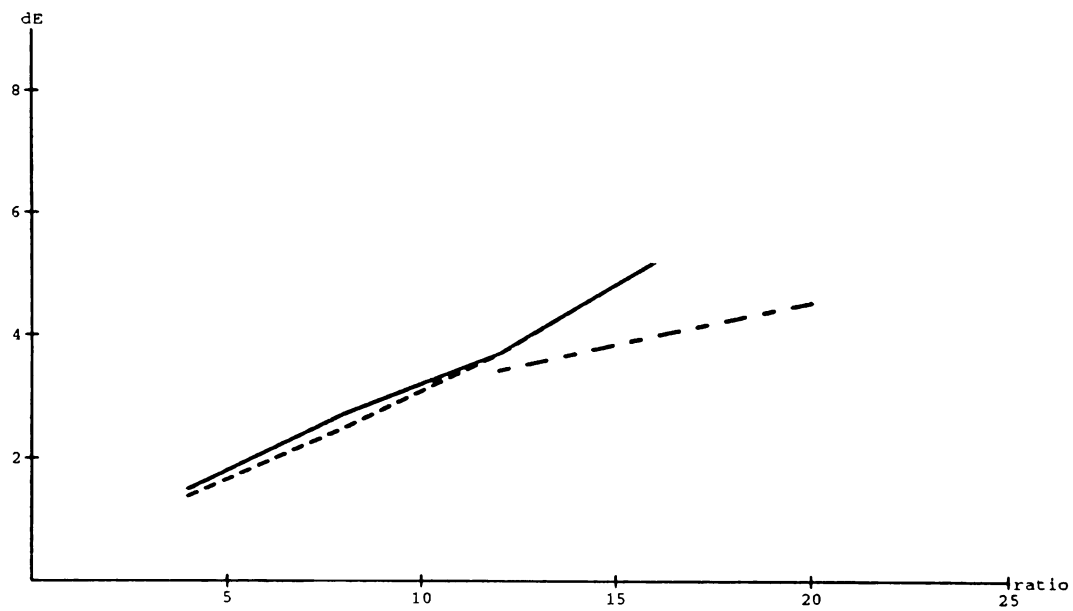


Figure 8.2: Average ΔE Distortion of the DOLL Image in $L^*a^*b^*$ Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.

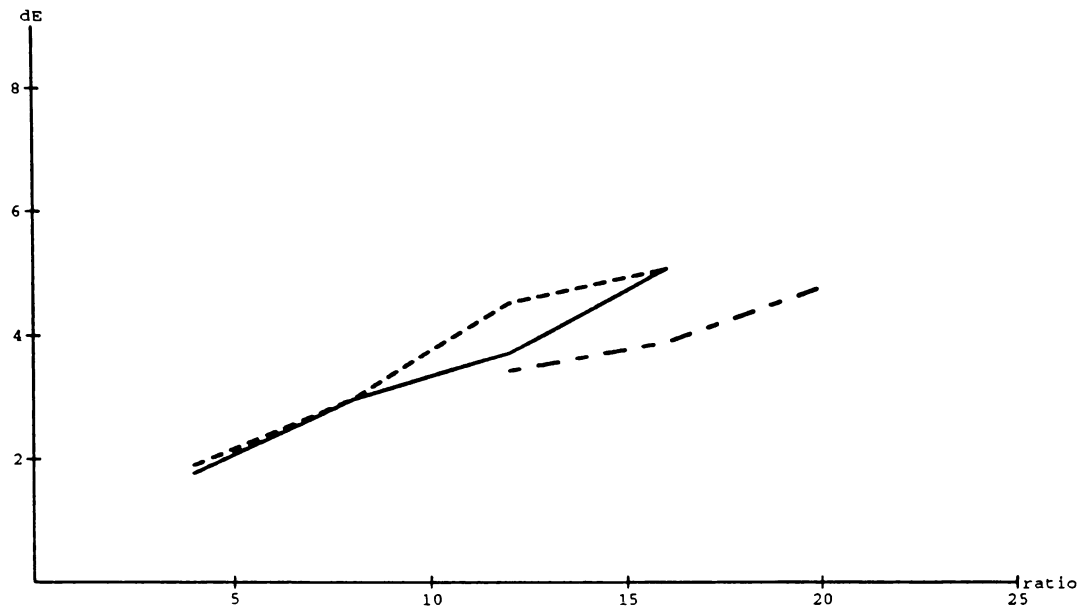


Figure 8.5: Average ΔE Distortion of the GIRL Image in AC_1C_2 Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.

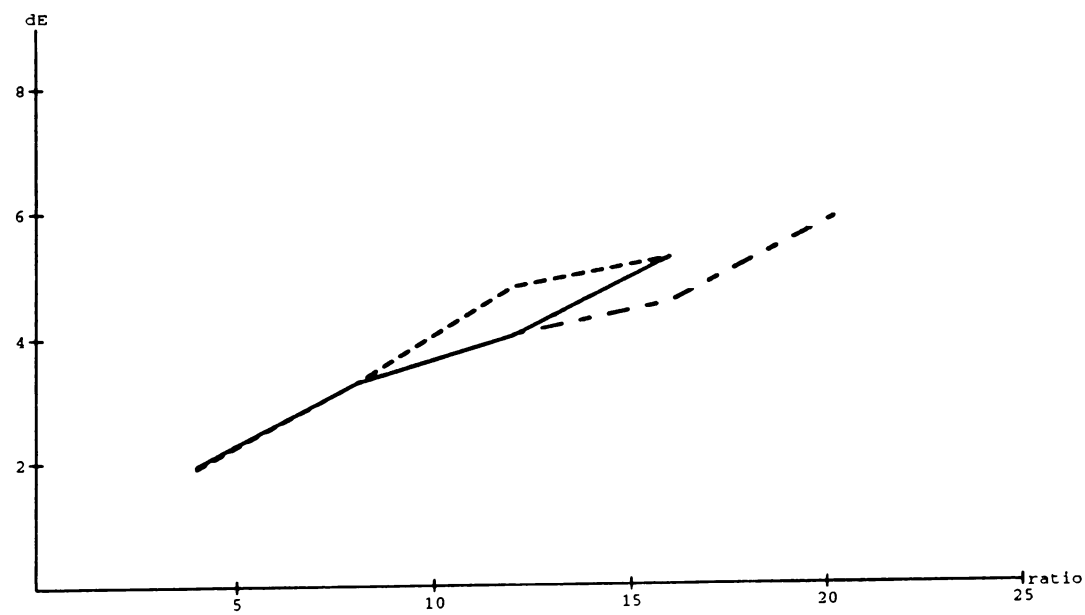


Figure 8.6: Average ΔE Distortion of the DOLL Image in AC_1C_2 Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.

