1-1-1990

Image Segmentation using Human Visual System Properties with Applications in Image Compression

Heidi A. Peterson
Purdue University

Sarah A. Rajala
North Carolina State University

Edward J. Delp
Purdue University

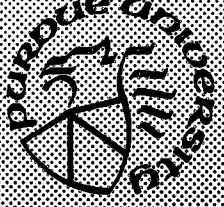


Image Segmentation using Human Visual System Properties with Applications in Image Compression

Heidi A. Peterson
Sarah A. Rajala
Edward J. Delp

TR-EE 90-4
January, 1990

School of Electrical Engineering
Purdue University
West Lafayette, Indiana 47907

IMAGE SEGMENTATION USING HUMAN VISUAL SYSTEM PROPERTIES WITH APPLICATIONS IN IMAGE COMPRESSION

Heidi A. Peterson

Sarah A. Rajala

Edward J. Delp

Purdue University

School of Electrical Engineering

West Lafayette, Indiana 47907

TR-EE-90-4

January 1990

S. A Rajala is with the Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, North Carolina.

Address all correspondence to E. J. Delp at Purdue University.

TABLE OF CONTENTS

Page

Page

LIST OF TABLES

Page

LIST OF FIGURES

Page

Page

Page

Page

Page

Page

Page

Page

Page

Page

NOTATION

HVS = human visual system

bpp = bits per pixel

$\mathbf{R}$ = the set of real numbers

$\mathbf{Z}$ = the set of integers

$\mathbf{E}$ = Euclidean space: $\mathbf{Z}^2$, $\mathbf{R}^2$, $\mathbf{Z}^3$, $\mathbf{R}^3$,$\cdots$

A, B, F, G = subsets of $\mathbf{E}$

a, b, c, s, t, x, y, z = elements or vectors in $\mathbf{E}$

e, f, g = functions in $\mathbf{E}$

d, h, m, r, u, v, w, C, E = real numbers

i, j, l, n, L, M, N, Q = integers

A^c = the complement of A with respect to $\mathbf{E}$

$\{x\colon \mathrm{P}\}$ = the set of points x which satisfy property P

$A \subseteq B$ = A is a subset of B

$\cup$ = set union

$\cap$ = set intersection

$\varnothing$ = the the empty set

$A - B$ = the set difference between A and B

$A_b = \{a + b\colon a \in A\}$ = the translate of A by b

$B^s = \{-b\colon b \in B\}$ = the symmetric set of B with respect to the origin

$A \oplus B$ = Minkowski set addition of A and B

$A \ominus B$ = Minkowski set subtraction of A and B

$A \, d \, B$ = dilation of A by B

$A \, e \, B$ = erosion of A by B

$\Psi(\cdot)$ = a morphological transformation

$nB = B \oplus B \oplus \cdots \oplus B$ (n times)

$S_n(A)$ = the n'th skeleton subset of A with respect to B

$SK(A)$ = the binary morphological skeleton of A

$$[skf(A)](i,j) = \text{the gray level skeleton function of } A = \begin{cases} n+1, & (i,j) \in S_n(A) \\ 0, & (i,j) \notin SK(A) \end{cases}$$

$$k_n(i,\ j) = \text{the binary characteristic function of the set } nB = \begin{cases} 1, & (i,\ j) \in nB \\ 0, & (i,\ j) \notin nB \end{cases}$$

$$[pgf(A)](i,j) = \text{the pseudo-graytone function of } A = \sum_{n=0}^{N} \sum_{(r,t) \in S_n(A)} k_n(i-r,\ j-t)$$

p, p_i = gray level of an image pixel

$\bar{p}$ = average gray level of a group of image pixels

P_i = an associated gray level

BDI = a binary decomposition image

a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q = image segments

$\mathcal{A}$, $\mathcal{B}$ = sets of image segments

ABSTRACT

In order to represent a digital image, a very large number of bits is required. For example, a 512×512 pixel, 256 gray level image requires over two million bits. This large number of bits is a substantial drawback when it is necessary to store or transmit a digital image. Image compression, often referred to as image coding, attempts to reduce the number of bits used to represent an image, while keeping the degradation in the decoded image to a minimum.

One approach to image compression is segmentation-based image compression. The image to be compressed is segmented, i.e. the pixels in the image are divided into mutually exclusive spatial regions based on some criteria. Once the image has been segmented, information is extracted describing the shapes and interiors of the image segments. Compression is achieved by efficiently representing the image segments.

In this thesis we propose an image segmentation technique which is based on centroid-linkage region growing, and takes advantage of human visual system (HVS) properties. We systematically determine through subjective experiments the parameters for our segmentation algorithm which produce the most visually pleasing segmented images, and demonstrate the effectiveness of our method. We also propose a method for the quantization of segmented images based on HVS contrast sensitivity, and investigate the effect of quantization on segmented images.

We apply these segmentation and quantization methods in a new compression technique which fits into the category commonly known as "second generation" image compression methods. Our compression method is designed for application single-frame images (i.e. not time-varying imagery). Other segmentation-based image compression techniques have typically represented the image segments by encoding the boundaries of the segments. We propose the use of morphological skeletons to represent the segments. The morphological skeleton of an image is similar to the medial axis. We describe the application of mathematical morphology to generate skeletons for the image segments, and discuss the advantages and disadvantages of using morphological skeletons in segmentation-based image compression.

INTRODUCTION

Image compression, often referred to as image coding, attempts to reduce the number of bits used to represent an image, while keeping the degradation in the decoded image to a minimum. Image compression is important in applications that require efficient storage or transmission of images or sequences of images.

Many different approaches to image compression have been investigated [1]. In Chapter 1 of this thesis an extensive overview of image compression is given. One approach to image compression discussed in Chapter 1 is segmentation-based image compression [2-4]. With this technique, the image to be compressed is segmented, i.e. the pixels in the image are divided into mutually exclusive spatial regions based on some criteria. Once the image has been segmented, information is extracted describing the shapes and interiors of the image segments, and compression is achieved by efficiently representing the image segments. In this thesis we present a new segmentation-based image compression technique.

Our compression technique is different from other segmentation-based image compression schemes in several ways. First, we employ an improved version of a previously proposed image segmentation technique, centroid-linkage region growing [5]. Since the decoded images will be viewed by humans, the motivation behind our adaptation of this algorithm is the production of visually pleasing segmented images. Our segmentation method takes advantage of human visual system (HVS) properties to achieve visually pleasing image segmentation. We present the results of systematic subjective experiments performed to determine the parameters of the segmentation algorithm which result in the most visually pleasing segmented images, and we demonstrate the effectiveness of our method. The segmentation algorithm is discussed in Chapter 2.

A second difference in our compression technique is the quantization of the segmented images. The segmented image is quantized to reduce the number of gray levels in the segmented image, which results in a reduction in the bit rate. We have investigated the effect of quantization on segmented images, and we show that a segmented image can be quantized from approximately 200 gray levels to approximately 25 gray levels, with virtually no visible degradation in the segmented image. We also propose a method for the quantization of segmented images based on

HVS contrast sensitivity, and compare this quantizer to both uniform and histogram-based quantizers. Quantization is discussed in Chapter 3.

A third difference in our compression technique involves the representation we use for the image segment shapes. Other segmentation-based gray level image compression techniques have typically represented the image segments by encoding the segment boundaries [2, 6, 7]. We propose the use of skeletons generated using mathematical morphology to represent the segment shapes. The basic operations in mathematical morphology [8] are reviewed in Chapter 4, and the process for morphological skeletonization of a binary image [9] is described in Chapter 5. Binary morphological skeletons have previously been used for compression of *binary* images[10]. We describe the application of binary mathematical morphology in a segmentation-based image compression scheme to compress *gray level* images.

The techniques described above have been combined and applied in a new segmentation-based image compression scheme. A block diagram of this method is shown in Figure 1. The complete image compression algorithm is described in Chapter 5. In the first steps of this algorithm, the segmentation and quantization techniques from Chapters 1 and 2 are applied to generate a segmented quantized image. The image resulting after segmentation and quantization is the image that will be decoded. Next the morphological operations described in Chapters 4 and 5 are used to generate gray level skeletons to represent the image segments for compression. Finally, these skeletons are coded. We have explored several different options for coding the segment skeletons and segment gray levels. These options are described in detail in Chapter 5, along with the decoding process for each option. Several test images have been coded and decoded to demonstrate our compression algorithm, and bit rates in the neighborhood of 0.5 to 2 bits per pixel (bpp) have been attained. Finally, we compare our skeleton-based method for coding the segment shapes to coding segment boundaries to represent the shapes, and discuss the advantages and disadvantages of using morphological skeletons in segmentation-based image compression.

Our research has resulted in contributions in the areas of image segmentation, quantization, and compression. We have systematically designed a centroid-linkage region growing algorithm which incorporates HVS properties to produce visually pleasing segmented images. We have also designed a method for filtering segmented images to remove visually insignificant segments. We then evaluated the effectiveness of our methods through subjective tests.

We have proposed quantization of segmented images and designed a HVS-based quantizer. This quantizer was then compared through subjective tests to several other quantizers. We also have investigated the interactions between various steps in the segmentation and quantization algorithms.

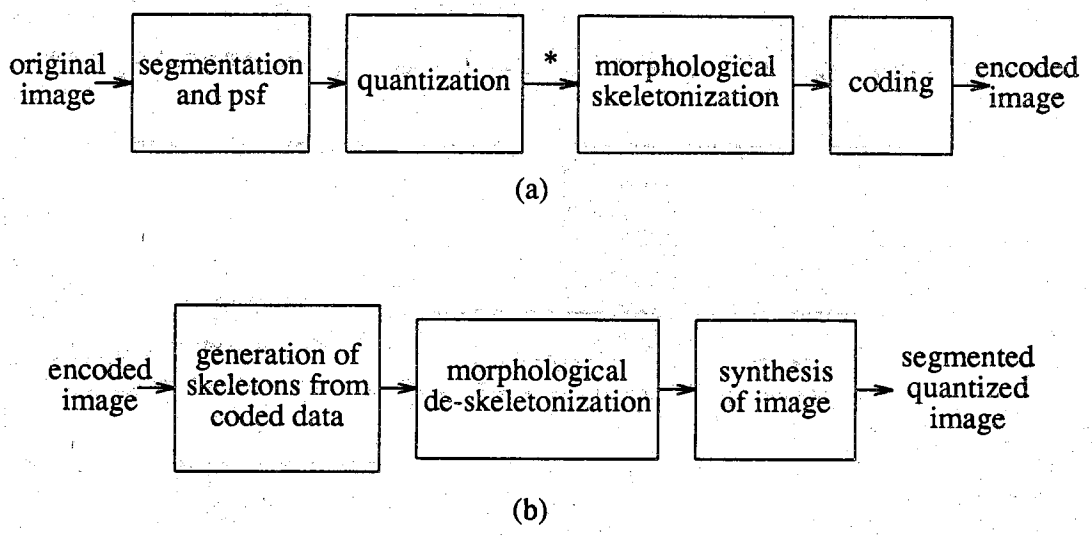


Figure 1. A new segmentation-based image compression technique. (a) Encoder. (b) Decoder. The image produced at "*" is the image that will be decoded ("psf" refers to post-segmentation filtering).

We have applied our results in segmentation and quantization to a new image compression technique. This technique uses morphological skeletons in a new way for image compression. We have also proposed the concept of the "minimal set of segments," which is useful in our compression technique. Finally, we have compared our compression technique to other segmentation-based image compression methods.

CHAPTER 1
AN OVERVIEW OF IMAGE COMPRESSION

In order to represent an image in a digitized format, a very large number of bits is required. This large number of bits is a substantial drawback when it is necessary to store or transmit an image or sequence of images. Image bandwidth compression techniques, often referred to as image coding, attempt to reduce the number of bits used to represent an image, while keeping the degradation in the quality of the decoded image to a minimum. In this chapter we review a wide variety of image coding methods. We divide image coding techniques into two general classes, and we describe coding methods which fit each of these classes. In addition, some of the important issues in image coding are discussed. We discuss the image model, the image quality measure, and the coding application. We also discuss the impact of broadband communication technology on the image coding problem.

1.1 Introduction

In society today there are a multitude of applications where the transmission or storage of images is required. Satellites transmit images to earth for use in areas such as remote sensing, the study of weather patterns, and military reconnaissance. Satellite links are used to transmit television programs around the world. Images must be transmitted for video-teleconferencing, for facsimile transmission of printed matter, and for deaf communication [11].

The transmission of images is either very time consuming or very expensive in bandwidth. To represent an uncompressed 512×512 pixel, 256 gray level image requires over two million bits. Transmission of this image over a 64Kbit/s channel requires more than thirty seconds. The requirements are even higher for a color image of the same size. This virtually precludes the transmission of real time digital video (time-varying imagery), or the transmission of large volumes of high quality still images in a reasonable amount of time. In order to accomplish these tasks, some scheme for image data compression is necessary, and/or the data rate of the channel must be dramatically increased beyond the 64Kb/s which is generally available today.

Currently a new generation of high speed communication channels, such as the Integrated Services Digital Network (ISDN) is being proposed. These channels may have data rates as high as 135Mb/s [12], which is fast enough to allow the transmission of most moderate resolution digital video in real time without the use of image compression (with the notable exception of most configurations of High Definition Television). However, these high data rate channels will undoubtedly be more expensive to use than their lower data rate counterparts, and therefore there will still be many applications where image compression will be economically desirable. Also, compression of image data will permit multiple signals to be transmitted simultaneously over one high speed channel.

Besides applications where the transmission of images is necessary, there are also many applications where the storage of images is required. Medical X-rays and fingerprints are two examples of images that may need to be stored [11]. Computer archiving of pictures such as architectural drawings would require the storage of digital images. As mentioned above, to represent a digital image can easily require over two million bits. Even with the computer memory density available today, this storage requirement per image is impractical.

The above mentioned requirements for image transmission and storage are what make image coding necessary. The goal of image coding is to compress the image; that is, to represent the image in some way that requires as few bits as possible, without noticeably degrading the image quality. This allows images to be transmitted or stored much more efficiently.

At a high level, image data compression can be thought of as a two-step process [2], as shown in Figure 1.1. In the first step of the process a digitized image is represented by a sequence of "messages". These messages can be chosen in a wide variety of ways; however they must be chosen so that a reasonable approximation of the original image can be reconstructed from a sequence of messages. In the second step of the compression process the message sequence is coded to reduce the redundancy in the sequence. The overall goal is to generate a coded version of the image which contains all the important image information with absolutely no redundancy.

Any image compression method can be broadly classified as being either statistically-based (algebraic) or symbolically-based (structural). Statistically-based image compression methods are discussed in Section 1.3. The statistical approach to image compression is based on information theoretic principles and the methods used usually involve very localized, pixel-oriented features of the image. Due to limitations of the statistical approach to image compression which will be discussed later, a new approach to image compression is necessary if very low bit rates are to be attained. This new approach is known as symbolically-based image compression. (Some have

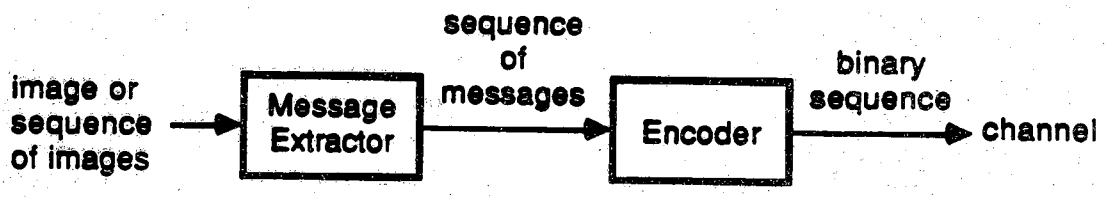


Figure 1.1. General image coder.

referred to this new approach as *second generation* image coding [2]). Symbolically-based image compression methods employ computer vision and image understanding techniques and human visual system (HVS) properties to achieve very low data rates. The geometric structure of the image scene is emphasized in symbolically-based compression methods, as opposed to the algebraic structure of the pixels used by statistically-based compression methods. In Section 1.4 we summarize the work in the developing area of symbolically-based image compression.

An image compression method can be further classified beyond the two main categories mentioned above. This further classification is based on the techniques the coding method employs, the type of image to which the coding method is applied, and the distortion the coding method introduces in the image. One such classification of coding methods is as adaptive or non-adaptive. The characteristics of an image almost always vary to some extent as the space (and/or time, for time-varying imagery) location in the image varies. To compensate for this, many compression techniques change some parameters of the coder as the space/time location in the image varies. A coder that employs such parameter variation techniques is classified as adaptive. If this type of variation is not used, the compression technique is non-adaptive. Some examples of adaptive image compression techniques are adaptive differential pulse code modulation, adaptive delta modulation, and adaptive transform coding [13-22].

Many image compression methods are implemented on a block basis. In block compression methods, the image is partitioned into non-overlapping blocks and each of these blocks is coded separately [19, 23]. Block coding is based on ideas from rate-distortion theory, which we will discuss in Section 1.3. One reason why block coding may be desirable is that dividing the image into blocks facilitates making the image compression algorithm adaptive to local image statistics. Also, by dividing up the image, coding of all blocks can be done in parallel. This is especially attractive when using a very computationally complex coding algorithm. One disadvantage of block compression techniques is that the borders of the blocks are often visible in the decoded image. Some common block image compression methods are block transform coding [24] and block truncation coding [23].

Virtually any image compression method can be applied to digital video (time-varying imagery) by applying the coding method to each of the "frames" of the image sequence. This basic approach simply codes the digital video signal as a sequence of single frame images. It is often possible to greatly reduce the data rate by exploiting the temporal redundancy that exists from frame-to-frame in the image sequence. For example, for a block coding method, three-dimensional blocks (two dimensions in space and one in time) can be used for time-varying imagery. Techniques that exploit the temporal redundancy in digital video can be quite sophisticated. One such technique is

motion-compensated coding, in which only the portions of the image that have changed from one frame to the next in the image sequence are coded [25]. Other examples of coding applied to time-varying imagery can be found in [22, 26-30].