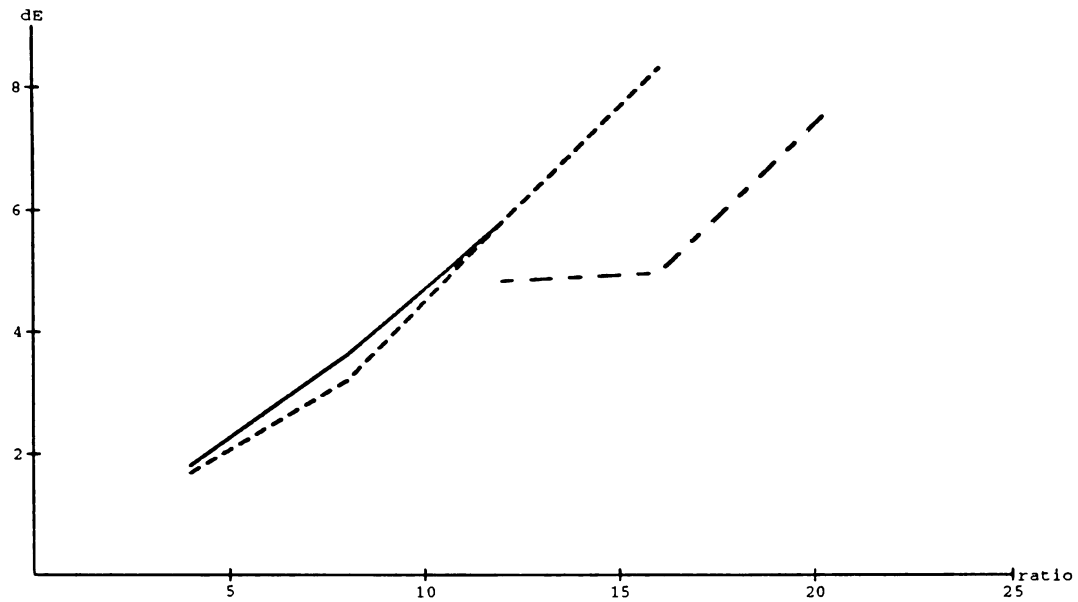


Figure 8.7: Average ΔE Distortion of the GIRL Image in YI^*Q^* Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.

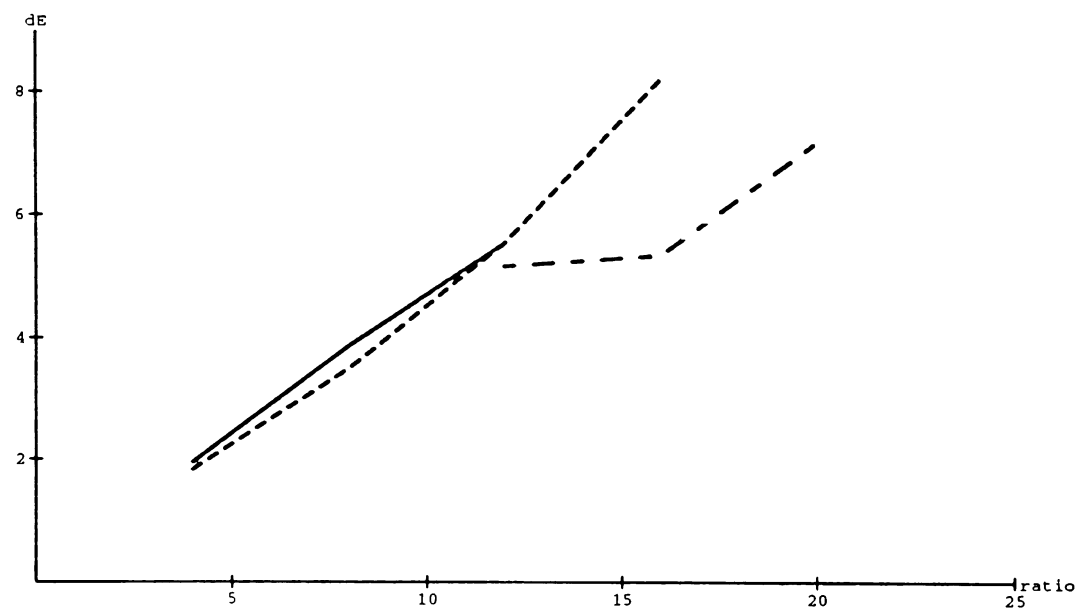


Figure 8.8: Average ΔE Distortion of the DOLL Image in YI^*Q^* Space versus Compression Ratio. The dotted line is Case 1, the solid line is Case 2, and the dot-dashed line is Case 3.

The images coded in YI^*Q^* space suffer from some desaturation, but are much more free from contouring and areas of visible color error than are the other two color spaces. Figure 8.9 shows the GIRL and DOLL images coded in this color space at a compression ratio of 20:1 for Case 3. There is some loss of sharpness and the yellow shirt of the clown is not as bright, but these images look far superior to those resulting from coding in the other two color spaces. The results at 16:1 suffer from even less of these problems.

For the rectangular configuration, it was recommended that this color space be used if some desaturation was allowable, but that $L^*a^*b^*$ space was a good compromise. For the diamond configuration, the recommendation is more strongly for YI^*Q^* space. Since this color space results in much smoother images and less ringing than does $L^*a^*b^*$ space, this space should be used whenever some desaturation is allowed. $L^*a^*b^*$ space may be better at compression ratios of 8:1 and even 12:1, but only if color fidelity is essential. Unfortunately, the color errors in AC_1C_2 space make this space the last one to use.

8.5 Rectangular Versus Diamond Configuration

Although the orientation of the lowest frequency subband in the diamond configuration more closely matches the orientation of the human visual system, its larger size contributes to the inferiority of this configuration when compared to the rectangular one. Even with vector quantization of this subband, the results are only good at a compression ratio of 12:1 and are only fair at 16:1. The rectangular configuration provides results at a compression ratio of 20:1 that are better than those of the diamond configuration at 16:1, and almost as good as those at 12:1.

In the practical implementation of a system, the rectangular configuration should be chosen. Not only are the results better, but the processing takes less time and uses less memory. A sixteen-band system would provide more degrees of freedom to the bit allocation algorithm than the seven-band system, and the added processing could be done in parallel. A seven-band system with diamond-shaped subbands obtained by using two-dimensional non-separable filters would yield better results than the five-band diamond system, and may provide results better than the seven-band rectangular system.

CHAPTER 9

SUMMARY AND CONCLUSIONS

9.1 Summary

The fundamentals of color science were given in Chapter 2. An understanding of color science and a knowledge of properties of the human visual system, allowed us to design better subband coding systems. After explaining the fundamentals of subband coding in Chapter 3, a new diamond subband coder was derived. The subbands in this coder more closely match the orientations of the HVS.

In Chapter 4, initial simulation results were presented for the coding of color images using fullband and subband coding with scalar quantization. These simulations showed that large color errors were possible at low compression ratios if a simple fullband system was used. The introduction of subband coding significantly improved the quality of the reconstructed images, but required the use of a bit allocation algorithm. A variance-based bit allocation was derived, and the resulting allocations were used to provide a preliminary assessment of the capabilities of subband coding of color images.

The theoretical formulation of vector quantization was given in Chapter 5, and the LBG algorithm was introduced. This algorithm allows one to design codebooks based on a training sequence. Various improvements to the basic Full Search Vector Quantizer were also given.

The problem of obtaining a perceptually optimal bit allocation was formulated and

solved in Chapter 6. The resulting Marginal Analysis algorithm uses the distortion-rate curves of the vector quantizers and a set of perceptual weights. These weights were derived using experimentally measured data of the mean detection threshold of the human visual system. A different set of weights was computed for the $L^*a^*b^*$, AC_1C_2, and YI^*Q^* color spaces.

Chapters 7 and 8 presented simulation results for the seven-band rectangular and five-band diamond subband configurations, respectively. The performance increase due to the use of the perceptual weights was demonstrated, and a comparison of the different vector quantizer cases was made. The types of artifacts resulting from coding in the different color spaces was also examined.

9.2 Contributions

Briefly, the major contributions of this dissertation are:

- The derivation of an iterative approximation to the optimal scalar quantizer.
- An investigation of image compression in various color spaces including perceptually uniform ones.
- The design of a general subband system for color image compression including the use of an internal color space, XYZ space.
- The design, analysis, and simulation of a new diamond subband configuration using separable quadrature mirror filters, and the comparison of this configuration with a standard rectangular one.
- The formulation of the bit allocation process for a color subband coding system as an optimization problem, and the solution of this problem yielding the perceptually optimal bit allocation.

• The derivation of perceptual weights need by the bit allocation algorithm from data that determined the mean detection threshold of the human visual system.

• The design of a practical subband/vector quantization coding system that achieves very high quality reconstructed images at low bit rates.

9.3 Future Extensions

9.3.1 Color Adaptive Systems

The perceptual weights derived in Chapter 6 were based on a background luminance of 5 cd/m^2 and a background color of white. Since the mean detection threshold experiments provide data for three different background luminances and four different background colors, a better bit allocation should be possible if the algorithm can be made color adaptive. In Chapter 5, Classified Vector Quantization was described. The encoder in this method attempts to classify the incoming vector into one of a number of classes and then chose a different codebook to encode each class.

This concept can be directly applied to color images. For a given color space and subband configuration, twelve different sets of perceptual weights are derived. Each set is based on one of the combinations of background color and background luminance. The bit allocation algorithm is then run for the desired bit rate using each set of perceptual weights. A classifier is incorporated into the encoder. This classifier determines to which of the twelve classes each input vector (or pixel) belongs. The chosen class determines the bit allocation to use, and this determines the set of codebooks used to encode the color components of the various subbands.

Since this is a subband - vector quantization system, the classifier and the bit allocation algorithm must be run on a block by block basis. These blocks correspond

to groups of pixels in the original image, and they must be large enough so that all of the vectors, created after the subband filtering, come from the same block. Specifically, for Cases 0 and 1, the block size must be at least 4×4 pixels. For Cases 2 and 3, the block size must be at least 8×8 pixels. To see why these sizes are necessary, consider Case 2. Each 8×8 block is filtered and decimated to create four subbands containing 4×4 blocks. The lowest frequency subband is then filtered and decimated to create four more subbands containing 2×2 blocks. These 2×2 blocks become the vectors sent to the vector quantizer. The use of smaller block sizes in the classifier would cause the vectors pertaining to a particular bit allocation to be created from overlapping blocks.

The advantage of this method is that the Marginal Analysis algorithm already requires that all of the codebooks for each subband color component be designed for bits rates from one bit/vector up to some maximum. The color information is then used to modify the selection of the codebooks to use. No additional codebooks have to be designed.

An initial investigation of this method was done. Sets of perceptual weights were derived for luminance values of 10 and 20 cd/m^2 for the white background. The bit allocation was run for Case 2 at compression ratios of 16:1, 20:1, and 24:1 in $L^*a^*b^*$ space. The bit allocations for $Y = 10$ cd/m^2 were the same as those for $Y = 5$ cd/m^2 for all three compression ratios. The bit allocation for $Y = 20$ cd/m^2 was the same at 16:1, and was very similar for compression ratios of 20:1 and 24:1. Additionally, sets of perceptual weights were derived for the background colors red, green, and blue for a luminance value of $Y = 5$ cd/m^2. The bit allocations for these three colors were exactly the same as those for the white background at the same luminance value for the 16:1 compression ratio. The bit allocations were also very similar for the other

two compression ratios.

It appears that the use of different sets of perceptual weights alone will not lead to significantly different bit allocations. This indicates that the gain in using this method will not be great. A further modification would be to also change the distortion-rate functions used in the bit allocation algorithm depending on where one is in the color space. This requires that different sets of codebooks be designed for the various combinations of background luminance and background color.

9.3.2 Incorporation in a Video Compression System

The major extension of the systems designed in this work is to video sequences such as high definition color television. By using quadrature mirror filters with a small number of taps and by doing the filtering in the time domain using a parallel architecture, it is possible to complete the processing of an image in one-thirtieth of a second. This straight-forward approach does not make use of the temporal correlation among the frames of a video sequence. Motion-compensated interframe prediction can be used to obtain an estimate of a particular frame. For color images, the motion estimate is usually determined from only the luminance component; the chrominance components use this estimate as well. The difference between the original image and the estimated image is then input to the subband system.

A three dimensional subband coding system can also be used to compress color video. Because of the high temporal correlation among frames, the video sequence is filtered into a number (often 2) of temporal subsequences, and these subsequences are then spatially filtered into a number of subimages. By using motion adaptive filtering, only two subbands are needed for stationary parts of a picture [79]. The bit rate depends upon how well the filters adapt to the velocity of moving objects in the

scene.

In addition to using the temporal correlations, entropy coding should also be added to a practical system. The quantized vectors created by the subband transmitter are not all equally probable. The use of entropy coding can reduce the bit rate still further, without any additional loss in quality. In one set of simulations, an arithmetic coder [67, 68, 59] was used to encode the four higher subbands after they were scalar quantized to eights bits/pixel per component in YIQ space. The bit rates of the color components of these subbands were less than one bit/pixel above the respective entropies. In this case, the arithmetic coding alone provided a 4:1 compression ratio for the four upper subbands. Since these subband components are allocated less than eight bits/pixel for useful compression ratios, the gain will not be as great. Yet, a ten to twenty percent gain can most likely be realized by entropy coding all of the subbands.

9.4 Conclusions

The combination of subband coding with vector quantization can yield images of very high quality while providing a compression ratio over 10:1. The use of a perceptually uniform color space for the quantization is integral to achieving high color fidelity at the higher compression ratios. Vector quantizing the color components separately adds extra degrees of freedom and leads to an increased performance. For compression ratios of 20:1 and greater, the chrominance components of the lowest frequency subband should also be vector quantized. Further improvements can be obtained by increasing the size of the vectors used in the higher frequency subbands.

The use of a perceptually uniform color space yields better color reproduction, but

can also lead to larger color artifacts in localized areas. For moderate compression ratios, the color artifacts are small enough that coding should be done in a perceptually uniform color space. Of the two examined, $L^*a^*b^*$ space has fewer areas of large color errors than AC_1C_2 space. The latter space has an exponential in the inverse transformation; color errors caused by the quantization are enhanced more in this space than in the former. If some error in the average colors of objects can be tolerated, YIQ space provides reconstructed images containing little contouring.

In this work, a rectangular configuration and a diamond configuration have been proposed. Both systems are capable of providing the compression ratio of roughly 9:1 that is needed to allow the transmission of distribution quality HDTV video over the 130 Mbits/sec H4 portion of the 155.52 Mbits/sec Sonet STS-3c channel. These systems are also capable of allowing high quality archiving of still images. Because of a larger number of subbands, the rectangular configuration provides more degrees of freedom to the bit allocation algorithm. This results in better quality images at the higher compression ratios. In particular, the rectangular configuration can achieve good quality images at compression ratios of 16:1 and higher.

CHAPTER 10

BIBLIOGRAPHY

[1] A.N. Akansu and M.S. Kadur, "Subband Coding of Video With Adaptive Vector Quantization," *Proc. ICASSP*, pp. 2109-2111, 1990.

[2] M. Antonini, M. Barlaud, P. Mathieu, and I. Daubechies, "Image Coding Using Vector Quantization in the Wavelet Transform Domain," *Proc. ICASSP*, pp. 2297-2300, 1990.

[3] R.H. Bamberger, "The Directional Filter Bank: A Multirate Filter Bank for the Directional Decomposition of Images," Doctoral Dissertation, School of Electrical Engineering, Georgia Institute of Technology, Nov. 1990.

[4] R.H. Bamberger, "New Subband Decompositions and Coders for Image and Video Compression," *Proc. ICASSP*, pp. 217-220. 1992.

[5] D. Barba and J. Hanen, "The Use of a Human Visual Model in Sub-band Coding of Color Video Signal with Adaptive Chrominance Signal Vector Quantization," *SPIE Conf. on Visual Comm. and Signal Proc.*, vol. 1605, pp. 408-419, 1991.