Image compression techniques can also be applied to color images. One approach is to decompose the image into three component images (e.g. luminance, chrominance, and saturation), and then code these three images individually, using appropriate coding methods. Often, better coder performance can be obtained by exploiting the spectral and temporal redundancy in the color signals. For example, some compression techniques encode the composite NTSC color baseband video signal directly. Methods for coding color images are discussed in [14, 29-31].

One more important classification of compression techniques has to do with whether the method is distortionless or non-distortionless. If a coding method is distortionless then the decoded image is a perfect recreation of the original image. Nearly all distortionless techniques are based on information theoretic approaches and usually attain data rates in the neighborhood of two to four bits per pixel [32]. Non-distortionless coding methods introduce differences between the decoded image and the original image, but they allow much lower data rates. These distortions in the decoded image must be kept as unobtrusive as possible. An important question in image coding is how to measure the severity of the distortions caused in the image by the coding and decoding process. This and other important general issues in image coding are discussed in Section 1.2.

1.2. General Issues in Image Compression

In this section we will discuss three of the most important issues in image coder design: the image model, the image quality measure, and the impact of the application on coder design. Since for most applications, a human is the image observer, some important HVS properties will also be discussed. Obviously these are not the only important issues in image compression. Other issues worthy of consideration include coding algorithm complexity and susceptibility of coding techniques to channel errors.

1.2.1. The Image Model

In order to design a compression method that is to perform well for a class of images, some characteristics of the image must be used. That is, a model of the image must be assumed. If the model of the image is not accurate, then the compression method based on the model cannot be expected to work well. The problem of finding a

good model for a natural scene is not simple, and it is even more difficult for time-varying scenes (digital video).

Many researchers have modeled images as random fields. This approach models the pixel statistics of the image. This has proven to be difficult, due in a large part to the highly nonstationary nature of images. Image pixel statistics can change dramatically with time and spatial position in the image [11, 33]. Also, there may be information in an image that cannot be readily represented with pixel statistics. For example, the idea that a particular scene is composed entirely of triangles of different sizes and orientations is difficult to express with pixel statistics. Another fact that further complicates image modeling is that different types of images have very different pixel statistics.

Therefore, despite much work on devising pixel-based statistical models of images, success has been limited. In [11] it was observed that better statistical models might be achieved by considering the image to be the output of many sources, each with its own type of statistics. In [34] and [17] this approach is taken, and leads to results that may be among the more realistic and promising of recent statistical models of images [11].

Another promising approach to image modeling is to not model the pixel statistics of the image, but rather the statistics of some more global feature of the image, such as the edges in the image. An example of this approach is an image model generated by random tessellations of the image plane. Other examples of this type of image model can be found in [35].

Despite all the difficulties, many different image models have been devised for various applications. For a discussion of image models relevant to image coding, see [11, 36].

1.2.2. The Image Quality Measure

As stated above, every image compression technique can be classified as either distortionless or non-distortionless. With distortionless coding methods, the decoded image is identical to the original image. Therefore, a distortionless compression technique can be evaluated solely on the basis of the merits of the coding algorithm. (For example, a robust distortionless compression algorithm should have a low data rate, should require a small number of low complexity computations, and should not be susceptible to channel errors.) To fairly evaluate non-distortionless coding methods, one must be able to measure the quality of the decoded image. A measure is needed of the severity of the degradation to the original image caused by the coding and decoding

process. This distortion measure is necessarily a function of the original image and the decoded image.

The specific method used to measure the distortion in a reconstructed image can vary greatly, depending on the application. For example, in a particular application the edges in an image may be very important. In such a case it is vital that edges are unaffected by the coding and decoding process. Therefore, the image quality measure used to evaluate coding schemes for that application should weigh heavily the accuracy of the edges in the reconstructed image. In other applications, other image characteristics may be important. The characteristics of the decoded image that are important for a given application should be reflected in the image quality measure used to appraise coding schemes for that application. The development of such a measure, however, is usually not straightforward. It is often very difficult to write an analytical expression that quantifies degradation of important image characteristics.

This difficulty in quantifying the distortion of important image characteristics has led to the use of traditional mathematical measures of image quality. Two frequently used measures are the mean-squared error (mse) [37], and the root mean-squared error (rmse) [11] between the original and distorted images. The appeal of mse-based distortion measures is their simplicity, however such simple distortion measures nearly always have poor correlation with human judgement of image quality.

To improve the performance of these measures, a weighted version of mse or rmse can be used [11, 38]. The weighting function is designed to take into account variations in sensitivity to distortion of the HVS with spatial frequency. As another alternative, mse and rmse can be applied after a non-linear conversion of the image [39, 40]. The non-linear operation uses HVS properties to transform the image to the perceptual domain, where a unit change is perceptually equivalent at all points in the gray level range. The validity of the distortion calculation for this technique is limited by the validity of the non-linear transformation.

A major problem with these traditional mathematical measures of image quality is that they are pixel-based. Few pixel-wise mathematical image quality measures have consistently high correlation with human judgement of image quality. Measures that correlate well with human judgement of image quality need to take into account both local, pixel-oriented distortions in the image, and more globally-oriented image distortions [41]. Examples of various image quality measures can be found in [37, 39, 41-43].

1.2.3. The Impact of the Application

As mentioned in the previous section, a basic understanding of how the the imagery will be used is needed in order to specify an accurate image quality measure. For example, questions may need to be answered having to do with the viewing distance to the image display, the frame rate necessary for perception of motion in the scene, and whether color images are necessary. An understanding of the application is invaluable not only in relation to the image quality measure, but throughout the whole image coder design process. If we thoroughly understand the application, then we will know better how to "hide" the inaccuracies introduced into the image by the coding and decoding process. Few bits can be used to code "parts" of the image that are unimportant to the image observer, while more bits can be spent coding the parts of the image that are important to the observer. In these ways a coding scheme can be tailored to the needs of the image observer. Therefore, a crucial factor in the design of an efficient coding method is a complete understanding of the image observer for the application under consideration.

For simplicity's sake let us assume that the application we are considering dictates that the image observer is a "typical" human. Then, ideally, the image coder should use very few bits to encode the information in the image that is not important for the human viewer and use more bits to encode the information that the HVS is most sensitive to. For this reason, the more that is known about the requirements of the HVS, the better the coding method that can be designed. The HVS is very complex, and the visibility of distortion in an image is a function of many things. For example, it is a function of the nature of the distortion itself, the image intensity in a space-time neighborhood of the distortion, the lighting in the room where the image is viewed, and the "busyness" of the image in a space-time neighborhood of the distortion. If more than one distortion is introduced into an image, as is usually the case, the interplay of these multiple distortions can be very complicated. The complexity of the image observer is a major reason why image coding is so difficult a problem. But this complexity also is a key to attaining very low bit rates with image coding. Further discussion on the important role of the image observer in the image coding problem can be found in [40, 41].

Despite the complexity of the HVS, a great deal of research has been done in an effort to determine some of its basic properties. This research is based on experiments with human subjects, so the results are necessarily subjective; however much useful information has been learned. Discussions of some of the basic techniques and significant results in the area of HVS research can be found in [1, 2, 11]. The books by Marr [44] and Cornsweet [45] are useful references on human vision. In this section we will briefly summarize some of the most well established properties of the HVS [1].

One aspect of human vision that has been studied extensively is the contrast sensitivity of the eye under varying conditions [46]. Contrast sensitivity is measured by showing a subject a test pattern, and varying the intensity of neighboring regions in the test pattern until the difference in intensity is *just noticeable*. Using the configuration shown in Figure 1.2a, a simple measurement of contrast sensitivity is obtained. The observer is shown a field of uniform brightness C with a circle in the center of brightness $C + \Delta C$ [1]. The just-noticeable-difference, ΔC, is measured as a function of C. The fraction $\Delta C/C$, known as the Weber fraction, is plotted as a function of C in Figure 1.2b [1]. The Weber fraction was found to be constant at about 2 percent over a wide range (known as Weber's region). Figure 1.2b also shows that the HVS has greatly reduced contrast sensitivity in very bright or very dark intensity regions of an image. However, the configuration of Figure 1.2a is not very realistic; the test pattern shown in Figure 1.3a [1] gives results which are more useful. Again the just-noticeable-difference, ΔC, is measured, this time as a function of C_0 and C. The results of this experiment are shown in Figure 1.3b [1]. From these plots it can be seen that the eye is most sensitive to contrast in a range of about 2.2 log units, centered about the background brightness. Notice that the eye is less sensitive to contrast as C_0 moves away from C. Knowledge of the variations in the contrast sensitivity of the eye can be useful for such things as quantization of images, and human vision based image distortion measurements.

Another important characteristic of human vision is the spatio-temporal frequency response of the HVS. This response is often referred to as the modulation transfer function (MTF). The spatial and temporal responses of the HVS have often been examined separately. However, it has been found that these frequency responses are closely interrelated; therefore more recent research has dealt with the two acting in concert.

The MTF is measured by presenting a test subject with a periodic wave of some type, usually a sine-wave or a square-wave, and then varying the modulation of this wave until the threshold of visibility is determined. (The modulation of a periodic wave is the ratio of the wave's amplitude to its average value.) The value of the MTF at a particular frequency is the threshold modulation at which a stimulus of that frequency is just visible.

Depalma and Lowry investigated the spatial MTF of the HVS under varying conditions using spatially varying sinusoidal and square-wave stimuli [47]. This research did not include any temporal frequency effects. They found that, depending on the viewing conditions, the HVS responded maximally to sine-waves at retinal frequencies around 7-15 cycles/mm; with the response declining for lower and higher frequencies. Similar research has been done to determine the temporal MTF using a test pattern that varied in time. For example, in [48] experiments were done to measure the temporal

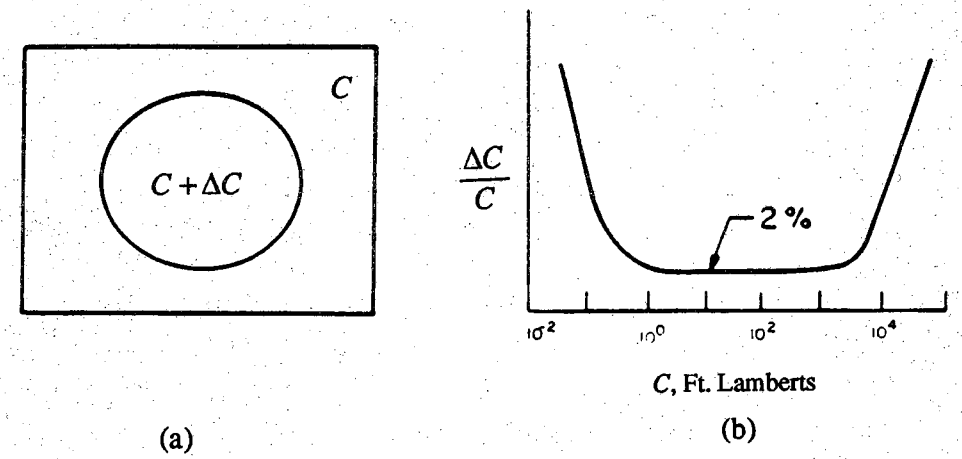


Figure 1.2. (a) Test pattern for measuring the contrast sensitivity of the HVS. (b) Contrast sensitivity of the HVS for the test pattern of Figure 1.2a (from [1]).

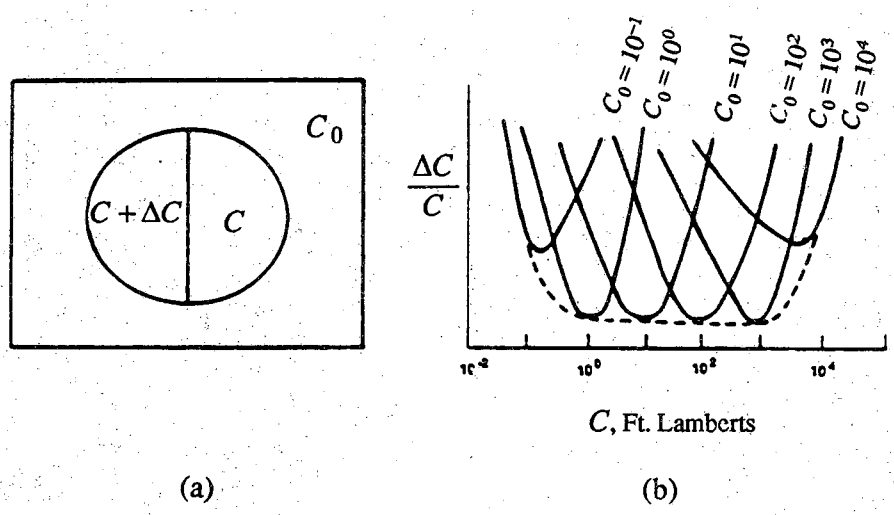


Figure 1.3. (a) Test pattern for measuring the contrast sensitivity of the HVS. (b) Contrast sensitivity of the HVS for the test pattern of Figure 1.3a (from [1]).

MTF.

In the early 1960's, the importance of the inter-connection between the spatial and temporal frequency response of the HVS was observed [48, 49]. One example of the research dealing with the two responses interacting is [50]. The spatio-temporal MTF is generally measured using either a flickering grating or a grating moving across the field of view [48, 50-52]. In [52] a sinewave moving across the field of view was used as the stimulus and the result of this research is the spatio-temporal MTF shown in Figure 1.4. The non-uniform frequency response of the HVS, as demonstrated by the spatio-temporal MTF, affects many aspects of human perception of images. For example, one consequence is that the eye is less sensitive to distortion in the parts of a scene that are moving.

There is another aspect of the time response of the HVS which is especially important for the coding of digital video. Research has shown that the human viewer takes a substantial fraction of a second to recover spatial acuity after a scene change [53]. It has been found that reducing spatial resolution for as long as .75 seconds after a scene change is not noticeable to a human observer [11].

All of the above properties help to determine the characteristic of human vision that is most important in the development of image compression techniques: the sensitivity of the human viewer to noise and distortion in images. If an absolutely complete description was known of the spatio-temporal response of the HVS, the visibility of any type of degradation in an image could be calculated and there would be no need for subjective observer tests of image quality. However, because of the complexity of human vision, we are far from any such complete description. Nonetheless, some general statements can be made about the response of the HVS to noise or distortion in images [1]: (1) Distortion is most visible in portions of the image that are constant in intensity; the more complicated a part of the image, the less visible noise will be there. That is, spatial "busyness" in an image has a masking effect on distortion. Temporal "busyness" in an image also effects the visibility of distortion, although in a more complicated way. (2) The sensitivity of the HVS to distortion varies depending on the way the distortion is correlated with the image. For example, quantization noise in an image is more annoying than a similar quantity of random noise. This fact can be unfortunate for the image coder designer, since many types of distortion introduced by the coding and decoding process are correlated with the image in ways for which the HVS has high sensitivity. (3) The HVS is more sensitive to distortion that is "structured" in some way than it is to distortion occurring randomly in the image plane. For example, the distortion that occurs along the grid that forms the block boundaries in a block compression method is more annoying to a human viewer than the same quantity of distortion distributed randomly in the image plane. (4) The sensitivity of the HVS to noise is affected by the

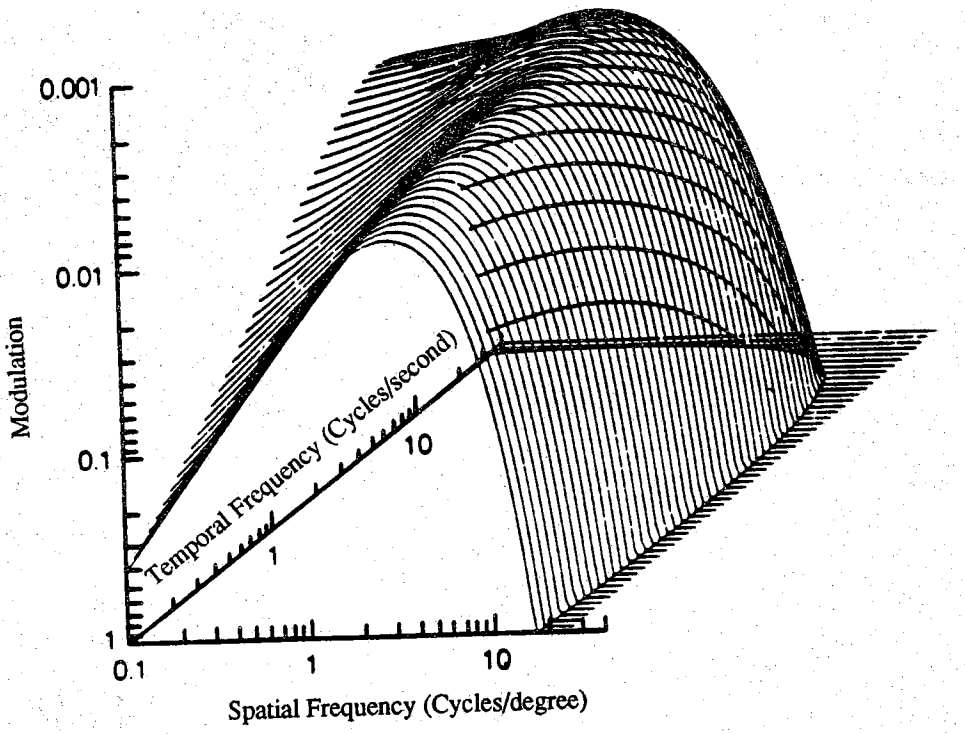


Figure 1.4. The HVS spatio-temporal modulation transfer function (from [52]).

frequency spectrum of the noise, in a complicated way which depends on the spectrum of the image. (5) The presence of any noise in an image reduces the contrast and sharpness of the image and degrades its quality significantly.