[6] C.D. Bei and R.M. Gray, "An Improvement of the Minimum Distortion Encoding Algorithm for Vector Quantization," *IEEE Trans. on Comm.*, vol. COM-33, pp. 1132-1133, Oct. 1985.

[7] F.J. Bingley, "Transfer Characteristics in NTSC Color Television," *Proceedings of the IRE*, pp. 71-78, Jan. 1954.

[8] S.E. Budge, T.G. Stockham Jr., D.M. Chabries, and R.W. Christiansen, "Vector Quantization of Color Digital Images within in Human Visual Model," *Proc. ICASSP*, pp. 816-819, 1988.

[9] Z.L. Budrikis, "Visual Fidelity Criterion and Modeling," *Proceedings of the IEEE*, vol 60, pp.771-779, July 1972.

[10] P.J. Burt and E.H. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Trans. Comm.*, vol. COM-31, pp. 532-540, Apr. 1983.

[11] C.K. Chan and J.C. Chow, "Optimal Display of True Color Images Using Vector Quantization Techniques," *Proc. IEEE VSPC*, pp. 5-8, 1991.

[12] P.A. Chou, T. Lookabaugh, and R.M. Gray, "Entropy-Constrained Vector Quantization," *IEEE Trans. on ASSP*, vol. ASSP-37, pp. 31-42, Jan. 1989

[13] R.E. Crochiere, S.A. Webber, and J.L. Flanagan, "Digital coding of speech in Subbands," *Bell System Technical Journal*, vol. 55, pp. 1069-1085., Oct. 1976.

[14] R.E. Crochiere and L.R. Rabiner, *Multirate Digital Signal Processing*, Prentice Hall, Englewood Cliffs, NJ, 1983.

[15] D.E. Dudgeon and R.M. Mersereau, *Multidimensional Digital Signal Processing*, Prentice Hall, Englewood Cliffs, NJ, 1984.

[16] M.O. Dunham and R.M. Gray, "An Algorithm for the Design of Labeled-Transition Finite-State Vector Quantizers,", *IEEE Trans. on Comm.*, vol. Com-33, pp. 83-89, Jan. 1985.

[17] D. Esteban and C. Galand, "Application of quadrature mirror filters to split band voice coding schemes," *Proc. ICASSP*, pp. 191-195, May 1977.

[18] H. Everett III, "Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources," *Operations Research*, vol. 11, pp. 399-417, 1963.

[19] M.H.A. Fadzil and T.J. Dennis, "Sample Selection in Subband Vector Quantization," *Proc. ICASSP*, pp. 2085-2088, April 1990.

[20] O.D. Faugeras, "Digital Color Image Processing Within the Framework of a Human Visual Model," *IEEE. Trans. on ASSP.*, vol. ASSP-27, pp. 380-393, Aug. 1979.

[21] O.D. Faugeras, "Digital Color Image Processing and Psychophysics Within the Framework of a Human Visual Model," Doctoral Dissertation, Dept. of Computer Science, University of Utah, Jun. 1976.

[22] J.C. Feauveau, P. Mathieu, M. Barlaud, and M. Antonini, "Recursive Biorthogonal Wavelet Transform for Image Coding," *Proc. ICASSP*, pp. 2649-2652, 1991.

[23] T.R. Fischer, "On the Rate-Distortion Efficiency of Subband Coding," *Proc. Globecom*, pp. 3.6.1-3.6.3, 1991.

[24] J. Foster, R.M. Gray, and M.O. Dunham, "Finite-State Vector Quantization for Waveform Coding," *IEEE Trans. on Info. Theory*, vol. IT-31, pp. 348-359, May 1985.

[25] B. Fox, "Discrete Optimization via Marginal Analysis," *Management Science*, vol. 13, pp. 210-216, Nov. 1966.

[26] G. Furlan, C. Galand, E. Lancon, and J. Menez, "Sub-Band Coding of Images Using Adaptive VQ and Entropy Coding," *Proc. ICASSP*, pp. 2665-2668, 1991.

[27] I. Furukawa, M. Nomura, and S. Ono, "Hierarchical Coding of Super High Definition Images with Subband + Multistage VQ," *Proc. ICASSP*, pp. 2637-2640, 1991.

[28] D.L. Gall, H. Gaggioni, and C. Chen, "Transmission of HDTV signals under 140 Mbits/s using a Sub-Band Decomposition and Discrete Cosine Transform Coding," *Signal Processing of HDTV*, North Holland, Amsterdam, pp. 287-293, 1988.

[29] R.S. Gentile, J.P. Allebach, and E. Walowit, "Quantization of Color Images Based on Uniform Color Spaces", *Journal of Imaging Technology*, vol. 16, pp. 11-21, Feb. 1990.

[30] A. Gersho and B. Ramamurthi, "Image Coding Using Vector Quantization," *Proc. ICASSP*, pp. 428-431, 1982.

[31] H. Gharavi and A. Tabatabai, "Sub-Band Coding of Monochrome and Color Images," *IEEE Trans. Circuits and Systems*, vol. 35, No. 2, pp. 207-214, Feb. 1988.

[32] H. Gharavi, "Differential Sub-Band Coding Of Video Signals," *Proc. ICASSP*, pp. 1819-1822, 1989.

[33] R.M. Gray, "Vector Quantization," *IEEE ASSP Magazine*, pp. 4-29, Apr. 1984.

[34] R.M. Gray and Y. Linde, "Vector Quantizers and Predictive Quantizers for Gauss-Markov Sources," *IEEE Trans. on Comm.*, vol. Com-30, pp. 381-389, Feb. 1982.

[35] R.M. Gray and H. Abut, "Full Search and Tree Searched Vector Quantization of Speech Waveforms," *Proc. ICASSP*, pp. 593-596, 1982.

[36] R.W. Hamming, *Digital Filters*, Prentice-Hall, Englewood Cliffs, NJ, 1983.

[37] J.J.Y. Huang and P.M. Schultheiss, "Block Quantization of Correlated Gaussian Random Variables," *IEEE Trans. on Comm. Systems*, vol. 11, pp. 289-296, Sept. 1963.

[38] R.W.G. Hunt, *The Reproduction of Colour*, John Wiley and Sons, New York, 1975.

[39] K. Irie and R. Kishimoto, "Adaptive Sub-Band Coder Design For HDTV Signal Transmission," *Proc. ICASSP*, pp. 2117-2120, 1990.

[40] O. Johnsen, O.V. Shentov, and S.K. Mitra, "A Technique for the Efficient Coding of the Upper Bands in Subband Coding of Images," *Proc. ICASSP*, pp. 2097-2100, 1990.

[41] J.D. Johnston, "A Filter Family Designed for use in Quadrature Mirror Filter Banks", *Proc. ICASSP*, pp. 291-294, Apr. 1980.

[42] G. Karlsson and M. Vetterli, "Theory of Two-Dimensional Multirate Filter Banks," *IEEE Trans. on ASSP*, vol. 38, pp. 925-937, Jun. 1990.

[43] C. Kim, J. Bruder, M. Smith, and R. Mersereau, "Subband Coding of Color Images Using Finite State Vector Quantization," *Proc. ICASSP*, pp. 753-756, 1988.

[44] C. Kim, M. Smith, and R. Mersereau, "An Improved SBC/VQ Scheme for Color Image Coding," *Proc. ICASSP*, pp. 1941-1944, 1989.

[45] Y.H. Kim and J.W. Modestino, "Adaptive Entropy Coded Subband Coding of Images," *IEEE Trans. Image Proc.*, vol. 1, pp. 31-48, Jan. 1992.