With this view of the important issues in image coding, we now proceed to present some specific image compression methods. We will discuss the statistically-based compression methods first.

1.3. Statistically-Based Image Compression Techniques

Much of the first twenty-five years of work in image compression, from approximately 1960 to the present, fits into the statistically-based category. A block diagram of the general statistical image compression system is shown in Figure 1.5. Statistically-based image coding techniques address the image compression problem from an information theoretic point of view, with the focus on eliminating the statistical redundancy among the pixels in the image.

The "ideal" preprocessor shown in Figure 1.5 is one where the pixels are mapped into independent data. For example, the mapping might be to take the Discrete Fourier Transform of the image pixels. Usually, however, the best one can do is find a preprocessor that makes the data uncorrelated. The desire for the pixels to be independent is based on rate-distortion theory. Rate distortion theory defines the optimum coder to be the coder that attains the best possible signal fidelity for a given data rate, or the coder that attains the best possible data rate for a given signal fidelity [54]. Shannon has shown that for any data source, better data rates can be achieved by coding blocks of the data, rather than individual data points. In fact, the optimal coder is achieved as $N \rightarrow \infty$, where N is the length of the block of data being coded [43, 55]. These block coders are now more popularly known as vector quantizers [54]. Obviously, a coder with infinite block length is impossible, and even a coder with reasonably long block length is difficult to design and implement. However, it can be shown that if the data samples are statistically independent, then N block length one coders are nearly as good (within about 0.25 bits/sample) as one block length N coder, for the squared error distortion measure [56]. So, if the data samples can be transformed so that they are statistically independent, then nearly optimum coder performance can be achieved with a block length one coder, i.e. a simple quantizer. The above facts form the theoretic basis for all types of statistically-based image coding.

For example, this is the reasoning behind the discrete Karhunen-Loeve transform (KLT) [24]. For Gaussian distributed pixels, the KLT transforms the data so that the samples are independent. These transformed pixels can be coded nearly optimally using a simple quantizer. Another example of this reasoning is predictive coding [57].

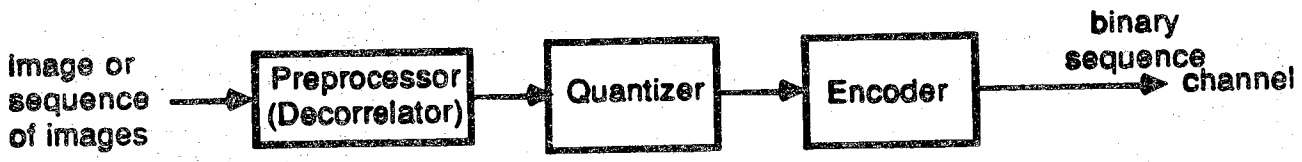


Figure 1.5. General statistically-based image compression system.

If the pixels can be modeled as a Markov random process [58], then the differences between consecutive pixels are independent. These differences can be coded nearly optimally using a simple quantizer.

Unfortunately, there are problems with the application of rate-distortion theory to image coding. In order to design the statistically-based coders discussed above two things are important: first, a valid random field model of the image is needed, and second a valid distortion measure is needed. However, as was discussed in Section 1.2.1, a simple statistical model of an image does not exist. Likewise, as discussed in Section 1.2.2, a simple distortion measure is not known for images. Much recent research has addressed issues having to do with vector quantizers such as finding efficient design methods and appropriate distortion measures.

There are many excellent reviews of statistical image compression techniques in the literature. A paper written by Schreiber in 1966 provides an interesting review of early image compression [1]. In [59], the editor presents an overall summary of the state of image compression in 1979. Netravali and Limb wrote an informative review of image compression techniques in 1980 [11], as did Jain in 1981 [38]. In addition, [38] contains an extensive bibliography of publications in image compression and related areas. In [29] a review is presented of the advances made in image compression techniques since 1981, with special emphasis placed on advances in the coding of color television and video-conference signals. In addition to these review papers, there are many books and special issues of professional journals which deal exclusively with image compression [60-63]. The above list is only a small subset of the published research in statistically-based image compression techniques.

Statistically-based image compression techniques can generally be separated into four categories: predictive coding, transform coding, interpolative and extrapolative coding, and a fourth category of miscellaneous statistically-based coding techniques [11]. A brief synopsis of each of these classes of coding techniques is given below.

1.3.1. Predictive Image Compression

The first category of statistically-based image compression techniques is predictive methods (also known as Differential Pulse Code Modulation (DPCM)) [57, 64]. The idea behind predictive image coding is to first predict the value of a pixel based on the values of a neighboring group of pixels. The group of pixels can be spatially distributed or, for digital video, temporally distributed. The error in the prediction is then quantized, coded, and transmitted. The basis of predictive techniques is that if the pixels can be modeled as a Markov process [58], then the differences between consecutive samples will be statistically independent, and a simple quantizer will be

nearly optimum. A variation of this technique is delta modulation [65]. Predictive coding results in data rates from one to two bits per pixel. Predictive coding methods can be made adaptive by varying the prediction algorithm or the difference quantizer [18-20, 66]. Adaptive predictive coding achieves bit rates ten to twenty percent lower than non-adaptive predictive coding.

1.3.2. Transform Image Compression

Another category of statistically-based image compression techniques is transform image coding methods [11, 24]. As has been mentioned, the motivation behind applying a transformation to an image before coding is to take the statistically dependent image pixels and convert them into independent transform coefficients. Unfortunately, with almost no exceptions it is impossible to obtain independent transform coefficients. However, it is sometimes possible to obtain nearly uncorrelated transform coefficients. After performing the transformation on the image pixels, the transform coefficients are quantized. The quantized values of the coefficients and the coefficients' locations are then encoded for transmission. Some examples of transforms used for image coding include Karhunen-Loeve [67], Fourier [68], Hadamard [69, 70], and Cosine [14, 27]. Bit rates of slightly less than one bit per pixel can be achieved with transform image compression methods. Transform coding can be made adaptive by varying the way the coefficients are quantized or by varying the transformation used [14, 21, 22]. These adaptive algorithms can improve the data rate by about twenty-five percent.

A disadvantage to transform coding is the number of computations required to perform the image pixel transformation. For this reason, fast transform algorithms have been developed and are often used for transform image coding [67, 71, 72]. Also, transform compression algorithms are nearly always implemented on a block-wise basis to help reduce the computation time required [24].

1.3.3. Interpolative and Extrapolative Image Compression

A third class of statistically-based image compression techniques are interpolative and extrapolative methods [11]. With these methods, a subset of the pixels is obtained by subsampling the image. This subset is then transmitted, and the decoder interpolates or extrapolates to fill in the missing pixels. The subsampling of the image can be done in either of the spatial dimensions, or in the temporal dimension, or in any combination. The interpolation function can use straight lines, or higher order polynomials. If higher

order polynomials are used in the interpolation, it may be necessary to transmit polynomial coefficients, in addition to the subset of image pixels. This class of compression techniques can be made adaptive by varying the degree to which the image is subsampled, the direction of the subsampling, or the function used to do the interpolation/extrapolation. Interpolative compression techniques achieve bit rates in the neighborhood of two bits per pixel. Examples of interpolative image compression techniques can be found in [73-75].

1.3.4. Other Statistically-Based Image Compression Techniques

Examples of some important statistically-based techniques that do not fit into any of the above categories include bit-plane coding, curve fitting methods, and run-length coding [76-80]. Some of these methods are simply one-dimensional compression methods applied to two-dimensional image signals.

1.4. Symbolically-Based Image Compression Techniques

In the last few years the bit rates that have been attained using statistically-based compression methods seemed to reach a saturation point at slightly less than one bit per pixel [2]. For many applications data rates as low as 0.01 to 0.1 bits per pixel are desirable. A new approach to image compression is necessary if these very low bit rates are to be attained. This new approach is known as symbolically-based, or "second generation" image compression. A block diagram of a general symbolic image compression system is shown in Figure 1.6.

Symbolically-based image compression methods employ techniques from image analysis, computer vision and artificial intelligence, along with HVS properties to achieve very low data rates. Global, rather than local pixel-oriented features of the image are emphasized. Examples of such global features include the size, shape, or orientation of objects in the image scene. These types of features can be used to provide a symbolic description of objects and their relationships in a scene. To obtain a complete high level description of the image scene is the ultimate goal of the "message extractor" in a symbolic image compression scheme. This symbolic description might take the form of a list of scene attributes, for example "there is a chair in the upper left corner of the scene," or "a man in a red shirt is running from left to right in the scene while turning his head and looking at the camera." Notice that these are very high level descriptions of the scene and do not deal with actual image pixel values, but with the scene content. The encoder then efficiently encodes these scene descriptions or

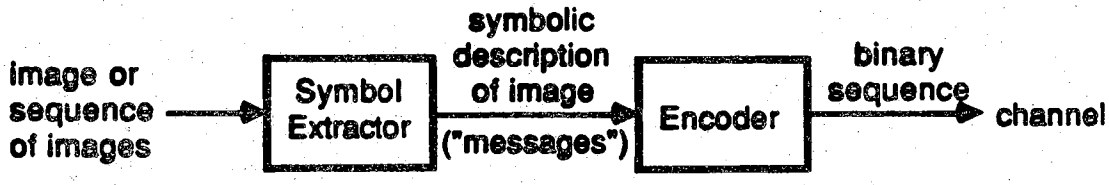


Figure 1.6. General symbolically-based image compression system.

"messages."

The current state-of-the-art in symbolic image compression does not use the complicated scene descriptions discussed above. Questions having to do with such issues as the optimal symbolic description of an image, the lowest achievable data rate for a given image, and how distortion manifests itself in the decoded image are all open research problems.

Since the symbolically-based approach is a fairly new direction in image compression, there have not been many general reviews of these types of compression methods published yet. There is, however at least one review of symbolically-based image compression techniques in the literature [2]. In addition to this paper, there is mention of some symbolically-based image compression techniques in [29] and [11].

The synthetic highs system of image compression will be discussed first in this section [81, 82]. This method is thought to be one of the earliest image compression techniques which can be classified as symbolically-based. Other symbolically-based compression techniques include segmentation-based compression, compression using fractals [83] , and subband type methods such as pyramidal compression, [84] and directional decomposition based compression [2]. Subband compression techniques [85] operate by using filters to frequency decompose the image into a series of images. These "subband" images are then coded. Compression is achieved by taking advantage of certain characteristic properties of the subband images. Several of these symbolically-based compression techniques will also be discussed in this section. In addition, Chapter Five of this thesis presents a new method of symbolically-based image compression.

1.4.1. Synthetic Highs Image Compression

The synthetic highs method was originally applied to an analog image signal. The basic idea behind the synthetic high method of image coding is to decompose the image into a high frequency component (containing edge information), and a low frequency component (containing general area brightness information). The two parts of the image are coded separately, using two different methods. By the two-dimensional sampling theorem, the low-pass component of the image can be represented with very few samples. These samples are coded to represent the low-pass component. An edge detector is used to locate edges in the original image, then the high-pass portion of the image is thresholded to determine which edge points are important. The locations and magnitudes of important edges are coded to represent the high pass component. The image is reconstructed by first using a filter to synthesize the high-pass part of the image from the edge information, and then adding to that the low-pass component of the

image. This method of coding leads to data rates slightly less than one bit per pixel. This method was first discussed in 1959 [81], and since then many other coding techniques have been proposed which make use of the same basic principle[31, 34, 86].

1.4.2. Segmentation-Based Image Compression

For segmentation-based image compression techniques [2-4, 6, 80, 82, 87-89]. the image to be compressed is first segmented. In image segmentation, the pixels in an image are divided into mutually exclusive spatial regions based on some criteria. The criteria used could be as simple as the similarity of the pixel gray levels (yielding flat image segments) [3, 6], or the criteria could be more complex, such as how well the pixels fit a given planar model (facet-based segmentation) [82], a two-dimensional polynomial model [87], or a statistical model (texture-based segmentation). After segmentation, the image consists of regions separated by contours. This segmented version of the original image is the versions that is reconstructed at the decoder.

After the image is segmented, information is coded describing the shapes and interiors of the segments. This description forms the symbolic representation for the image. In most segmentation-based compression schemes, the shapes of the image segments are represented by encoding the segment boundaries. These boundaries may be coded by approximating them with straight lines and circle segments and then coding the information describing this approximation [82], or by a more simple approach, such as coding a binary image describing where segment boundaries are located in the image [3, 6]. The interiors of the segments are represented by encoding, for example, the coefficients in the polynomial models describing each segment, or for flat segments, the average gray level of the pixels in each segment. Segmentation-based compression methods typically achieve data rates in the neighborhood of 0.5 bpp.

1.4.3. Pyramidal Image Compression

Pyramidal image compression [84] employs a hierarchical representation for the image. The representation is generated using iterative applications of the low-pass filtering idea introduced in the synthetic highs compression method described in Section 1.4.1. Pyramidal coding begins by low-pass filtering the original image, using local averaging with a unimodal Gaussian-like two-dimensional impulse response. Viewing the low-pass filtered image as a prediction of the original image, the difference between the original image and the low-pass filtered image can be interpreted as a prediction error. Clearly, coding the low-pass image and the prediction error is equivalent to

coding the original image. Compression is achieved with this representation due to two factors: (1) Since the error image is high-pass in nature, and the HVS has reduced sensitivity at high frequencies, the error image could possibly be coded with fewer bits than the original image. (2) By the two-dimensional sampling theorem, the low-pass filtered image can be represented with fewer samples than the original image.

Up to this point, the pyramidal method follows the same philosophy as synthetic highs compression. The difference in pyramidal coding is that the procedure described above is applied iteratively. Specifically, the low-pass filtered image is filtered a second time, at a lower cut-off frequency (typically half the frequency of the first filtering operation). This twice-filtered image is now a prediction for the once-filtered image, and the difference between the two filtered images is a new error image. By repeating (say n times) the low-pass filtering and differencing operations, a series of n error images can be obtained. At each iteration the dimensions of the error image are reduced (through spatial decimation) by a factor equal to the ratio of the cutoff frequencies used in that iteration and the previous iteration (typically a factor of two). The resulting error images are quantized and coded to represent the image for coding.

To generate the decoded image, interpolation filters are used to reconstruct the error images from their decimated versions. The pixel-by-pixel sum of the reconstructed error images yields the decoded image. A desirable feature of this compression technique is that it facilitates progressive reconstruction of the decoded image, and provides for convenient data rate/image quality trade-offs. Pyramidal compression typically achieves data rates in the neighborhood of 0.8 bpp.

1.4.4. Directional Decomposition Based Image Compression

Directional decomposition image compression [2] is largely motivated by the existence of direction-sensitive neurons in the HVS. In this compression technique the original image is decomposed into a series of images using filtering operations employing Gaussian windows. The entire spatial frequency plane is covered with one low-pass filter, plus a set of high-pass, directional filters. The purpose of each directional filter is to extract edges in the image with a particular spatial orientation. The filtered versions of the original image are coded to form the compressed image.

The low-pass image is coded using transform coding. Each of the directionally filtered images is spatially decimated and then represented by coding the positions and magnitudes of the edges in the decimated image. The edge positions are coded using a run-length Huffman code, and the magnitudes of the edges are quantized and coded using 3 bit codewords. This coarse quantization is possible due to the reduced contrast sensitivity of the HVS at high spatial frequencies.

To generate the decoded image, first the low-pass image is reconstructed by inversely transforming the coded coefficients. The high frequency directional edge images are reconstructed by decoding the edge information and interpolating. Once all the filtered images have been reconstructed, they are summed to form the final decoded image. Directional decomposition based compression typically achieves data rates in the neighborhood of 0.25 bpp.

As the techniques discussed above indicate, symbolic image compression techniques rely on the nature of the image scene and the relationships of objects in the scene, as described by image features such as edges and regions. These symbolically-based techniques do not rely on the statistical properties of the image pixels.

1.5 Conclusions

In this chapter we have discussed some of the important issues in image compression, and provided an overall review of past approaches to the image compression problem. We then examined a new approach, symbolically-based compression, that can lead to lower data rates than have been achieved with more traditional methods. Even with the advent of high speed, broadband channels, bandwidth will never be so cheap as to be of no economical consideration for the users of these channels. In addition, it will always be economically advantageous to store digital images using as few bits as possible. For these reasons, image coding will continue to be important for the economical storage and transmission of both large volumes of fairly conventional images, and the new breed of high definition, high quality, digital video.

CHAPTER 2
IMAGE SEGMENTATION USING HUMAN VISUAL SYSTEM PROPERTIES

In this chapter we discuss a technique for the segmentation of discrete gray level images. In image segmentation, the pixels in an image are divided into mutually exclusive spatial regions based on some criteria. Segmentation is a fundamental step in computer vision [90]. There are several approaches to segmentation, including simple thresholding, edge detection, and various forms of region growing [5, 91-94]. A great deal of work has also been done on segmentation techniques which are not based on gray level edges, such as texture-based segmentation. The output of an image segmentation scheme is usually used to identify objects in the image scene. Such identification requires a one-to-one correspondence between the image segments and the objects. This is fundamentally different from the approach we take. We are using image segmentation for compression purposes. The segmented image will be the output of a decoder (described in Chapter 5) and will be viewed by humans. This segmented image is, therefore, the "final product" of our algorithm, the decoded image. For such an application it is *not* important to have of one-to-one correspondence between objects and image segments as noted above. It is only important to design our segmentation algorithm so that image segments are allocated in a way that results in a visually pleasing segmented image. This is achieved by incorporating properties of the human visual system (HVS) at various stages in the segmentation algorithm. By using knowledge of HVS properties to guide the image segmentation, the segments can be chosen to produce a visually pleasing segmented image.

In segmentation-based image compression algorithms, information is encoded describing the segments in the segmented image. Thus, the number of image segments will determine, for the most part, the bit rate of the compressed image. For this reason, producing an image with the minimum number of segments is critical. The goal of the segmentation algorithm we propose is, for a given desired segmented image quality, to produce a segmented image which has the minimum number of image segments, allocated in a visually pleasing way.

The segmentation technique we present consists of two steps, an initial segmentation step, and a post-segmentation filtering step. The initial segmentation

algorithm uses a variation of centroid-linkage region growing [5]. This portion of the algorithm is described in Section 2.1. The second step of the image segmentation algorithm involves a filtering operation applied to the initial segmented image to determine which image segments are visually insignificant. Insignificant image segments are then merged with neighboring segments. The filtering operation is described in Section 2.2. In Section 2.3 we explore the interaction between the initial segmentation and the post-segmentation filtering steps.

Because of the wide variety of image types, different images will require different numbers of segments in order to achieve the same segmented image quality. It is often useful to know, before actual segmentation, an estimate of the number of segments that will be needed for a given image to achieve a particular image quality. In Section 2.4 we propose a quantitative measure that can be applied to an image to obtain such an estimate.

At several points in this thesis it will be necessary to measure, in some sense, the "quality" of our images. Since the images are to be viewed by humans, we would like this measure to reflect human judgement of the images' quality. However, as was discussed in Chapter 1, it is difficult to specify a quantitative measure that has consistent correspondence with human judgement of image quality. Therefore, it becomes necessary to compare images based on subjective visual quality evaluations. In this thesis, the visual quality of the images is usually determined based on careful, but nonetheless, subjective evaluation of the images by the authors. In addition, in some cases experiments have been performed using test subjects to determine the visual quality of the images. The images were observed on a DeAnza CRT monitor (manufactured by Mitsubishi Electric, model C-3910), with 512×512 pixel resolution, and 256 possible gray levels. The monitor was calibrated for a linear relationship between gray level numeric value and output luminance, using the procedure described in Appendix E.

2.1 Human Visual System Based Image Segmentation

The initial segmentation algorithm uses a variation of centroid-linkage region growing [5], and is based on a technique presented in [6, 95]. With centroid-linkage region growing, the image pixels are scanned in a raster fashion. At each pixel, there are three possible actions by which new image segments can be created, and already existing segments can be increased in size: (1) two segments neighboring each other (and the current pixel) can be merged with each other, (2) the current pixel can be merged with an already existing neighbor segment, or (3) a new segment can be created with the current pixel as its first member. Note that at any one current pixel, actions (2)

and (3) are mutually exclusive. Intensity difference thresholds are used to determine when each of the actions should be taken.

In relation to action (1) above, at each pixel the average intensities of segments neighboring the current pixel are compared to each other to determine whether any of these segments should be merged. If any two neighboring segments have average intensities within a HVS-based threshold of each other, the segments are merged to form a new, larger segment. This action is taken, for example, at the vertex of an upright "V" shaped segment. Since region growing is a raster scan method, before reaching the vertex of the "V," the two "legs" of a "V" shaped segment appear to be two separate segments. Only when the raster scan reaches the vertex of the "V" does it become apparent that the two "legs" are really parts of the same segment, and therefore should be merged.

Once all merging under action (1) above is complete, actions (2) and (3) are considered. To determine whether the current pixel should be merged with an already existing neighbor segment (action (2)), or used to start a new segment (action (3)), the intensity of the current pixel is compared to the average intensity of each of its neighbor segments. If the intensity difference between the current pixel and some neighbor segment is less than a HVS-based threshold, then the current pixel is merged with that neighbor segment, and the neighbor segment's average intensity is updated (action (2)). If the current pixel matches more than one neighbor segment, it is merged with the segment it matches best. If the current pixel does not match any of its neighbor segments, then a new segment is started with the current pixel as its first member (action (3)).

After the image has been completely divided into segments, each segment is filled in with the gray level closest to the average intensity of that segment. The result of initial image segmentation is a gray level image composed of a number of regions, each with uniform gray level.

An important reason why region growing was selected for our image segmentation is that this method is guaranteed to produce disjoint segments with closed boundaries. This will be necessary when the segmentation technique is used in the image compression algorithm we describe in Chapter 3. Other segmentation algorithms satisfy these conditions (e.g. split-and-merge [5]), and would also be acceptable for use in image segmentation for compression. A technique such as edge detection for segmentation, or segmentation by thresholding the gray levels in the image, would not be applicable in our compression algorithm, because these techniques are not guaranteed to produce closed boundaries. A second reason for selection of centroid-linkage region growing is that HVS properties can be readily incorporated into the algorithm via the segmentation thresholds.

2.1.1 Selection of the Segmentation Threshold

A key feature of the segmentation algorithm described above is the threshold used to determine when regions and pixels should be merged. We have investigated several different thresholds, some based on HVS properties. The HVS-based thresholds we propose are adapted to local intensity characteristics of the image. As the segmentation algorithm progresses spatially through the image, the segmentation threshold is varied, depending on the intensity of the image in a local area. The thresholds have all been designed for use on images with 256 gray levels, and an average gray level of 128.

The simplest threshold possible is a constant that is used for the entire image. We refer to the constant threshold as $threshold_1$.

We will attempt to incorporate HVS properties in our segmentation algorithm with the following threshold:

$$threshold_2 = (m \times \bar{p}) + d, \tag{2.1}$$

where $\bar{p}$ is the average gray level of the eight pixels neighboring the current pixel, and m and d are the slope and y-intercept, respectively, of the threshold function. The units on the threshold are gray levels. This function is an approximation of Weber's Law [45, 46], and is illustrated in Figure 2.1. (Weber's Law is discussed in more detail in Section 1.2.3.) Recall that Weber's Law says that the contrast sensitivity of the eye varies with intensity. The threshold defined above is designed to take advantage of this variation. Since the eye is less contrast sensitive in certain parts of the gray level range, it is possible to segment more coarsely (that is, using fewer, larger segments) portions of the image composed of pixels with gray levels in that range, without the coarseness of the segmentation being noticeable to a human viewer. The threshold defined in Equation 2.1 implements this idea. The threshold varies from a maximum in the highest intensity areas of the image, to a minimum in the lowest intensity areas of the image. This will result in fine segmentation (that is, with numerous small segments) in low intensity image areas (where Weber's Law says HVS contrast sensitivity is highest), and coarser segmentation in higher intensity image areas (where Weber's Law says HVS contrast sensitivity is lowest). This threshold is robust with respect to noise in an image because of the averaging operation in $\bar{p}$. The total number of segments in the segmented image will depend on m and d.

A refinement of $threshold_2$ can be made based on the fact that Weber's Law does not hold for the very highest and very lowest intensities. The new threshold is defined as follows:

$$threshold_{2a} = \begin{cases} thmin, & threshold_2 < thmin \\ threshold_2, & thmin \le threshold_2 \le thmax \\ thmax, & threshold_2 > thmax. \end{cases} \tag{2.2}$$

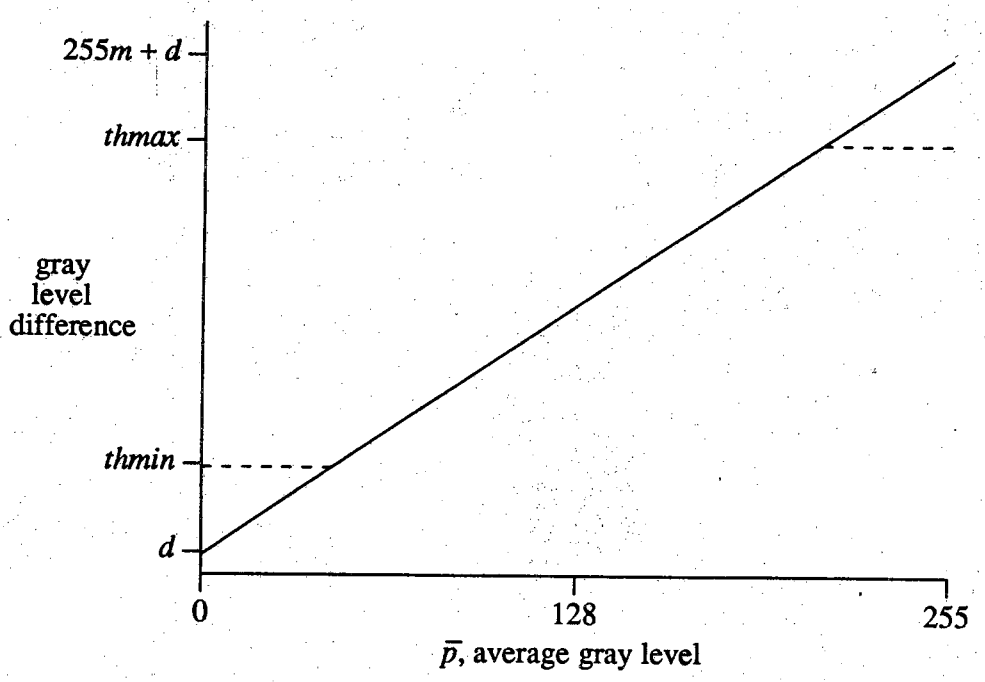


Figure 2.1. Plot of $threshold_2$ (Equation 2.1) and $threshold_{2a}$ (Equation 2.2).

With this threshold, illustrated in Figure 2.1, Weber's Law is no longer used to determine the segmentation threshold in the very highest and lowest intensity areas of the image, but rather *thmax* or *thmin*, accordingly, is used in these areas. By introducing an additional parameter in the segmentation threshold, this refinement permits further variation of the number of segments created in the image.

As a third threshold consider

$$threshold_3 = (m \times | 128 - \bar{p} |) + d. \tag{2.3}$$

This function is an approximation for another contrast sensitivity curve, which was determined by an extension of the Weber's Law experiment [59], and is illustrated in Figure 2.2 (The contrast sensitivity curve from which $threshold_3$ is modeled is discussed in Section 1.2.3.). The motivation behind this threshold is similar to that discussed in relation to $threshold_2$. This threshold is largest in the highest and lowest intensity areas of the image and smallest in the middle intensity areas of the image. The result is coarse segmentation of the image in low and high intensity areas, and finer segmentation of the image in middle intensity areas. As above, the number of segments in the segmented image can be varied by changing m or d. A refinement can also be made to $threshold_3$. Consider

$$threshold_{3a} = \begin{cases} threshold_3, & threshold_3 < thmax \\ thmax, & threshold_3 \geq thmax. \end{cases} \tag{2.4}$$

This threshold does not use the contrast sensitivity model in the very highest and lowest intensity areas of the image, but rather *thmax* is used for the segmentation threshold in these areas. This refinement, which is illustrated in Figure 2.2, also permits additional variation of the number of segments created in the image.

One further refinement of the segmentation thresholds proposed above has been considered. Since segments are generally spatially larger than single pixels it may be appropriate to apply tighter restrictions when determining if two segments should be merged than those applied when determining if a pixel should be merged with a segment. This translates to a smaller threshold for merging two segments than for merging a pixel and a segment. In terms of the description at the beginning of this section, we propose using a smaller threshold for action (1), than the threshold used for actions (2) and (3). Let w be the ratio between the action (1) threshold and the actions (2),(3) threshold (from the above discussion, $w \leq 1$). For $threshold_1$ we implement this idea via TH_1. We specify two different constant thresholds, one for merging pixels, and another for merging segments:

$$TH_1 = \begin{cases} th_{seg}, & \text{for action (1)} \\ th_{pix}, & \text{for actions (2) and (3),} \end{cases} \tag{2.5}$$

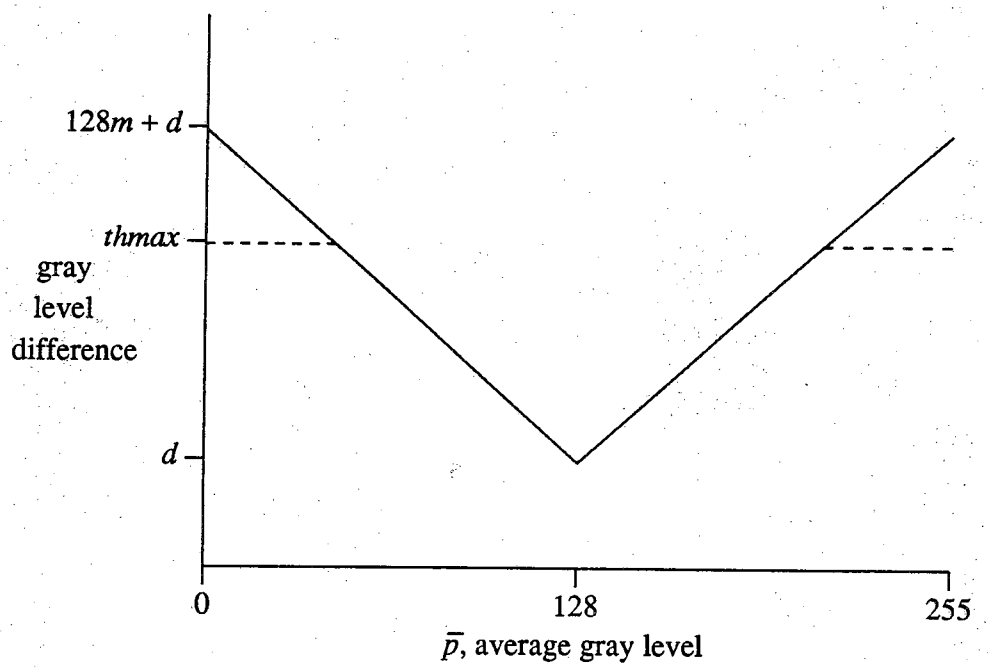


Figure 2.2. Plot of $threshold_3$ (Equation 2.3) and $threshold_{3a}$ (Equation 2.4).

In this case $w = th_{seg}/th_{pix} \leq 1$. Note that $th_{seg} = 0$ implies that no segments are ever merged; i.e. action (1) is never taken.

For $threshold_2$ and $threshold_3$ we implement this idea with the following thresholds: and

$$TH_i = \begin{cases} w \times threshold_{ia}, & \text{for action (1)} \\ threshold_{ia}, & \text{for actions (2) and (3),} \end{cases} \qquad i = 2, 3. \tag{2.6}$$

2.1.2 Experimental Results

The thresholds described above were used to segment the test images shown in Figures 2.3a-f. These test images are 256 × 256 pixels, with 256 gray levels. Histograms of all the test images are given in Appendix D. The first issue was to determine for each of the thresholds, what combination of threshold parameters (th_{pix}, th_{seg}, d, m, w, *thmax*, and *thmin*) resulted in the subjectively best visual quality segmented image, for a given number of image segments. Once the parameters for the three different segmentation thresholds (TH_1, TH_2, and TH_3) were chosen, they were compared to each other in order to determine which segmentation threshold resulted in the most visually pleasing segmented images.

First TH_1 was examined. This version of the segmenter requires two constant thresholds, th_{seg} and th_{pix}, be specified (see Equation 2.5). th_{seg} is used to decide when to merge two segments, and th_{pix} is used to decide when to merge a pixel with a segment. We wished to determine approximately what ratio, w, between th_{seg} and th_{pix} resulted in the subjectively best visual quality segmented image, for a fixed number of image segments. Examples of images compared in making this determination are shown in Figures 2.4a-d. The images in any set of Figure 2.4 (for example, 2.4a) have approximately the same number of segments, and the images in the same position in each set all have approximately the same w ratio. The exact number of segments for each image, and the values for th_{pix} and th_{seg} used to segment each image are given in the figure. Comparing images in any of the sets in Figure 2.4, it can be seen that the best visual quality segmented image for a fixed number of segments is consistently the image with w closest to 1:2. These images indicate that, when using a constant threshold for centroid-linkage region growing, a ratio of approximately 1:2 between the threshold used to decide when to merge two segments, and the threshold used to decide when to merge a pixel with a segment, produces the most visually pleasing segmented image.

The second segmentation threshold, TH_2, is given by Equation 2.6, and is illustrated in Figure 2.1. This threshold has five parameters: the slope m, the y-intercept d,

(a) Natalie

(b) Girl

(c) House

(d) Krista

(e) Eric

(f) Airpl

Figure 2.3. (a-f) Original test images. Each image is 256×256 pixels, with 256 gray levels. (a) Natalie. (b) Girl. (c) House. (d) Krista. (e) Eric. (f) Airpl.

$th_{pix} = 12$, $th_{seg} = 12$,
$w = 1.0$, # segments = 821

$th_{pix} = 14$, $th_{seg} = 7$,
$w = 0.5$, # segments = 815

$th_{pix} = 15$, $th_{seg} = 4$,
$w = 0.27$, # segments = 819

$th_{pix} = 20$, $th_{seg} = 0$,
$w = 0.0$, # segments = 877

(a)

Figure 2.4. (a-d) Images compared to determine best w ratio in TH_1. The parameters used in TH_1 and the number of segments in each image are given below each image.