[46] B. Krishnakumar, "Visual Sensitivity to Color-Varying Stimuli," Master's thesis, Department of Electrical and Computer Engineering, North Carolina State University, Dec. 1991.

[47] H.C. Lee, "A Computational Model of Human Color Encoding," *Kodak Technical Report*, pp. 1-27, Apr. 1989.

[48] H. Li and Z. He, "Directional Subband Coding of Images," *Proc. ICASSP*, pp. 1823-1826, 1989.

[49] Y. Linde, A. Buzo, and R.M. Gray, "An Algorithm for Vector Quantizer Design," *IEEE Trans. on Comm.*, vol. Com-28, pp. 84-95, Jan. 1980.

[50] D.C. Livingston, "Colorimetric Analysis of the NTSC Color Television System," *Proc. IRE*, vol. 42, pp. 138-150, Jan. 1954.

[51] S.P. Lloyd, "Least Squares Quantization in PCM," *IEEE Trans. on Info. Theory*, vol. IT-28, pp. 129-137, Mar. 1982.

[52] K. Ma, "Subband Image Coding Using Absolute Moment Block Truncation with the Shannon-Bound Sequential Dynamic Bit Allocation," Doctoral Dissertation, Department of Electrical and Computer Engineering, North Carolina State University, Feb. 1992.

[53] D.L. MacAdam, "Uniform Color Scales," *Journal of the Optical Society of America*, vol. 64, pp. 1691-1702, Dec. 1974.

[54] B. Mahesh and W. Pearlman, "Hexagonal Sub-Band Coding For Images," *Proc. ICASSP*, pp. 1953-1956, 1989.

[55] S. Mallat, "A Theory for Multiresolution Signal Decomposition: The Wavelet Representation," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. II. No. 7, pp. 674-693, July 1989.

[56] V. Mani, "Calibration of Color Monitors," Master's thesis, Department of Electrical and Computer Engineering, North Carolina State University, 1991.

[57] J.L. Mannos and D.J. Sakrison, "The Effects of a Visual Fidelity Criterion on the Encoding of Images," *IEEE. Trans. on Information Theory*, vol. IT-20, pp. 525-536, July 1974.

[58] J. Max, "Quantizing for Minimum Distortion", *IRE Trans. Information Theory*, vol. IT-6, pp. 7-12, Mar. 1960.

[59] J.L. Mitchell and W.B. Pennebaker, "Software Implementations of the Q-Coder," *IBM J. Res. Develop.*, vol. 32, pp. 753-774, Nov. 1988.

[60] H. Mostafavi and D.J. Sakrison, "Structure and Properties of a Single Channel in the Human Visual System," *Vision Research*, vol. 16, pp. 957-968, 1976.

[61] N.M. Nasrabadi and R.A. King, "Image Coding Using Vector Quantization: A Review," *IEEE Trans. on Comm.*, vol. Com-36, pp. 957-971, Aug. 1988.

[62] M. Nomura, J. Suzuki, and N. Ohta, "Sub-band Coding for Super High Resolution Images," *Proc. Picture Coding Symposium*, pp. 10.3-1 10.3-2, 1990.

[63] J.B. O'Neal, Jr. "Predictive Quantizing Systems (Differential Pulse Code Modulation) for the Transmission of Television Signals," *Bell System Technical Journal*, vol. 45, pp. 689-721, May-Jun. 1966.

[64] J.B. O'Neal, Jr. "Entropy Coding in Speech and Television Differential PCM Systems," *IEEE Trans. on Info. Theory*, vol. IT-17, pp. 758-761, Nov. 1971.

[65] P.F. Panter and W. Dite, "Quantizing Distortion in Pulse-Count Modulation with Nonuniform Spacing of Levels," *Proc. IRE*, vol. 39, pp. 44-48, Jan. 1951.

[66] T.W. Parks and J.H. McClellan, "A Program for the Design of Linear Phase Finite Impulse Response Digital Filters," *IEEE Trans. on Audio and Electroacoustics*, vol. AU-20, pp. 195-199, Aug, 1972.

[67] W.B. Pennebaker, J.L. Mitchell, G.G. Langdon Jr., and R.B. Arps, "An Overview of the Basic Principles of the Q-Coder Adaptive Binary Arithmetic Coder," *IBM J. Res. Develop.*, vol. 32, pp. 717-726, Nov. 1988.

[68] W.B. Pennebaker and J.L. Mitchell, "Probability Estimation for the Q-Coder," *IBM J. Res. Develop.*, vol. 32, pp. 737-752, Nov. 1988.

[69] M.G. Perkins and T. Lookabaugh, "A Psychophysically Justified Bit Allocation Algorithm for Subband Image Coding Systems," *Proc. ICASSP*, pp. 1815-1818, 1989.

[70] C.I. Podilchuk, N.S. Jayant, and P. Noll, "Sparse Codebooks for the Quantization of the Non-Dominant Sub-Bands in Image Coding," *Proc. ICASSP*, pp. 2101-2104, April 1990.

[71] L.R. Rabiner, J.H. McClellan, and T.W. Parks, "FIR Digital Filter Design Techniques Using Weighted Chebyshev Approximation," *Proceedings of the IEEE*, vol 63. pp. 595-610, Apr. 1975.

[72] S.A. Rajala, H.J. Trussell, and B. Krishnakumar, "Visual Sensitivity to Color-Varying Stimuli," *SPIE/IS&T's Symposium on Electronic Imaging Science & Technology*, Feb. 1992.

[73] B. Ramamurthi and A. Gersho, "Image Vector Quantization with a Perceptually-Based Cell Classifier," *Proc. ICASSP*, pp. 32.10.1-32.10.4, 1984.

[74] B. Ramamurthi and A. Gersho, "Classified Vector Quantization of Images," *IEEE Trans. on Comm.*, vol. Com-34, pp. 1105-1115, Nov. 1986.

[75] T.A. Ramstad, "Considerations on Quantization and Dynamic Bit-Allocation in Subband Coders," *Proc. ICASSP*, pp. 841-844, 1986.

[76] E.A. Riskin and R.M. Gray, "A Greedy Tree Growing Algorithm for the Design of Variable Rate Vector Quantizers," *IEEE Trans. on Signal Proc.*, vol. 39, pp. 2500-2507, Nov. 1991.

[77] R.J. Safranek and J.D. Johnston, "A Perceptually Tuned Sub-band Image Coder with Image Dependent Quantization and Post-quantization Data Compression," *Proc. ICASSP*, pp. 1945-1948, 1989.

[78] L.E. Scales, *Introduction to Non-Linear Optimization*, Springer-Verlag, New York, 1985, pp. 142-144.

[79] G. Schamel, "Motion-Adaptive Subband Coding of HDTV Signals with 140 MBits/s," *Proc. Picture Coding Symposium*, pp. 1.7-1 1.7-3, 1990.

[80] A. Segall, "Bit Allocation and Encoding for Vector Sources," *IEEE Trans. on Info. Theory*, vol. IT-22, pp. 162-169, Mar. 1976.

[81] C.E. Shannon, "A Mathematical Theory of Communication," *Bell System Technical Journal*, vol. 27, pp. 379-423, 623-656, Jul. 1948.

[82] D.F. Shen, "Segmentation and HVS Based Image Coding with Motion Compensation for Packet Switched Network Environments," Doctoral Disseration, Dept. of Electrical and Computer Engineering, North Carolina State University, 1991.

[83] Y. Shoham and A. Gersho, "Efficient Bit Allocation for an Arbitrary Set of Quantizers," *IEEE Trans. on ASSP*, vol. ASSP-36, pp. 1445-1453, Sep. 1988.