$th_{pix} = 11$, $th_{seg} = 11$,
$w = 1.0$, # segments = 1003

$th_{pix} = 13$, $th_{seg} = 6$,
$w = 0.46$, # segments = 1002

$th_{pix} = 14$, $th_{seg} = 3$,
$w = 0.21$, # segments = 1029

$th_{pix} = 16$, $th_{seg} = 0$,
$w = 0.0$, # segments = 1169

(b)

Figure 2.4. (continued)

$th_{pix} = 16$, $th_{seg} = 16$,
$w = 1.0$, # segments = 1816

$th_{pix} = 19$, $th_{seg} = 8$,
$w = 0.42$, # segments = 1801

$th_{pix} = 20$, $th_{seg} = 5$,
$w = 0.25$, # segments = 1790

$th_{pix} = 23$, $th_{seg} = 0$,
$w = 0.0$, # segments = 1820

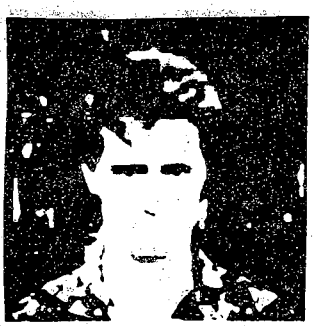
(c)

Figure 2.4. (continued)

$th_{pix} = 20$, $th_{seg} = 20$,
$w = 1.0$, # segments = 1244

$th_{pix} = 23$, $th_{seg} = 11$,
$w = 0.48$, # segments = 1286

$th_{pix} = 25$, $th_{seg} = 7$,
$w = 0.28$, # segments = 1215



$th_{pix} = 30$, $th_{seg} = 0$,
$w = 0.0$, # segments = 1285

(d)

Figure 2.4. (continued)

the maximum value *thmax*, the minimum value *thmin*, and the ratio *w*. To begin with, *w* was fixed to be 1:1. We wished to determine what slope, *m*, resulted in the subjectively best quality segmented image, for a fixed number of image segments. Three values for *m* were considered: 0.100, 0.123, and 0.246.

In order to fairly compare segmented images generated using different slopes, the images must have approximately the same number of segments. The parameters *d*, *thmin*, and *thmax* were used to control the number of segments in the segmented image for each slope. However, for a particular slope there can be several different combinations of *d*, *thmax*, and *thmin* that result in approximately the same number of segments. Therefore, before the best value for *m* could be established, the best values for *d*, *thmax*, and *thmin* had to be determined for each *m* value.

Several sets of images segmented using TH_2 are shown in Figures 2.5a-e. The images in each set (e.g. Figure 2.5a) were all generated using the same slope, and all images in a set have an approximately equal number of segments. Each image in a particular set was generated using a different combination of *d*, *thmax*, and *thmin* (the exact parameter values are given in the figure). Comparing the images in any set in Figure 2.5, one sees that the best quality image in each set is consistently the image in the upper left corner (the images numbered (*i*)). Figures 2.6a-e show plots of the segmentation thresholds used to generate, respectively, the images in Figures 2.5a-e. The threshold plots are numbered (*i*) through (*iv*) in correspondence with the images in Figures 2.5a-e. It can be seen from these plots that the images in Figure 2.5 with the best visual quality were all generated using parameters such that TH_2 was nearly constant, rather than parameters such that TH_2 realistically modeled Weber's Law.

These results indicate that a constant threshold produces better quality segmented images than a threshold modeled after Weber's Law. There is an explanation for why Weber's Law did not perform well as a segmentation threshold. Weber's Law is based on empirical data taken from a *very* simple visual stimulus (see Figure 1.2a), and describes HVS contrast sensitivity at the most basic level. Since Weber's Law describes a very low level visual process, it cannot simply be directly applied to describe HVS contrast sensitivity in the context of the complex images we are dealing with. There are numerous factors not accounted for in Weber's Law, which affect HVS contrast sensitivity. For example, Weber's Law does not take into account the masking effect of spatial "busyness" in the image on HVS contrast sensitivity. Because of the poor performance of TH_2 as a segmentation threshold, investigations to determine the most visually pleasing values for *m* and *w* in TH_2 were not performed.

The third segmentation threshold, TH_3 is described by four parameters (see Figure 2.2): the slope *m*, the y-intercept *d*, the maximum *thmax*, and the ratio *w*. We wished to determine what values of *m* and *w* resulted in the subjectively best visual quality

(i) $th_{\max} = 11$, $th_{\min} = 10$, $d = 2$, # segments = 1111

(ii) $th_{\max} = 14$, $th_{\min} = 9$, $d = 1$, # segments = 1109

(iii) $th_{\max} = 13$, $th_{\min} = 7$, $d = 2$, # segments = 1111

(iv) $th_{\max} = 20$, $th_{\min} = 8$, $d = 0$, # segments = 1105

(a)

Figure 2.5. (a-e) Images compared to determine best parameter values in TH_2. The parameters used in TH_2 and the number of segments in each image are given below each image. (a) m=.100. (b)-(d) m=.123. (e) m=.246.

(i) $th_{\max} = 11$, $th_{\min} = 10$, $d = 2$, # segments = 1067

(ii) $th_{\max} = 13$, $th_{\min} = 7$, $d = 1$, # segments = 1061

(iii) $th_{\max} = 14$, $th_{\min} = 5$, $d = 1$, # segments = 1061

(iv) $th_{\max} = 15$, $th_{\min} = 3$, $d = 1$, # segments = 1067

(b)

Figure 2.5. (continued)

(i) $th_{\max} = 12$, $th_{\min} = 11$, $d = 2$, # segments = 852

(ii) $th_{\max} = 14$, $th_{\min} = 10$, $d = 0$, # segments = 854

(iii) $th_{\max} = 16$, $th_{\min} = 8$, $d = 1$, # segments = 848

(iv) $th_{\max} = 25$, $th_{\min} = 7$, $d = 1$, # segments = 855

(c)

Figure 2.5. (continued)

(i) $th_{\max} = 23$, $th_{\min} = 22$, $d = 10$, # segments = 1005

(ii) $th_{\max} = 25$, $th_{\min} = 21$, $d = 8$, # segments = 1009

(iii) $th_{\max} = 33$, $th_{\min} = 20$, $d = 7$, # segments = 1007

(iv) $th_{\max} = 50$, $th_{\min} = 17$, $d = 1$, # segments = 1005

(d)

Figure 2.5. (continued)

(i) $th_{\max} = 23$, $th_{\min} = 22$, $d = 0$, # segments = 999

(ii) $th_{\max} = 28$, $th_{\min} = 13$, $d = 0$, # segments = 1001

(iii) $th_{\max} = 33$, $th_{\min} = 8$, $d = 0$, # segments = 994

(iv) $th_{\max} = 42$, $th_{\min} = 5$, $d = 9$, # segments = 1003

(e)

Figure 2.5. (continued)

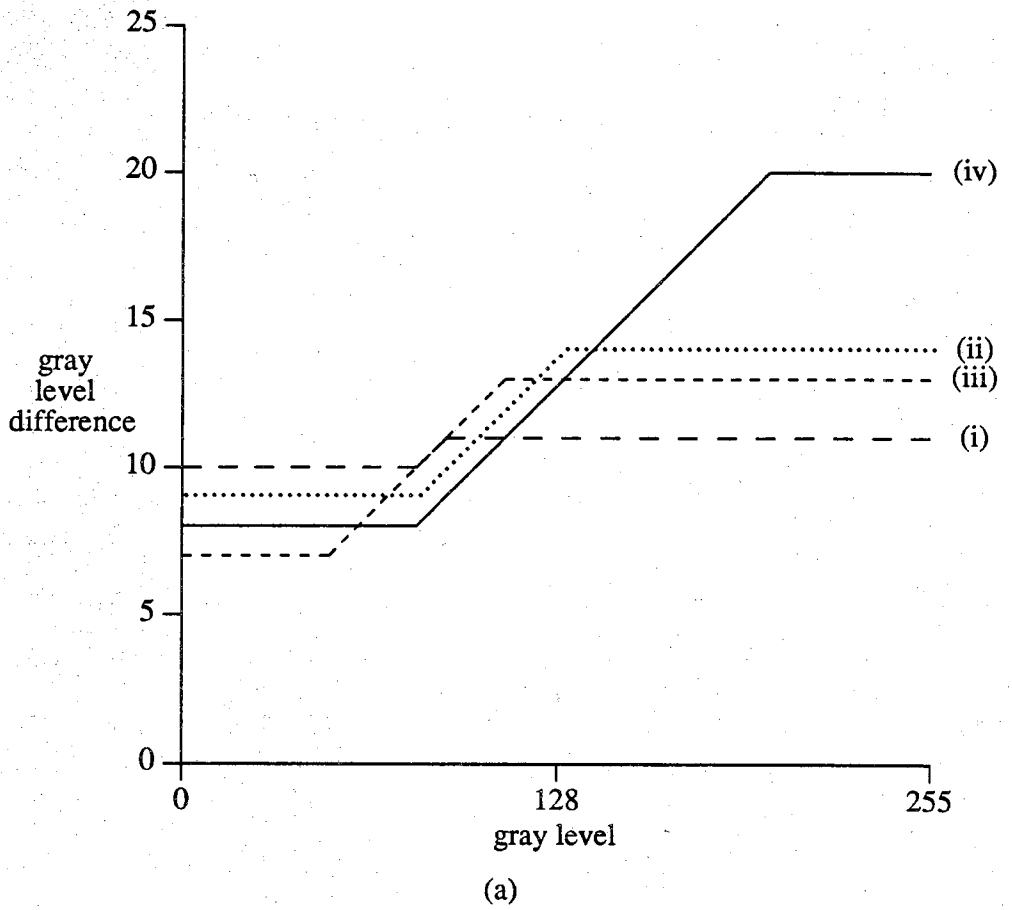
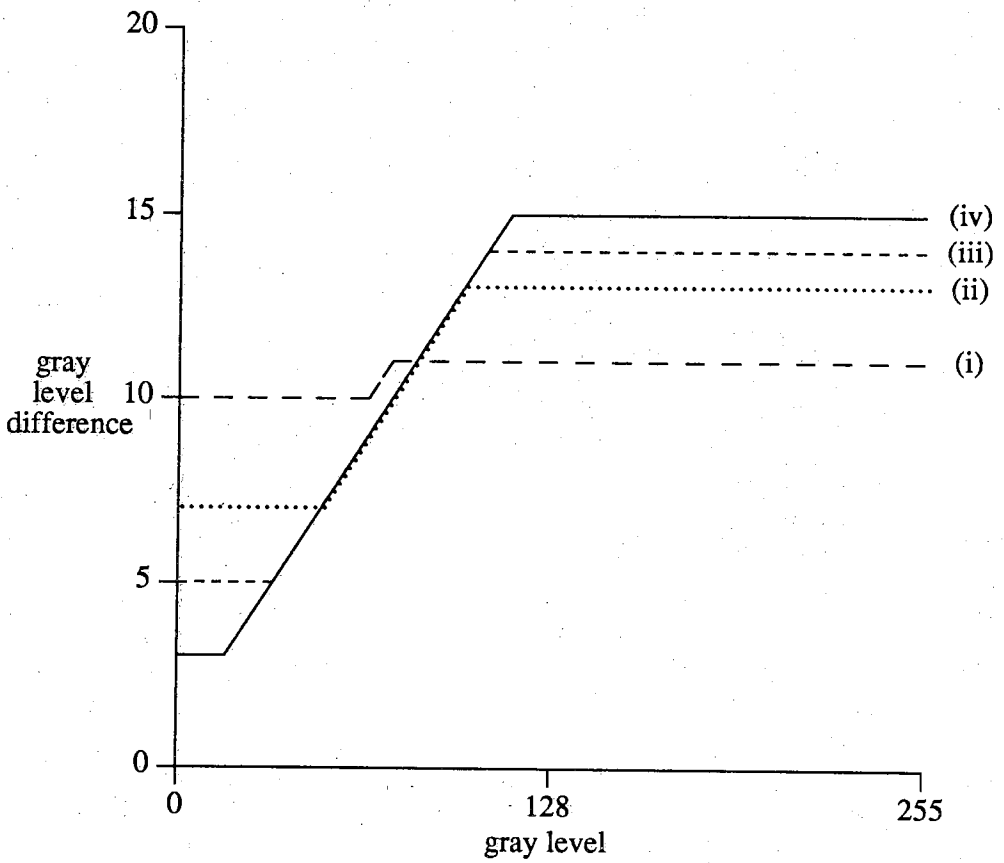
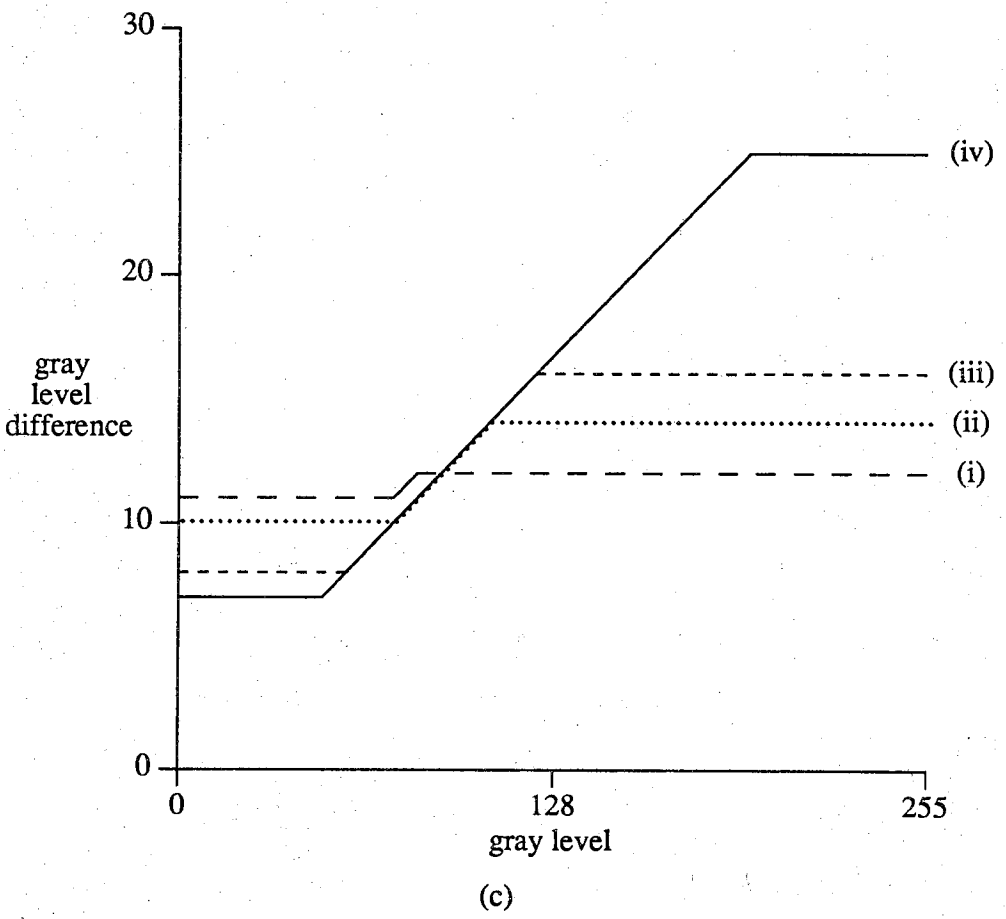


Figure 2.6. (a-e) Plots of the segmentation thresholds used to segment the images shown in Figures 2.5 (a-e).



(b)

Figure 2.6. (continued)



(c)

Figure 2.6. (continued)

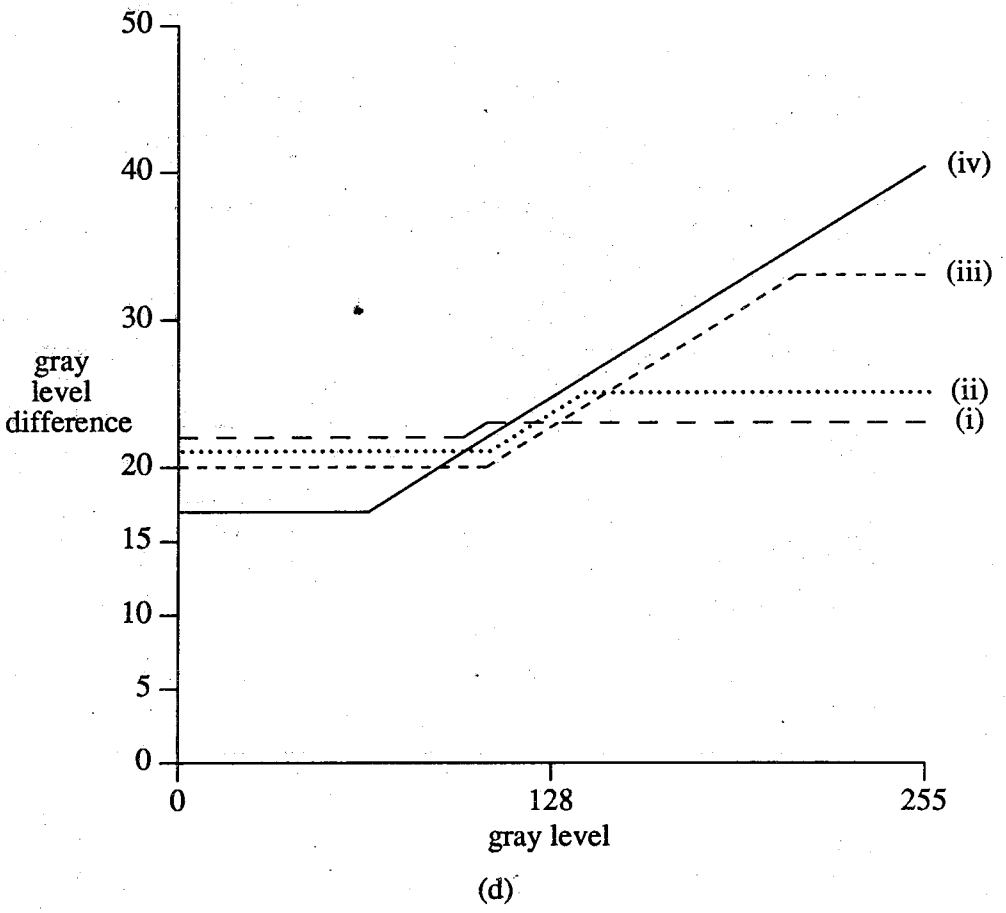
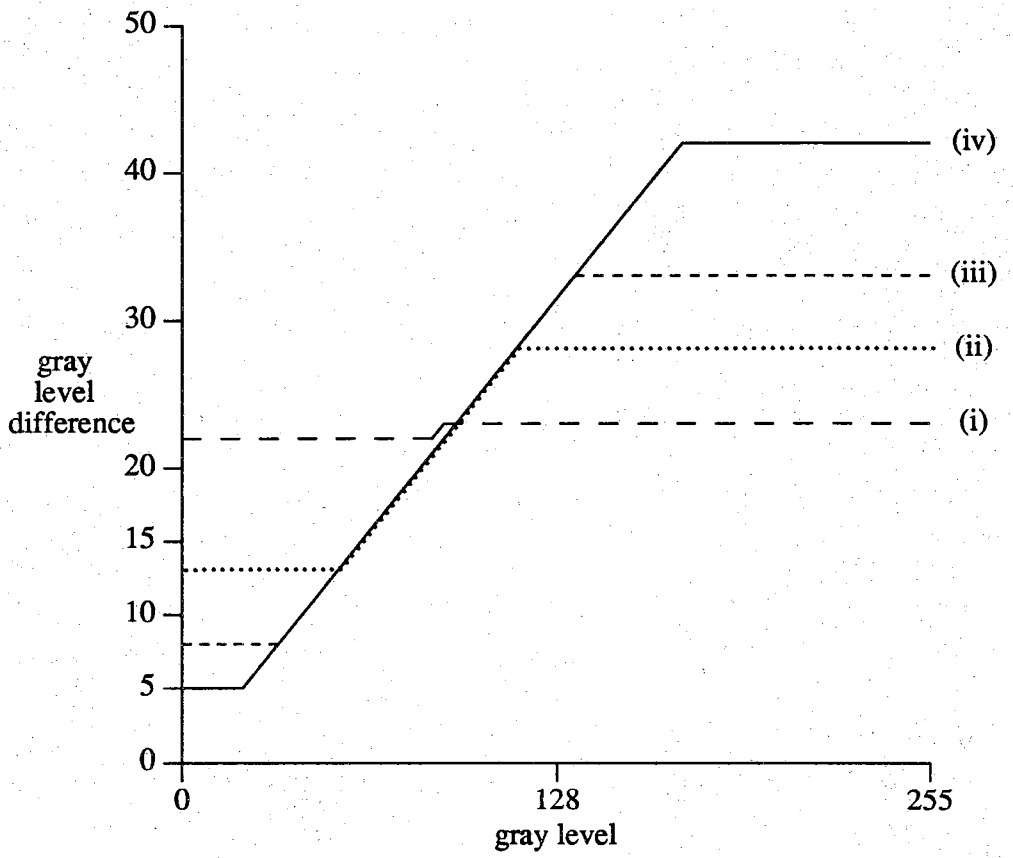


Figure 2.6. (continued)



(e)

Figure 2.6. (continued)

segmented image, for a fixed number of image segments. This determination was accomplished in two stages. In the first stage, w was held at 1:1 and only m was examined. Three values of m were considered: 0.123, 0.246, and 0.400. In order to fairly compare images generated using different values of m , it is necessary that the images have approximately the same number of segments. The parameters d and *thmax* were used to vary the number of segments in an image for a particular m.

Examples of images compared in choosing the best slope are shown in Figures 2.7a-c. The images in any particular set of Figure 2.7 all have approximately the same number of segments, and all images in a given position in the sets were generated using the same value of m in TH_3. The exact number of segments for each image, and the values for w, d, m and *thmax* for each image are given in the figure. The images in Figure 2.7 show that a slope of 0.123 results in segmented images of clearly better quality than 0.246 or 0.400. Large values of m consistently produce segmented images with low subjective visual quality.

The second variable to be examined in relation to TH_3 was w, the ratio between the segment threshold and the pixel threshold. Three values of w were considered: 1:1, 1:2, and 1:3.33. Examples of images compared in choosing the best value of w are shown in Figures 2.8a-d. The images in any given set of Figure 2.8 all have approximately the same number of segments, and the images in a given position in the subsets were all generated using the same w-ratio in TH_3. In addition, all the images in Figures 2.8a-d have $m = 0.123$, to avoid any bias in the judgement of w due to variations in m. The exact number of segments for each image, and the values of d, w, and *thmax* for each image are given in the figure. Comparing images in Figure 2.8, it can be seen that for TH_3 the best visual quality segmented image for a fixed number of segments is consistently the image with w = 1:2. From the comparisons discussed above it can be concluded that, for TH_3 the subjectively best visual quality segmented images are obtained with $m = 0.123$ and w = 1:2.

The best parameters for TH_1 and TH_3 have been determined and TH_2 has been shown to be inferior. Next, comparisons were made between segmented images generated using TH_1 and TH_3. Examples of images compared in choosing the best threshold function are shown in Figures 2.9a and b. The images across a row in either set of Figure 2.9 all have approximately the same number of segments, and the images down a column were generated using the same segmentation threshold (with various threshold parameter values). The original images before segmentation are repeated at the top of each set for comparison purposes. The exact number of segments in each image, and the parameters in the segmentation threshold used to generate each image are given in the figure. From these sets of images it can be seen that TH_3 produces as good or slightly better quality segmented images than TH_1. This is as expected because TH_1

$m = .123$, $th_{\max} = 17$,
$d = 5$, # segments = 1552

$m = .246$, $th_{\max} = 26$,
$d = 3$, # segments = 1516

$m = .400$, $th_{\max} = 15$,
$d = 2$, # segments = 1508

(a)

Figure 2.7. (a-c) Images compared to determine best value of m in TH_3. The parameters used in TH_3 and the number of segments in each image are given below each image. ($w = 1.0$ for all the images.)

$m = .123$, $th_{\max} = 14$,
$d = 8$, # segments = 800

$m = .246$, $th_{\max} = 24$,
$d = 5$, # segments = 792

$m = .400$, $th_{\max} = 16$,
$d = 4$, # segments = 795

(b)

Figure 2.7. (continued)

$m = .123$, $th_{\max} = 23$,
$d = 19$, # segments = 1033

$m = .246$, $th_{\max} = 24$,
$d = 13$, # segments = 1044

$m = .400$, $th_{\max} = 24$,
$d = 10$, # segments = 1044

(c)

Figure 2.7. (continued)

$w = 1.0$, $th_{\max} = 10$,
$d = 7$, # segments = 1445

$w = 0.5$, $th_{\max} = 11$,
$d = 8$, # segments = 1496

$w = 0.33$, $th_{\max} = 12$,
$d = 9$, # segments = 1490

(a)

Figure 2.8. (a-d) Images compared to determine best value of w in TH_3. The parameters used in TH_3 and the number of segments in each image are given below each image. (m = .123 for all the images.)

$w = 1.0$, $th_{\max} = 13$,
$d = 9$, # segments = 778

$w = 0.5$, $th_{\max} = 15$,
$d = 11$, # segments = 780

$w = 0.33$, $th_{\max} = 16$,
$d = 12$, # segments = 796

(b)

Figure 2.8. (continued)

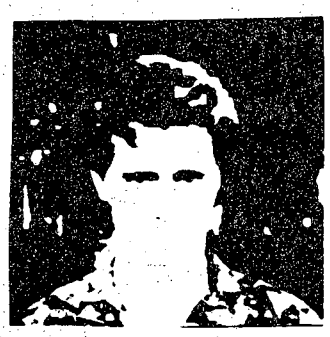
$w = 1.0$, $th_{\max} = 22$,
$d = 15$, # segments = 1255

$w = 0.5$, $th_{\max} = 25$,
$d = 18$, # segments = 1277

$w = 0.33$, $th_{\max} = 27$,
$d = 20$, # segments = 1251

(c)

Figure 2.8. (continued)



$w = 1.0$, $th_{\max} = 25$,
$d = 16$, # segments = 1077

$w = 0.5$, $th_{\max} = 29$,
$d = 20$, # segments = 1105

$w = 0.33$, $th_{\max} = 31$,
$d = 22$, # segments = 1064

(d)

Figure 2.8. (continued)

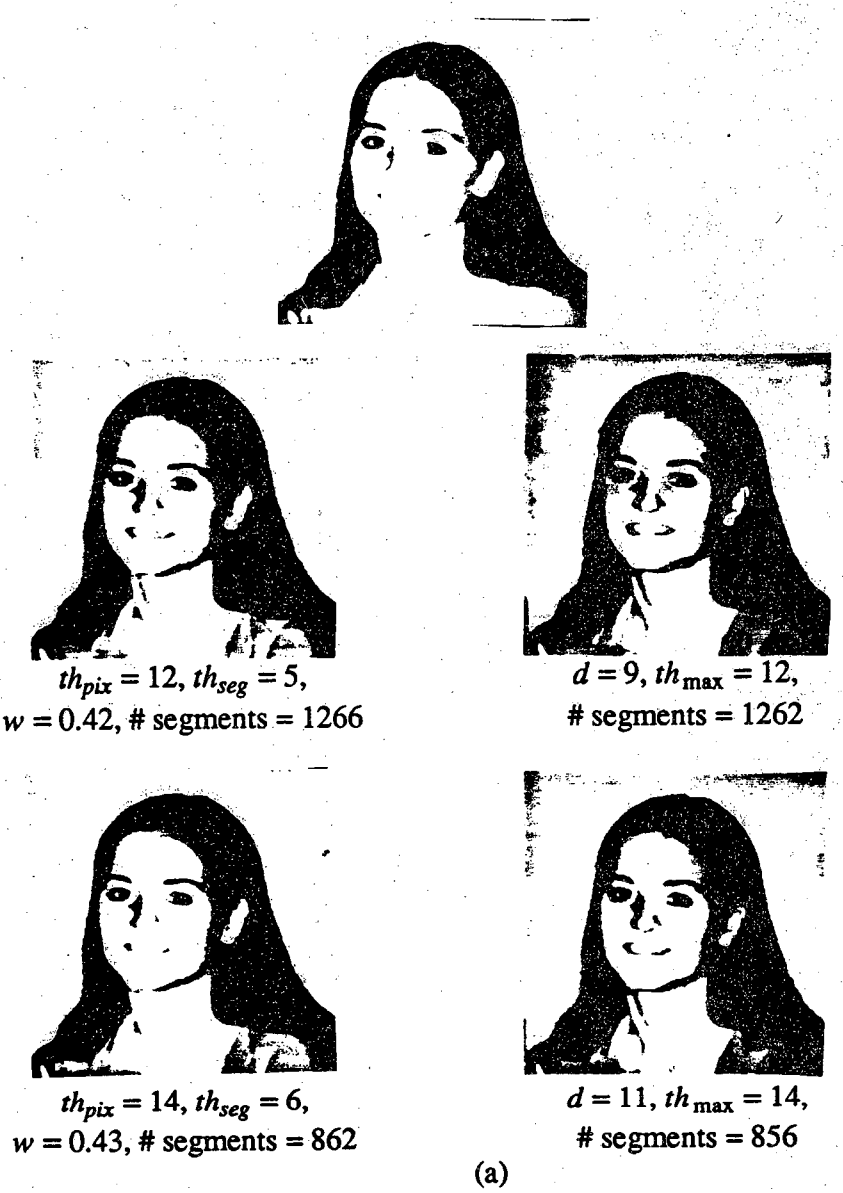
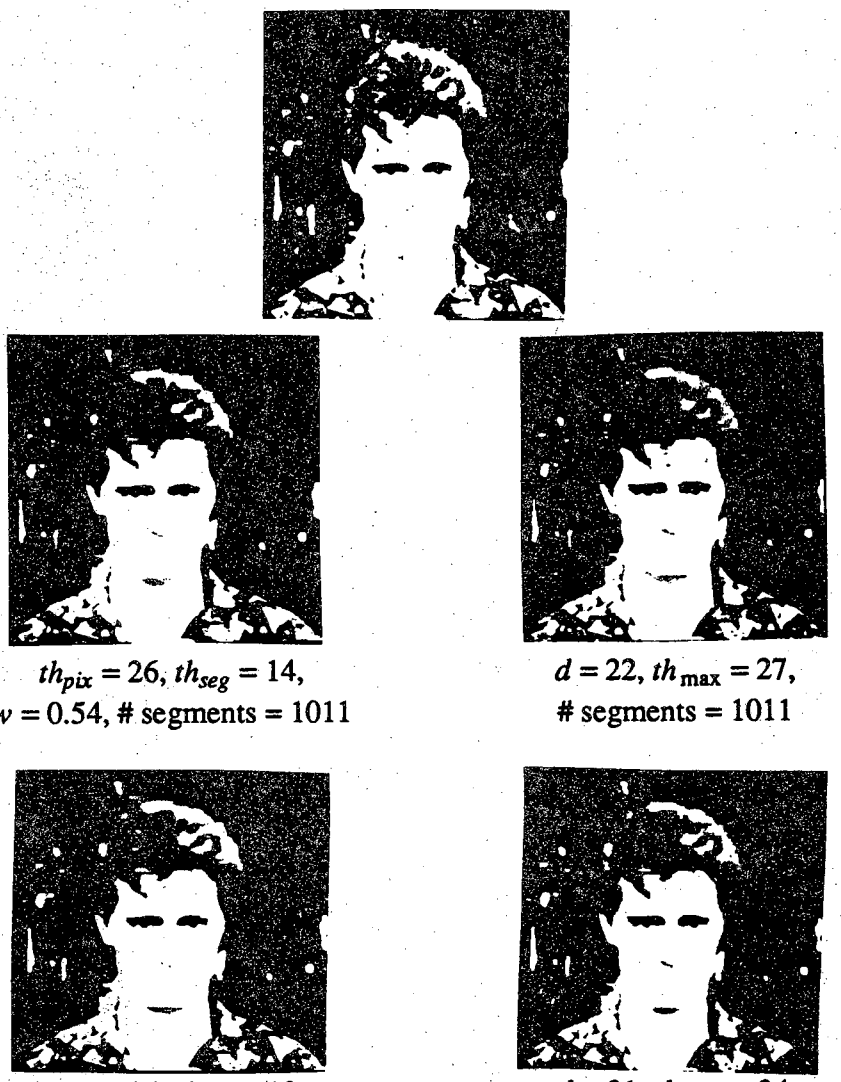


Figure 2.9. (a-b) Images used to compare TH_1 to TH_3. The original test images are shown at the top of (a) and (b). The segmented images are shown below them. The segmented images on the left were generated using TH_1 and the segmented images on the right were generated using TH_3. (The parameters used in the segmentation thresholds and the number of segments in each segmented image are given below each image. $w = 0.5$ and $m = .123$ for all the segmented images generated using TH_3.)



$th_{pix} = 26$, $th_{seg} = 14$,
$w = 0.54$, # segments = 1011

$d = 22$, $th_{\max} = 27$,
segments = 1011

$th_{pix} = 24$, $th_{seg} = 13$,
$w = 0.54$, # segments = 1162

$d = 21$, $th_{\max} = 24$,
segments = 1159

(b)

Figure 2.9. (continued)

does not in any way take advantage of the HVS properties we have discussed.

The conclusion of this investigation into an initial image segmentation algorithm is the selection of TH_3 with parameters $m = 0.123$, and $w = 1{:}2$. This segmentation threshold, which is based on HVS contrast sensitivity properties, has been systematically shown to produce better quality segmented images than the two other proposed thresholds. By using TH_3 to vary the coarseness of the image segmentation according to the intensity of the image and a model of HVS contrast sensitivity, we have designed a segmentation technique that produces, for the same number of image segments, segmented images with better visual quality than those produced using TH_1 or TH_2. TH_3 has successfully incorporated HVS properties to improve the visual quality of a segmented image.

2.2 Human Visual System Based Filtering of Segmented Images for Elimination of Visually Insignificant Segments

In this section we discuss a filtering technique for the elimination of visually insignificant segments from a segmented discrete gray level image [6]. This filtering operation, which we refer to as post-segmentation filtering, takes advantage of HVS properties relative to contrast sensitivity. The goal of post-segmentation filtering is to detect image segments that are so small or so weakly contrasted with their neighboring segments that they are insignificant to the human viewer. Such visually insignificant image segments are merged with a neighboring segment. Since post-segmentation filtering is designed to eliminate only those regions in the segmented image which are unimportant to the human viewer, the filtering operation should not degrade the visual quality of the segmented image.

In order to determine the visual significance of an image segment, some understanding of HVS contrast sensitivity is needed. The contrast sensitivity of the HVS, as a function of spatial frequency, is shown in Figure 2.10 [47]. From this it can be seen that HVS contrast sensitivity is reduced for high spatial frequencies. Therefore, high spatial frequency components of an image must have greater contrast than lower spatial frequency components, for the two to be equally noticeable to a human viewer. It can also be said, in a general sense, that the smaller an image segment, the higher in frequency is the spatial frequency content of the image near that segment. Therefore, the smaller an image segment, the more contrast is necessary between the segment and its neighbors for the segment to be visible to a human viewer. Following this reasoning, small regions are relatively less visually significant than larger regions with similar contrast. Likewise, highly contrasted regions are relatively more visually significant than lower contrasted regions of similar size. We will take advantage of this property of the HVS

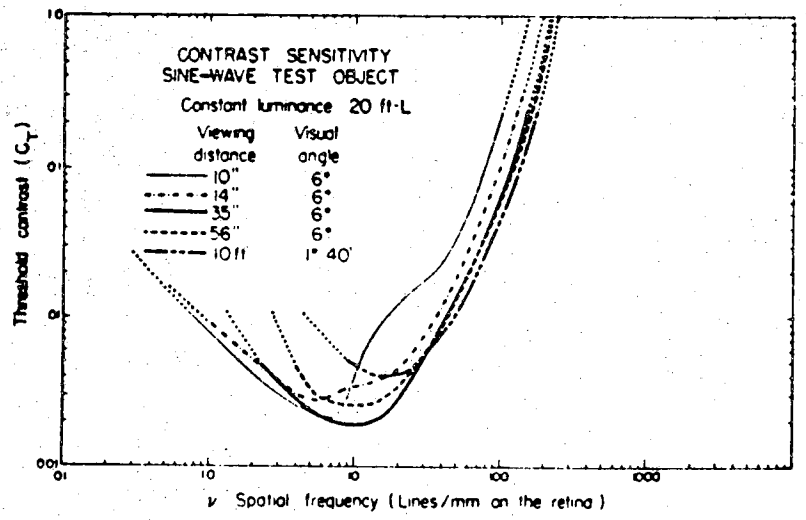


Figure 2.10. The HVS spatial frequency contrast sensitivity (from [47]).

in the design of our post-segmentation filter.