[84] E. Simoncelli and E. Adelson, "Subband Image Coding with Hexagonal Quadrature Mirror Filters," *Proc. Picture Coding Symposium*, pp. 10.7-1 to 10.7-5, 1990.

[85] M.J.T. Smith and T.P. Barnwell III, "Exact Reconstruction Techniques for Tree-Structured Subband Coders," *IEEE Trans. on ASSP*, vol. ASSP-34, pp. 434-441, Jun. 1986.

[86] M.J.T. Smith and T.P. Barnwell III, "A New Filter Bank Theory for Time-Frequency Representation," *IEEE Trans. on ASSP*, vol. ASSP-35, pp. 314-327, Mar. 1987.

[87] W.N. Sproson, *Colour Science in Television and Display Systems*, Adam Hilger Ltd, Bristol, 1983.

[88] J.R. Sullivan, "Separable Subsampling of Digital Image Data with General Periodic Symmetry," Patent appl. no. 822768, Feb. 16, 1992.

[89] A.V. Trushkin, "Bit Number Distribution Upon Quantization of a Multivariate Random Variable," *Problems of Information Transmission*, vol. 16, pp. 76-79, 1980.

[90] A.V. Trushkin, "Optimal Bit Allocation Algorithm for Quantizing a Random Vector," *Problems of Information Transmission*, vol. 17, pp. 156-161, 1981.

[91] H.J. Trussell, "Applications of Set Theoretic Methods to Color Systems," *Color Research and Application*, vol. 16, pp. 31-41, Feb. 1991.

[92] P.P. Vaidyanathan, "Quadrature Mirror Filter Bands, M-Band Extensions and Perfect-Reconstruction Techniques," *IEEE ASSP Magazine*, pp. 4-20, Jul. 1987.

[93] P.P. Vaidyanathan, "Perfect Reconstruction QMF Bands for Two-Dimensional Applications," *IEEE Trans. on Circuits and Systems*, vol. CAS-34, pp. 976-978, Aug. 1987.

[94] R.E. Van Dyck and S.A. Rajala, "Sensitivity to Color Errors Introduced by Processing in Different Color Spaces", *Proc. IEEE VSPC*, pp. 192-195, 1991.

[95] R.E. Van Dyck and S.A. Rajala, "Design of Quantizers for Alternative Color Spaces," Presented at *IEEE MDSP Workshop*, p. 7.4, 1991.

[96] R.E. Van Dyck and S.A. Rajala, "Subband/VQ Coding in Perceptually Uniform Color Spaces," *Proc. ICASSP*, pp. 237-240, 1992.

[97] R.E. Van Dyck and S.A. Rajala, "Subband/VQ Coding of Color Images Using Separable Diamond-Shaped Subbands," *Proc. IEEE VSPC*, 1992.

[98] M. Vetterli, "A Theory of Multirate Filter Banks," *IEEE Trans. on ASSP*, vol. ASSP-35, pp. 356-372, Mar. 1987.

[99] M. Vetterli and D. Le Gall, "Perfect Reconstruction FIR Filter Banks: Some Properties and Factorizations," *IEEE Trans. on ASSP*, vol. 37, pp. 1057-1071, Jul. 1989.

[100] M.J. Vrhel and H.J. Trussell, "Filter Considerations in Color Correction," submitted to *IEEE Trans. on Image Proc.*

[101] P.H. Westerink, J. Biemond, and D.E. Boekee, "An Optimal Bit Allocation Algorithm for Sub-Band Coding," *Proc. ICASSP*, pp. 757-760, 1988.

[102] P.H. Westerink, D.E. Boekee, J. Biemond, and J.W. Woods, "Subband Coding of Images Using Vector Quantization," *IEEE Trans. on Comm.*, vol. Com-36, pp. 713-719, 1988.

[103] P.H. Westerink, J. Biemond, and D.E. Boekee, "Scalar Quantization Error Analysis for Subband Coding Using QMF's," *IEEE Trans. on Signal Processing*, vol. 40, pp. 421-428, Feb. 1992.

[104] J.W. Woods and S.D. O'Neil, "Subband Coding of Images," *IEEE Trans. on ASSP*, vol. ASSP-34, pp. 1278-1288, Oct. 1986.

[105] G. Wyszecki and W.S. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formulae*, John Wiley & Sons, New York, pp. 164-169. 1982.

[106] E. Yair, K. Zeger, and A. Gersho, "Competitive Learning and Soft Competition for Vector Quantizer Design," *IEEE Trans. on Signal Processing*, vol. 40, pp. 294-309, Feb. 1992.

[107] K. Zeger, J. Vaisey, and A. Gersho, "Globally Optimal Vector Quantizer Design by Stochastic Relaxation," *IEEE Trans. on Signal Processing*, vol. 40, pp. 310-322, Feb. 1992.

Appendix A

MEAN DETECTION THRESHOLD DATA

The mean detection threshold data used in Chapter 6 was measured by Krishnakumar, and results where published in [46, 72]. Tables of the mean detection threshold data were presented in [46] after a transformation to $L^*a^*b^*$ space. The transformation from XYZ to $L^*a^*b^*$ used the white point $(5\ 5\ 5)^T$. Since we desire to us a different white point, the original data was needed. Tables A.1, A.2, A.3, and A.4 contain the original mean detection threshold data in xyY space. These tables were obtained personally from Krishnakumar.

A few words must be said about the units of these measurements. In [72] it is stated that the variations in chromaticity use a contrast unit which corresponds to an absolute difference of $\Delta xy = 0.0006$. The contrast unit was given as $\Delta xy = 0.000745$ in Chapter 6. The difference is due to the fact that the data in the next four tables must be multiplied by 1.24 before conversion to XYZ space. This scaling was originally done by Krishnakumar so that the maximum values would be 100. Instead of multiplying all of the numbers in the tables by this factor, this factor was multiplied into the Δx and Δy values to give the values in Table 6.4.

LUM.	HUE	DIR.	FREQUENCY					
			0.5	1.0	2.0	5.0	10.0	20.0
Y = 5	Neutral	R-G	4.667	4.750	4.750	7.167	17.417	77.417
		B-Y	11.667	6.000	6.833	32.667	70.167	**
		Lum	8.167	6.750	6.333	7.250	13.500	65.083
	Green	R-G	3.083	4.000	4.667	14.167	29.250	*80.600
		B-Y	9.917	10.167	12.333	33.583	*106.500	**
		Lum	3.500	2.750	2.417	2.417	13.000	68.500
	Red	R-G	4.000	3.583	3.833	12.000	24.500	78.917
		B-Y	3.750	5.000	6.000	35.333	*81.100	**
		Lum	2.667	2.667	1.833	2.333	14.750	77.667
	Blue	R-G	2.750	1.917	2.417	6.917	15.167	75.583
		B-Y	6.917	13.917	20.250	31.917	*75.400	**
		Lum	2.417	1.583	1.417	1.750	21.583	81.250
Y = 10	Neutral	R-G	4.000	4.667	4.500	7.167	14.083	70.583
		B-Y	11.500	9.000	9.917	53.417	80.583	*113.750
		Lum	11.083	8.250	8.417	9.083	19.833	93.250
	Green	R-G	3.000	4.583	2.250	12.000	20.250	84.500
		B-Y	4.083	6.250	4.250	28.667	*100.333	**
		Lum	3.250	3.917	2.667	3.500	19.333	91.833
	Red	R-G	4.833	4.083	3.667	8.583	16.500	71.500
		B-Y	10.417	9.750	13.833	40.667	58.667	**
		Lum	11.667	8.833	8.667	9.417	22.000	98.667
	Blue	R-G	3.500	2.750	3.250	7.583	11.250	71.417
		B-Y	9.000	10.000	12.667	40.833	61.417	*102.250
		Lum	13.333	11.333	11.333	12.083	31.750	*99.600
Y = 20	Neutral	R-G	3.750	4.750	4.500	6.000	11.583	51.583
		B-Y	8.083	9.333	12.583	36.000	53.250	107.750
		Lum	10.917	7.917	6.333	7.250	31.500	108.667
	Green	R-G	3.333	4.417	3.833	10.500	17.083	65.750
		B-Y	11.667	10.417	12.333	45.500	*85.900	**
		Lum	11.417	6.500	7.000	9.500	26.667	102.500
	Red	R-G	4.417	4.500	3.500	7.917	11.833	69.333
		B-Y	8.667	8.333	14.083	38.500	53.417	98.667
		Lum	16.833	12.750	13.083	14.500	32.583	*109.100
	Blue	R-G	2.917	2.250	2.417	6.333	11.167	*83.400
		B-Y	6.750	6.083	13.833	23.500	43.000	*84.125
		Lum	20.917	19.750	18.750	18.750	36.833	*107.167