Only image segments with fewer than 16 pixels are considered as candidates for elimination. This choice of size is based on a typical viewing distance of six times the image height and image resolution of approximately 256×256 pixels [6]. For a 4×4 pixel segment, this corresponds to a spatial frequency of 3.36 cycles per degree of subtended arc. These small segments represent the high spatial frequency parts of the image, for which the HVS has reduced contrast sensitivity. The "energy" of the segment under test is measured using a technique that takes into account the size of the segment under test, and the contrast of the segment under test with its neighboring segments. The energy measurement is designed so that the energy of a segment is directly proportional to the visual significance of that segment. Highly contrasted segments should have relatively higher energy than lower contrasted segments of like size, since the more highly contrasted segments are more visually significant. Similarly, small segments should have relatively lower energy than larger segments with like contrast. The energy calculated for a segment is compared to a predefined fixed threshold to determine whether the segment is visually significant. Any segment with energy below the threshold is considered visually insignificant and is merged with the neighbor segment which has average intensity closest to the average intensity of the segment under test. The intensity of this new segment is the average intensity of the two segments which were merged. Image segments with energy above the threshold remain unchanged. The threshold can be adjusted to vary the number of segments eliminated from the image.

2.2.1 Selection of the Energy Measure

The key feature of the post-segmentation filtering technique described above is the measure used for the energy of an image segment. We have examined six different energy measures designed to model the HVS properties described above, and selected the one that resulted in the subjectively best visual quality filtered segmented image, for a given number of image segments.

Four of the energy measures considered involve applying a window operator [6] to each pixel in the region under test (a "region under test" is one of those segments with fewer than 16 pixels in the segmented image). The window, which is based on the spatial frequency contrast sensitivity of the HVS [96], is equivalent to a two-dimensional separable spatial high-pass filter, and is given by:

$$\begin{matrix} 1/16 & -1/8 & 1/16 \\ -1/8 & 1/4 & -1/8 \\ 1/16 & -1/8 & 1/16. \end{matrix}$$

The frequency transfer function of this window is shown in Figure 2.11. Let h_i be the result of applying the window at pixel i in the region under test. When the window is applied at a pixel in the interior of the region under test, so that the window is entirely contained in the region under test, h_i is zero. This is because all the pixels within a segment are the same gray level, and the filter has zero dc response. When the window is applied at a pixel near the border of the region under test, the window overlaps with segments neighboring the segment under test, and h_i may be non-zero. By spatially high-pass filtering the image, we measure the amplitude of the spatial high frequency content of the image in the neighborhood of the region under test. This indicates the amount of contrast among the high frequency components of the image in that neighborhood, and therefore the visual significance of the segment.

The first energy measure examined was proposed in [6] and is given by

$$E_1 = \frac{1}{N} \sum_i h_i^2, \tag{2.7}$$

where N is the number of pixels in the region under test, and the summation is over all i such that pixel i is in the region under test. We proposed three variations of E_1:

$$E_{1a} = \sum_i h_i^2, \tag{2.8}$$

$$E_2 = \frac{1}{N} \sum_i |h_i|, \quad \text{and} \tag{2.9}$$

$$E_{2a} = \sum_i |h_i|. \tag{2.10}$$

Finally, we proposed two other energy measures for consideration:

$$E_3 = \frac{1}{N} \sum_i |\bar{p} - p_i|, \quad \text{and} \tag{2.11}$$

$$E_{3a} = \sum_i |\bar{p} - p_i|, \tag{2.12}$$

where N is the number of pixels in the region under test, $\bar{p}$ is the average intensity of the region under test, p_i is the intensity of pixel i, and the summation is over all i such that pixel i is eight-connected to the region under test, but *not in* the region under test. These last two energy measures, rather than using the window described above, simply measure the absolute value of the difference in gray level between the region under test and its neighboring regions.

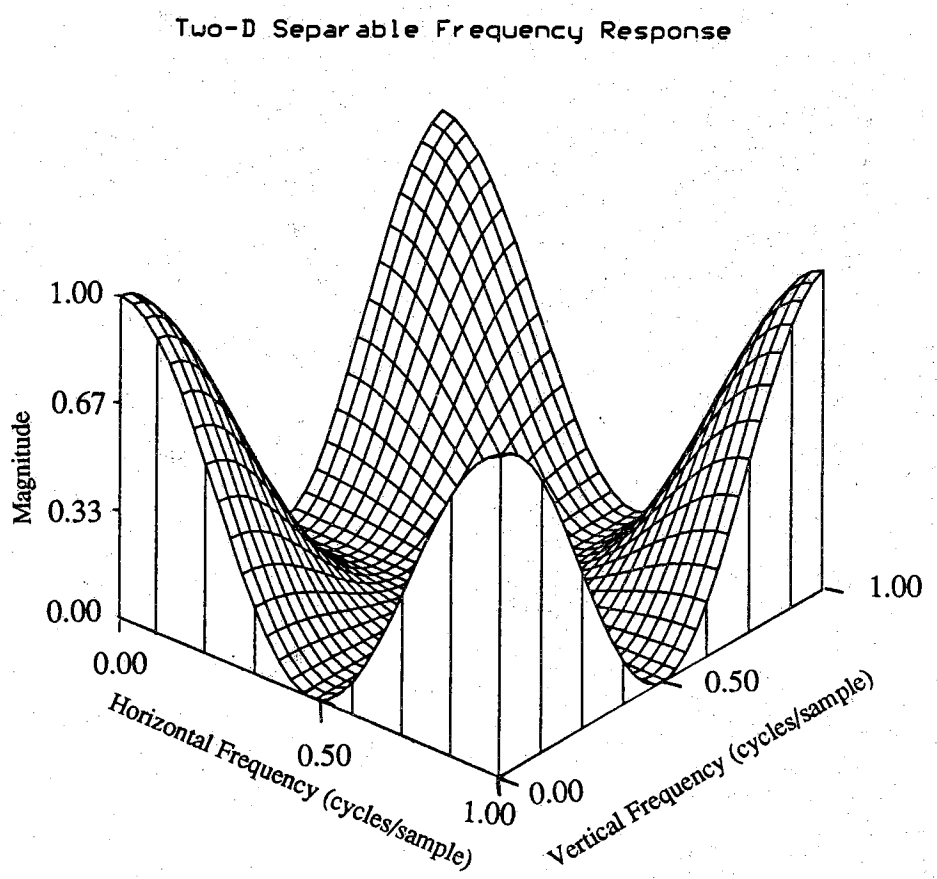


Figure 2.11. The frequency transfer function of the window given in Section 2.2.1.

2.2.2 Results

The six energy measures described above were compared using the test images from Figure 2.3. The comparisons were performed in two steps. First E_1 was compared to E_{1a}, E_2 was compared to E_{2a}, and E_3 was compared to E_{3a}. These three comparisons were to determine the effect of the averaging term ($1/N$) in the energy measures. In order to fairly compare the energy measure pairs, it was necessary to generate post-segmentation filtered images with approximately equal number of segments. The number of segments in the post-segmentation filtered images was varied by changing the energy threshold used in the filtering operation.

Figures 2.12a-c show examples of images comparing the energy measure pairs. The segmented test images (generated using TH_3) before post-segmentation filtering are shown in Figure 2.12a. The images across each row in Figure 2.12b or 2.12c all have approximately the same number of segments and were all segmented using the same segmentation threshold. The images in the left columns of Figures 2.12b and 2.12c were all post-segmentation filtered using an energy measure that included a $1/N$ factor (E_1, E_2, or E_3), and the images in the right columns were all post-segmentation filtered using an energy measure without the $1/N$ factor (E_{1a}, E_{2a}, or E_{3a}). The exact numbers of segments in the images, and the energy thresholds used for filtering the images are given in the figure. Since post-segmentation filtering mainly changes small image segments, the images in Figure 2.12 must be examined carefully in order to see any differences. However, close examination of each row reveals that the energy measures without the $1/N$ term consistently produce slightly better visual quality post-segmentation filtered images. This is most apparent in the areas around the eyes of the images in Figure 2.12b. In order to more clearly see the differences in these images, we show in Figure 2.12d enlarged versions of the eye area of the Krista image from 2.12a, and the two images in the first row of Figure 2.12b.

The following example readily illustrates a reason for the superior performance of the energy measures without the $1/N$ term. Consider E_1 and E_{1a} for the 1-valued segments in the following two simple configurations:

```
0 0 0 0 0        0 0 0 0 0 0 0
0 1 1 1 0        0 1 1 1 1 1 0
0 0 0 0 0        0 0 0 0 0 0 0.
```

The value of h_i in Equations 2.7 and 2.8 is non-zero only at the endpoints of the 1-valued segments, where $h_i = 1/8$. Since the segment on the left has only 3 pixels versus 5 pixels in the segment on the right, the value of E_1 for the segment on the left is larger than the value of E_1 for the segment on the right ($(1/3)\times(2/64) > (1/5)\times(2/64)$). However, this is not consistent with HVS spatial frequency contrast sensitivity. Since the

$m = .123, w = 1.0$
$d = 3, th_{\max} = 40,$
segments = 3069

$m = .123, w = 0.5$
$d = 9, th_{\max} = 40,$
segments = 2719

(a)

Figure 2.12. (a-d) Images used to compare energy measures with and without $1/N$ for post-segmentation filtering. (a) The original segmented test images (generated using TH_3). (b) The post-segmentation filtered versions of the Krista test image. (c) The post-segmentation filtered versions of the Eric test image. (d) Enlarged versions of the eye areas of the original segmented Krista image from Figure 2.12a, and the post-segmentation filtered images in the first row of (b). In (b) and (c) the images in the left column were post-segmentation filtered using an energy measure with a $1/N$ factor, and the images in the right column were filtered using a measure without a $1/N$ factor. (The parameters used in segmentation, the energy thresholds used in post-segmentation filtering, and the number of segments in each image are given below the images in (a-c).)

E_1, # segments = 658,
energy threshold = 16

E_{1a}, # segments = 654,
energy threshold = 50

E_2, # segments = 643,
energy threshold = 4

E_{2a}, # segments = 654,
energy threshold = 14

E_3, # segments = 653,
energy threshold = 254

E_{3a}, # segments = 653,
energy threshold = 548

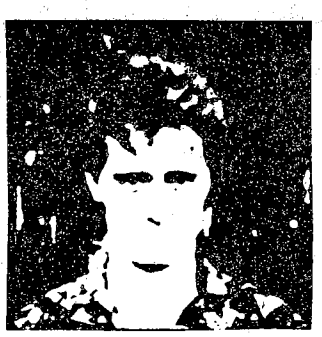
(b)

Figure 2.12. (continued)

E_1, # segments = 1047,
energy threshold = 45

E_{1a}, # segments = 1048,
energy threshold = 120

E_2, # segments = 1053,
energy threshold = 6



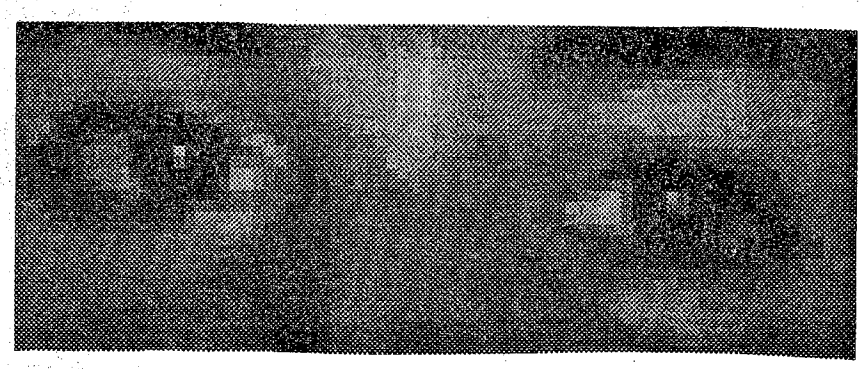
E_{2a}, # segments = 1032,
energy threshold = 16

E_3, # segments = 1018,
energy threshold = 363

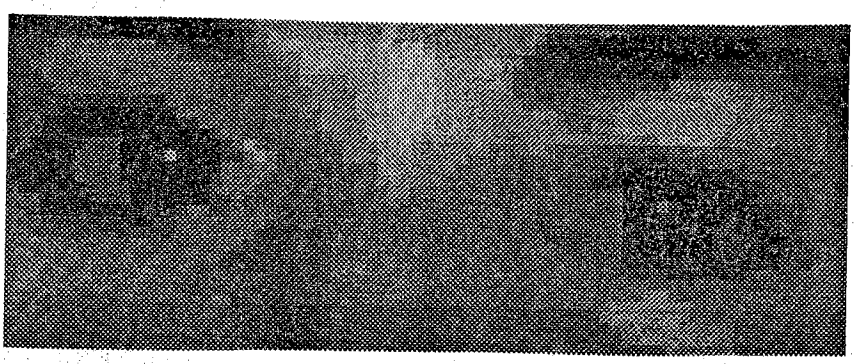
E_{3a}, # segments = 1016,
energy threshold = 680

(c)

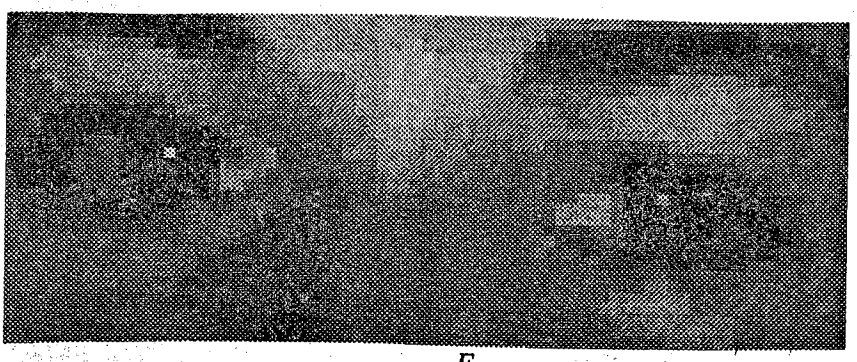
Figure 2.12. (continued)



original



E_1



E_{1a}

(d)

Figure 2.12. (continued)

two segments have like contrast and the segment on the left is smaller, the energy of the segment on the left should be, at most, less than or equal to the energy of the segment on the right. Certainly the energy of the smaller segment should not be greater than the energy of the larger segment, as it is with E_1. E_{1a} leads to visually better quality filtered segmented images because E_{1a} assigns equal energy to the two segments above, which is in better agreement with HVS spatial frequency contrast sensitivity. A similar result holds for E_2 versus E_{2a}, and E_3 versus E_{3a}. This example illustrates the general result that division by N results in a measure of average energy *per pixel* in a segment. The *total* energy in a segment is desired to measure a segment's visual significance; therefore energy measures *without* the $1/N$ term perform better for post-segmentation filtering.

The next comparison to be made is between E_{1a}, E_{2a}, and E_{3a}, to determine which of these energy measures results in the subjectively best visual quality post-segmentation filtered images. Examples of images compared in making this determination are shown in Figures 2.13a and b. In both sets of images, the segmented test image before post-segmentation filtering is shown in the upper left corner for comparison purposes. The remaining three images in each set have approximately the same number of segments, and the images in like positions in the two sets were post-segmentation filtered using the same energy measure. The exact numbers of segments for each filtered image are given in the figure. Careful examination of these images reveals that E_{2a} is slightly better at removing visually insignificant image segments, without removing visible segments. For the images Figure 2.13a, there are several visible segments in the background of the original segmented image that are not in the images filtered using E_{1a} and E_{3a}, but are preserved in the image filtered using E_{2a}. In the case of Figure 2.13b, the superiority of E_{2a} is most evident in the areas around the eyes.

The superiority of E_{2a} over E_{3a} is explained by the fact that, E_{3a} only measures the total contrast of the segment under test with its neighbor segments. E_{3a} does not take into account the spatial frequency content of the image in the neighborhood of the segment under test. For example consider E_{2a} versus E_{3a} for the 5-valued segments in the following two simple configurations:

0 1 0 1 0 1	1 0 0 0 0 0
0 5 5 5 5 1	1 5 5 5 5 0
0 1 0 1 0 1	1 1 1 1 1 0.

The value of E_{3a} is identical for these two configurations (7 × (5+4)). However, the value of E_{2a} is different for these two configurations (13/8 for the segment on the left and 9/8 for the segment on the right). Since E_{2a} takes into account the spatial frequency content of the image in the neighborhood of the segment under test, it is a better segment energy measure.

$m = .123$, $w = 1.0$,
$d = 3$, $th_{\max} = 40$,
segments = 3069

E_{1a}, # segments = 634,
energy threshold = 58

E_{2a}, # segments = 632,
energy threshold = 15

E_{3a}, # segments = 631,
energy threshold = 600

(a)

Figure 2.13. E_{3a} for post-segmentation filtering. The original segmented test images (generated using TH_3) are shown in the upper left corners. The parameters used in segmentation, the energy thresholds used in post-segmentation filtering, and the number of segments in each image are given below each image.