Table A.1: Mean Detection Threshold Contrasts in xyY Space. Horizontal Orientation

LUM.	HUE	DIR.	FREQUENCY					
			0.5	1.0	2.0	5.0	10.0	20.0
Y = 5	Neutral	R-G	5.083	5.583	7.083	9.250	23.000	-90.375
		B-Y	11.000	6.667	9.417	31.833	*65.700	**
		Lum	7.000	6.833	6.250	6.833	22.500	*77.800
	Green	R-G	5.167	4.167	4.500	14.750	31.083	**
		B-Y	9.833	11.833	15.333	37.750	*102.750	**
		Lum	5.917	4.500	3.250	4.917	16.083	*78.125
	Red	R-G	5.167	3.583	4.000	13.250	26.583	*89.750
		B-Y	4.250	6.167	9.833	37.833	83.583	**
		Lum	4.167	3.000	3.667	5.250	20.417	*83.000
	Blue	R-G	3.500	4.000	4.833	9.917	18.833	*87.167
		B-Y	9.000	18.167	21.333	31.750	*72.400	**
		Lum	4.583	4.000	4.917	6.167	26.750	*91.333
Y = 10	Neutral	R-G	4.500	5.417	5.667	9.583	18.167	80.917
		B-Y	11.417	7.583	11.583	52.833	78.750	*117.000
		Lum	11.750	10.417	9.833	11.000	23.083	*104.100
	Green	R-G	3.583	5.000	3.333	12.667	25.083	*86.833
		B-Y	6.083	8.500	10.500	38.417	*98.400	**
		Lum	4.917	4.250	4.333	4.167	21.917	*92.833
	Red	R-G	6.250	4.583	5.250	8.917	20.500	*86.250
		B-Y	10.417	9.333	18.417	37.333	60.417	**
		Lum	13.167	10.917	10.917	10.333	29.667	*103.667
	Blue	R-G	4.000	4.250	4.917	9.917	16.500	*81.200
		B-Y	9.583	11.417	16.667	39.250	*49.600	*119.000
		Lum	12.417	11.917	11.917	14.333	42.167	*91.000
Y = 20	Neutral	R-G	4.333	4.167	5.333	7.833	15.333	63.417
		B-Y	9.500	11.083	13.083	39.750	56.083	*111.750
		Lum	12.167	9.917	6.833	11.917	45.250	*110.750
	Green	R-G	4.417	5.083	4.833	12.667	22.500	*80.000
		B-Y	11.333	13.083	12.250	*46.400	*83.200	**
		Lum	13.500	11.500	9.167	12.750	33.417	*103.333
	Red	R-G	5.083	5.917	5.667	8.917	16.000	*60.167
		B-Y	9.333	10.833	20.750	39.750	51.083	*108.500
		Lum	17.083	14.833	15.000	16.917	40.250	*109.500
	Blue	R-G	4.333	4.250	4.583	8.833	13.917	71.250
		B-Y	8.917	8.750	15.417	27.250	*38.700	**
		Lum	22.083	20.917	20.333	23.833	50.750	*101.000

Table A.2: Mean Detection Threshold Contrasts in xyY Space. Vertical Orientation.

LUM.	HUE	DIR.	FREQUENCY					
			0.5	1.0	2.0	5.0	10.0	20.0
Y = 5	Neutral	R-G	7.167	7.917	7.167	16.083	37.500	*100.750
		B-Y	16.000	12.417	18.500	*45.500	*86.500	**
		Lum	8.250	7.667	6.917	11.167	37.083	*49.000
	Green	R-G	7.667	5.917	6.250	21.500	57.000	*98.500
		B-Y	18.750	17.583	25.833	*58.250	*108.000	**
		Lum	6.833	5.417	4.917	10.583	38.917	*61.000
	Red	R-G	7.833	7.917	8.333	19.000	44.167	*62.000
		B-Y	13.583	13.583	17.333	*49.300	*93.000	**
		Lum	8.750	8.000	7.417	13.833	33.833	*54.000
	Blue	R-G	5.167	5.417	6.500	15.583	34.500	*79.750
		B-Y	15.667	13.500	31.833	69.000	*82.000	**
		Lum	8.750	7.833	7.250	16.917	*44.400	*65.500
Y = 10	Neutral	R-G	5.333	5.333	4.833	13.000	33.333	*75.000
		B-Y	13.833	13.667	24.250	58.167	*83.100	**
		Lum	11.750	10.167	9.000	19.417	47.917	*99.500
	Green	R-G	5.833	4.167	7.250	16.250	50.500	*70.000
		B-Y	11.500	12.750	25.833	*67.100	*110.000	**
		Lum	6.083	5.583	4.750	11.417	47.667	*78.000
	Red	R-G	4.750	4.667	5.250	14.750	35.083	*89.000
		B-Y	9.000	12.333	23.417	52.000	*75.750	**
		Lum	9.083	6.917	6.417	14.583	*46.800	*101.500
	Blue	R-G	5.083	4.667	6.083	14.417	37.333	*73.667
		B-Y	14.167	14.000	29.583	60.417	*54.333	**
		Lum	15.833	14.083	13.167	21.917	*54.750	*110.500
Y = 20	Neutral	R-G	4.083	4.250	5.000	9.167	27.417	*84.625
		B-Y	11.333	12.083	25.583	53.833	72.417	**
		Lum	16.833	11.000	9.333	15.333	58.417	**
	Green	R-G	7.417	6.750	7.333	16.083	42.083	*92.667
		B-Y	15.500	15.333	26.500	*66.100	*89.500	**
		Lum	18.083	15.667	13.167	23.000	*61.400	*86.500
	Red	R-G	7.500	5.917	6.333	14.000	31.333	*72.800
		B-Y	11.083	11.750	22.500	52.250	*75.000	*111.750
		Lum	19.833	15.083	14.417	26.167	*62.900	*83.000
	Blue	R-G	6.333	5.417	6.083	12.750	30.417	*88.500
		B-Y	12.250	12.333	24.750	52.750	*71.375	**
		Lum	30.583	24.333	22.500	38.667	*77.375	*61.000

Table A.3: Mean Detection Threshold Contrasts in xyY Space. Left Diagonal Orientation.

LUM.	HUE	DIR.	FREQUENCY					
			0.5	1.0	2.0	5.0	10.0	20.0
Y = 5	Neutral	R-G	7.250	7.750	6.333	13.833	35.417	*103.500
		B-Y	16.417	13.750	19.750	*47.750	*83.000	*114.000
		Lum	8.583	8.083	7.583	9.167	42.583	*85.750
	Green	R-G	7.417	6.333	5.750	21.583	55.583	*110.000
		B-Y	19.250	17.917	27.083	*58.750	*106.250	**
		Lum	7.083	5.000	5.083	9.000	41.667	*82.500
	Red	R-G	8.333	8.333	7.417	19.500	47.917	*109.000
		B-Y	13.917	13.583	16.750	*50.900	*95.375	**
		Lum	9.500	7.667	6.667	15.917	33.167	*78.000
	Blue	R-G	5.250	6.583	6.417	13.083	35.917	*91.000
		B-Y	16.083	13.167	34.250	71.917	*84.667	**
		Lum	8.917	8.250	7.833	17.417	*38.500	*97.250
Y = 10	Neutral	R-G	5.750	5.500	4.500	14.583	34.667	*96.250
		B-Y	14.417	15.417	24.917	57.750	*88.300	*117.000
		Lum	13.417	10.083	8.750	18.750	36.750	**
	Green	R-G	6.750	4.333	7.000	14.917	45.000	*69.500
		B-Y	12.083	13.667	25.583	*66.200	*108.500	**
		Lum	6.333	6.167	3.667	12.167	53.250	**
	Red	R-G	5.250	4.500	5.167	16.917	32.917	*53.000
		B-Y	10.583	12.667	24.083	52.000	*58.000	**
		Lum	9.417	5.917	6.250	14.000	*50.800	*100.500
	Blue	R-G	5.583	4.917	4.583	14.667	33.833	*66.250
		B-Y	15.083	14.917	29.750	63.667	*81.333	**
		Lum	16.750	14.500	12.583	22.833	*42.125	**
Y = 20	Neutral	R-G	3.833	4.333	4.583	9.167	27.667	*73.375
		B-Y	11.583	11.667	26.417	57.000	*51.375	**
		Lum	16.083	12.750	9.083	16.583	60.167	**
	Green	R-G	8.083	6.667	7.417	17.167	42.083	*72.500
		B-Y	16.250	14.583	26.750	*63.400	*65.667	**
		Lum	19.083	16.250	13.417	25.833	67.000*	**
	Red	R-G	7.750	6.167	6.417	14.917	27.833	*77.700
		B-Y	11.083	11.000	23.250	52.333	*75.900	*114.000
		Lum	20.917	15.500	13.917	26.167	*45.250	**
	Blue	R-G	6.750	5.833	7.667	11.583	28.667	*99.750
		B-Y	12.417	13.083	25.250	53.667	*61.500	*111.000
		Lum	31.583	24.417	23.583	42.417	*74.500	*97.500

Table A.4: Mean Detection Threshold Contrasts in xyY Space. Right Diagonal Orientation.

Appendix B

PHOTOGRAPHIC PROCESSING

All of the color photographs that appear in Chapters 2, 4, 7, and 8 were taken with an Olympus OM-1 35 mm single lens reflex camera with focal plane shutter. The lens used was an F Zuiko Auto-S with a 50 mm focal length and a minimum f-stop of 1.8. This lens was used in combination with a Hoya close-up lens having +2 magnification.

Except for the toy store image, all of the images were of size 256×256 pixels. These images were zoomed to a size of 512×512 using pixel replication. The zoomed images were the ones photographed. This was done for two reasons. The first one is that the larger size made the photography easier. The second reason is that the point spread function of the phosphors on the picture tube extends over at least two pixels, so the replication reduces aliasing. The toy store image had to be subsampled to fit on the display. The subsampled image was the one photographed.

The film used was Kodak Gold Plus with an ASA speed of 200. The shutter speed was either one-half of a second with an f-stop of 5.6, or one second with an f-stop of 8; the former combination was used in all but a few photographs. The negatives were developed with a "normal" density, and all of the prints of the DOLL images were made using a +4 density. Most of the prints of the GIRL image also were made using a +4 density, but a few were made using a +3 density.

Since the development process was beyond the control of the researcher, the photographs can not be considered calibrated. That is, the colors in the photographs are not guaranteed to match those seen on the calibrated color monitor. Instead, these photographs are used to illustrate the types and locations of the coding artifacts, and to give some indication as to their severities.

to the four smaller subbands, and let the subscripts m and l refer to the three larger subbands.

Through algebraic manipulation of Eq. (4.11), one can show that the bit rates for the components of the smaller subbands are related by

$$b_{k,c} = b_{j,s} + \frac{1}{2}\log(\frac{\sigma_{k,c}^2}{\sigma_{j,s}^2}) \quad k,j = 1\ldots4, \quad c,s = 1\ldots3. \tag{4.12}$$

Similarly, the bit rates for the components of the larger subbands are related to the bit rates for components of the smaller subbands according to

$$b_{m,c} = b_{j,s} + \frac{1}{2}\log(\frac{\sigma_{m,c}^2}{4\sigma_{j,s}^2}) \quad m = 5\ldots7, \quad j = 1\ldots4, \quad c,s = 1\ldots3. \tag{4.13}$$

Finally, the bit rates for the components of the larger subbands are related by

$$b_{m,c} = b_{l,s} + \frac{1}{2}\log(\frac{\sigma_{m,c}^2}{\sigma_{l,s}^2}) \quad m,l = 5\ldots7, \quad c,s = 1\ldots3. \tag{4.14}$$

Substitution of Eqs. (4.12), (4.13), and (4.14) into Eq. (4.7) yields the following solution

$$\begin{aligned} b_{j,s} &= \frac{1}{3}B + \frac{1}{2}\log\sigma_{j,s}^2 + \frac{3}{4} - \frac{1}{24}\log\sigma_{gm}^2 \\ b_{l,s} &= \frac{1}{3}B + \frac{1}{2}\log\sigma_{l,s}^2 - \frac{1}{4} - \frac{1}{24}\log\sigma_{gm}^2 \end{aligned} \tag{4.15}$$

where $j = 1\ldots4$, $l = 5\ldots7$, $s = 1\ldots3$, σ_{gm}^2 is the geometric mean given by

$$\sigma_{gm}^2 = (\prod_{k=1}^{4}\prod_{c=1}^{3}\sigma_{k,c}^2)^{1/4}(\prod_{m=5}^{7}\prod_{c=1}^{3}\sigma_{m,c}^2), \tag{4.16}$$

and the logarithms are base 2.

Given the variances for each subband color component, one uses Eqs. (4.15) and (4.16) to compute the bit allocations for the three color components for each of the four smaller and three larger subbands. If a calculated allocation is negative, it is set to zero. Positive, non-integer bit allocations are then rounded to the nearest integer. If the total bit rate does not match the desired rate, then some of the allocations must be adjusted. Usually this requires decreasing some of the allocations to compensate for the increase caused by making the negative bit allocations zero. These decrements are best made in the higher frequency chrominance components since the human visual system is less sensitive to high frequency chrominance transitions.

4.5 Subband - Optimal Scalar Quantizer Results

In this section, we present results of using the variance-based bit allocation for the subband coder with scalar quantization. Since the distortion will be less than that obtained with scalar quantization alone, compression ratios of 4:1 and 8:1 are used. These results were first shown in [95]. The simulations were run in $L^*a^*b^*$, YIQ, and RGB color space.

The variances of the subband color components of the GIRL image were used to determine the bit allocation using Eqs. (4.15) and (4.16). The results were rounded to the nearest non-negative integer and adjusted so that the total bit rate was satisfied. These bit allocations are given in Tables 4.2 and 4.3 for the 4:1 and 8:1 compression ratios, respectively. The maximum number of bits allowed per component was limited to seven.

Once the number of bits to be allocated to each subband color component was known, the actual quantizers had to be designed. Since the derivation of the variance-