$m = .123, w = 0.5,$
$d = 9, th_{\max} = 40,$
segments = 2719

E_{1a}, # segments = 941,
energy threshold = 156

E_{2a}, # segments = 939,
energy threshold = 20

E_{3a}, # segments = 944,
energy threshold = 730

(b)

Figure 2.13. (continued)

The superiority of E_{2a} over E_{1a} is explained by the fact that, for certain image segment configurations, it does a slightly better job of modeling HVS spatial frequency contrast sensitivity. An example can be given of image segment configurations which illustrate this. Consider E_{1a} and E_{2a} for the 1-valued segments in the following two simple configurations:

0 0 0 0 0	0 0 0 0 0
0 0 0 0 0	0 1 1 1 0
0 1 1 1 0	0 1 1 1 0
0 0 0 0 0	0 0 0 0 0

The value of h_i in Equations 2.8 and 2.10 is non-zero only at the endpoints and corners of the 1-valued segments. The value of E_{1a} for the segment on the left is larger than the value of E_{1a} for the segment on the right (2/64 > 1/64). However, this is not consistent with HVS spatial frequency contrast sensitivity. Since the two segments have like contrast and the segment on the left is smaller, the energy of the segment on the left should be, at most, less than or equal to the energy of the segment on the right. Certainly the energy of the smaller segment should not be greater than the energy of the larger segment, as it is with E_{1a}. By contrast, E_{2a} is the same (1/4) for the two segments above. E_{2a} leads to visually better quality filtered segmented images because E_{2a} assigns equal energy to the two segments above, which is in better agreement with HVS spatial frequency contrast sensitivity.

The conclusion of this investigation into post-segmentation filtering of segmented discrete gray level images, is the selection of E_{2a} as the best measure of the visual significance of small image segments. For the images tested, this HVS contrast sensitivity based energy measure has been shown to produce better quality post-segmentation filtered images than the other proposed energy measures, for the same number of image segments.

One further relevant issue in relation to post-segmentation filtering is evaluation of its overall effectiveness at eliminating visually insignificant regions in a segmented image, without degrading the visual quality of the image. Figures 2.14a and b show two sets of segmented images. Each set consists of a segmented image before post-segmentation filtering (in the upper left corner), and versions of that segmented image after increasing degrees of post-segmentation filtering. By "degrees" of post-segmentation filtering, we refer to the number of segments removed during the filtering operation. Increasing degrees of post-segmentation filtering result in increasing numbers of segments removed from the segmented image. (Recall that the number of segments removed during post-segmentation filtering is determined by the energy threshold used.) The number of segments in each of the images is given in the figure. These images show that post-segmentation filtering can reduce the number of segments in the

$m = .123$, $w = 1.0$,
$d = 3$, $th_{\max} = 40$,
segments = 3069

E_{2a}, # segments = 813,
energy threshold = 10

E_{2a}, # segments = 583,
energy threshold = 20

E_{2a}, # segments = 516,
energy threshold = 60

(a)

Figure 2.14. (a-b) Images demonstrating post-segmentation filtering. The original segmented test images (generated using TH_3) are shown in the upper left corners. The parameters used in segmentation, the energy thresholds used in post-segmentation filtering, and the number of segments in each image are given below each image.

$m = .123$, $w = 0.5$,
$d = 4$, $th_{\max} = 8$,
segments = 2848

E_{2a}, # segments = 714,
energy threshold = 14

E_{2a}, # segments = 579,
energy threshold = 20

E_{2a}, # segments = 475,
energy threshold = 45

(b)

Figure 2.14. (continued)

image by as much as a factor of four with virtually no degradation in the quality of the segmented image. Several pairs of segmented images before and after post-segmentation filtering are shown in Figure 2.15 to further demonstrate the effectiveness of the post-segmentation filtering operation. These images demonstrate that our filtering technique is very successful at exploiting HVS properties to eliminate visually insignificant regions from a segmented image.

2.3 Interaction Between Segmentation and Post-Segmentation Filtering

An important question in relation to the segmentation algorithm described in Sections 2.1 and 2.2 has to do with the interaction between the two steps: segmentation and post-segmentation filtering. What combination of segmentation and post-segmentation filtering results in the subjectively best visual quality segmented image, for a given number of image segments? Does very fine segmentation followed by filtering which removes a large number of image segments, or coarse segmentation without any filtering, or something in between, lead to a subjectively better visual quality segmented image? This question was addressed through a series of subjective tests.

The subjective tests were performed using a variation of a method of psychophysics discussed by Stevens [46]. Stevens mentions seven different psychophysical methods:

(1) the *adjustment method*, where the subject adjusts a stimulus until it is subjectively equal to, or in some desired relation to a criterion,

(2) the *minimal change* method, where the experimenter varies the stimulus and the subject indicates its apparent relation to a criterion,

(3) the *paired comparison*, where stimuli are presented in all possible pairwise combinations, and the subject indicates which in each pair is greater with respect to a given attribute,

(4) the *constant stimuli* method, where stimuli are paired with a fixed standard and the subject indicates whether the stimulus is greater or less than the standard,

(5) the *quantal* method, where various fixed increments are added to a standard, each several times in succession, and the subject indicates the presence or absence of the increment,

(6) the *order of merit* method, where groups of stimuli are presented and the subject sets the stimuli in apparent rank order,

(7) the *rating scale* method, where the subject gives each of the stimuli an absolute rating in terms of some attribute (the rating may be numerical or descriptive).

$m = .123, w = 0.5,$
$d = 13, th_{\max} = 20,$
segments = 3474

E_{2a}, # segments = 1084,
energy threshold = 20

$m = .123, w = 0.5,$
$d = 5, th_{\max} = 12,$
segments = 4094

E_{2a}, # segments = 986,
energy threshold = 20

$m = .123, w = 0.5,$
$d = 9, th_{\max} = 40,$
segments = 2719

E_{2a}, # segments = 791,
energy threshold = 25

Figure 2.15. Images demonstrating the effectiveness of post-segmentation filtering. The images on the left are original segmented images (generated using TH_3), and the images on the right are post-segmentation filtered versions of the images. The parameters used in segmentation, the energy thresholds used post-segmentation filtering, and the number of segments in each image are given below each image.

These methods can be used to measure response to any type of sensory stimulus, for example taste, hearing or vision. This list is not exhaustive; for any method proposed for one problem, there are many variations that suggest themselves for other problems.

For our subjective tests, we use a variation on the *order of merit* method ((6) above), which incorporates an aspect of the *constant stimuli* method ((4) above). The subject is presented with sets of stimuli and is instructed to rank the stimuli in the sets, as in (6). However, the subject is at the same time given a fixed standard, as in (4), and the rankings are determined based on subjective "closeness" to the standard. We designed our method based on the question we are trying to answer. We wish to determine which of the three options for segmentation and post-segmentation filtering proposed above, results in the segmented image which subjectively looks most like the original, unsegmented image. Since we want to determine which method is "best", the *order of merit* method is appropriate. Since the judgement of "best" is based on which segmented image looks most like the original unsegmented image, the use of the original image as a standard, as in the *constant stimuli* method, is appropriate.

The subjective tests were performed using 10 test subjects, the six test images shown in Figure 2.3, and 18 test image sets. A test subject was presented with one test image set at a time. Each test image set consisted of four images: an unsegmented test image (the "standard"), and three segmented versions of that image (the "stimuli" images to be ranked). One segmented "stimuli" image was generated using coarse segmentation and no post-segmentation filtering, one was generated using slightly finer segmentation and moderate post-segmentation filtering, and the last segmented "stimuli" image was generated using very fine segmentation and extensive post-segmentation filtering. (More segments are removed from the segmented image as a result of "extensive" post-segmentation filtering than are removed as a result of "moderate" post-segmentation filtering.) All the images were segmented using TH_3 with m=.123 and w=0.5, and post-segmentation filtered using E_{2a}. The parameters d and *thmax* were adjusted to vary the coarseness of the segmentation, and the energy threshold was adjusted to vary the extent of post-segmentation filtering. The four images were arranged in a square configuration, with the unsegmented "standard" image in the upper left corner of the square. An example of a test image set is shown in Figure 2.16.

Each test image set was presented to a test subject twice, with the placement of the three segmented "stimuli" images varied. All of the segmented images in any one test image set had approximately equal number of segments; and for each test image there were two or three different test image sets, each composed of images having a different number of image segments. For example, two test image sets were constructed from the "house" test image. One set was composed of the unsegmented standard "house" image

Figure 2.16. An example of a test image set used in the subjective tests to determine the interaction between segmentation and post-segmentation filtering.

and three segmented versions of "house," all having approximately 2000 segments, and the other was composed of the unsegmented standard "house" image and three segmented versions of "house," all having approximately 1500 segments. Both of these test image sets were presented twice to each test subject, with the locations of the segmented "stimuli" images varied.

The test image sets were presented to the test subject on a DeAnza CRT monitor (manufactured by Mitsubishi Electric, model C-3910) in a darkened room. This monitor has 512×512 pixel resolution, with 256 possible gray levels. The monitor was calibrated for a linear relationship between gray level numeric value and output luminance. The calibration procedure is described in Appendix E. The test subject sat a distance of approximately six times the image height away from the screen. Each test subject was given approximately three minutes before the start of the experiment, to allow for adaption to the room's illumination (known as "dark adaption"). The test subject was instructed to rank the three segmented images in each test image set in order from the one that most closely resembled the original image, to the one that least resembled the original image. The subject was given 30 seconds to make this determination. An entire trial with one test subject took approximately 25 minutes. In order to compensate for any "learning" by a test subject, or any fatigue in a test subject during the 25 minute testing process, each subject viewed the test image sets in a different order. This variation also compensates for any additional dark adaption by the test subjects after the initial three minutes.

The raw data from the experiment described above is summarized in Appendix A. The median rankings of the three types of segmented images in each test image set, for both presentations of that test image set to all the test subjects, are given in Table 2.1. The overall median rankings of the three types of segmented images for each test image are given in Table 2.2. The coarsely segmented image with no post-segmentation filtering had a median ranking of last for all of the test image sets. The moderately segmented and filtered image had a median ranking of second for five of the six test images, and the finely segmented and extensively filtered image had a median ranking of first for five of the six test images. Overall, the coarsely segmented image was ranked last in 91 percent of the trials, the moderately filtered image was ranked second in 71 percent of the trials, and the extensively filtered image was ranked first in 74 percent of the trials. This data strongly indicates that post-segmentation filtering is very useful for removing visually insignificant image segments. For a given number of image segments, a much better visual quality segmented image is generated by doing fairly fine segmentation followed by extensive post-segmentation filtering, than by only coarsely segmenting the image.

Table 2.1 Median rankings of the three types of segmented images in each test image set, for both presentations of that test image set to all the test subjects ("psf" refers to post-segmentation filtering).

image	approx. number of segments	Median Rankings		
		coarse segmentation, no psf	medium segmentation, moderate psf	fine segmentation, extensive psf
Airpl	2684	3	2	1
	2688	3	2	1
	2911	3	2	1
Eric	1434	3	2	1
	1916	3	2	1
	2301	3	2	1
Girl	1064	3	2	1
	1273	3	2	1
	1652	3	2	1
House	2314	3	2	1
	2774	3	1	2
	2834	3	2	1
	3504	3	2	1
Krista	685	3	1	2
	948	3	1	2
Natalie	714	3	2	1
	855	3	2	1
	1076	3	2	1

Table 2.2. Overall median rankings (for both presentations, to all test subjects, for all different number of image segments) of the three types of segmented images for each test image ("psf" refers to post-segmentation filtering).

image	Median Rankings		
	coarse segmentation, no psf	medium segmentation, moderate psf	fine segmentation, extensive psf
Airpl	3	2	1
Eric	3	2	1
Girl	3	2	1
House	3	2	1
Krista	3	1	2
Natalie	3	2	1

This result can be explained by examining centroid-linkage region growing versus post-segmentation filtering. Centroid-linkage region growing is a raster scan method. Therefore, the algorithm only allows for segments to be grown from above and the immediate left of the current pixel. No segments can exist below and to the immediate right of the current pixel, so characteristics of image segments in those areas cannot be accounted for as segments are grown. Also, as the centroid-linkage algorithm progresses, the average intensity and shape of an image segment may change considerably. This means that decisions about how segments should be formed are made with incomplete information about the configuration of the segments that already exist in the image. In contrast, during post-segmentation filtering, characteristics of segments neighboring the segment under test in all directions are considered, and in addition, most of those neighbor segments are in their final form. The information available on which to base decisions about how the image should be segmented is more complete at the post-segmentation filtering stage.

2.4. A Quantitative Measure for the Number of Segments Required by an Image

In this section we propose a quantitative measure which specifies the number of image segments necessary for an image to achieve a particular segmented quality. Since the number of segments in the image plays a major role in determining the bit rate for the image, such a measure would allow estimation of the bit rate required for an image, without actually compressing the image.

The number of segments required by an image depends on two basic image characteristics: the high spatial frequency content of the image and the amount of contrast in the image. In general, for two images with similar spatial frequency content, the image with the greater contrast will require more segments. As an example of this consider two images of a sinewave grating with a particular spatial frequency. Suppose that the sinewave in the first image has several times the amplitude of the sinewave in the second image. In order to achieve the same quality in the segmented versions of these images, the first image will require more segments, because a wider range of gray levels must be represented. Likewise, for two images with similar contrast, the image with higher spatial frequency content will require more segments. As an example of this consider two images of a squarewave grating with a particular contrast. Suppose that the squarewave in the first image has several times the spatial frequency of the squarewave in the second image. Since each "stripe in the image requires a segment to represent, in order to achieve the same quality in the segmented versions of these images the first image will obviously require more segments than the second (because

there are more "stripes" in the higher frequency image). Taken together these two ideas lead to the conclusion that, in general, images with large amounts of high spatial frequency content, accompanied by high contrast, require numerous segments.

The measure we propose quantifies the combination of high contrast and high spatial frequency content in the image by measuring the average differences between neighboring pixels in the image. If the image has high contrast, differences between neighboring pixels will tend to be large. However, if the image in addition has relatively low frequency content, then these large differences will occur infrequently, and therefore the average difference will be relatively small. By contrast, if the image has high contrast accompanied by significant high frequency content, large differences will occur frequently in the image, and the average difference will be large. Therefore, by averaging the differences between neighboring pixels in the image, we obtain a measure which reflects the number of segments required by an image.

The measurement technique we propose involves taking horizontal, vertical, and diagonal "slices" through the image. The average of the absolute value of the difference between successive pixels along each slice is calculated. In terms of the ith row in an $N \times N$ image this can be written:

$$\mathcal{M}r_i = \sum_{j=0}^{N-2} |p_{i,j} - p_{i,j+1}|, \tag{2.13}$$

where $p_{i,j}$ is the gray level of the pixel in the ith row and jth column of the image. A similar expression can be written for the jth image column:

$$\mathcal{M}c_j = \sum_{i=0}^{N-2} |p_{i,j} - p_{i+1,j}|, \tag{2.14}$$

and for the two corner-to-corner image diagonals:

$$\mathcal{M}d = \sum_{i=0}^{N-2} |p_{i,i} - p_{i+1,i+1}| + |p_{i,N-1-i} - p_{i+1,N-2-i}|. \tag{2.15}$$

Then the total segment measure, $\mathcal{M}$, for an image is the average of these measures over all the rows and columns plus the two diagonals of the image:

$$\mathcal{M} = \frac{1}{(2N+2)(N-1)} \{ \mathcal{M}d + \sum_{i=0}^{N-1} \mathcal{M}c_i + \mathcal{M}r_i \}. \tag{2.16}$$

Is is also possible to use a subset of the rows and columns, for example every third row and column. This saves on computation time, however as the subset becomes smaller, the reliability of the measure may be reduced.

2.4.1 Experimental Verification

The measure described above was computed for five of the test images shown in Figure 2.3 in an effort to determine its accuracy at estimating the number of segments required by an image. This verification was difficult due to problems in determining when the segmented quality of two different segmented images was equal. Previously in this chapter, we have compared a segmented version of an image to other segmented versions of the *same* image. However, in order to evaluate the validity of $\mathcal{M}$, it is necessary to compare the quality of two *different* segmented images. In this comparison we must subjectively determine when the quality of the two different segmented images is equal. Though this determination can be made approximately, it is virtually impossible to make with any precision.

If a meaningful quantitative quality measure for segmented images existed, each image could be segmented to have a specific, precise quality measure value. Then the number of segments in each image could be compared to each image's $\mathcal{M}$ measure in order to verify $\mathcal{M}$. Since no applicable quantitative quality measure is known, we are left to subjectively evaluate the equality of the visual quality of different segmented images

Despite these difficulties, segmented versions of the test images in Figure 2.3a-e were generated having, as closely as could be determined, equal subjective visual quality. These segmented images are shown in Figure 2.17. The number of segments in each of these images, and the $\mathcal{M}$ values for each image are shown in Table 2.3. Table 2.3 also gives values for $\mathcal{M}$ calculated using several different subsets of the image rows and columns. Comparing $\mathcal{M}$ values for one image using various row and column subsets, we see that computing $\mathcal{M}$ using a few as every eighth column and row does not have a significant effect on the value of $\mathcal{M}$.

Comparing the values for $\mathcal{M}$ to the number of segments in each of the segmented images, we see that the two numbers are nearly monotonically related. The data also shows that, for the most part, the larger the difference between the number of segments in two images, the larger the difference in the $\mathcal{M}$ values for the two images. However, since this verification is based on a difficult subjective comparison of the quality of different segmented images, $\mathcal{M}$ cannot be absolutely verified as a estimator for the number of segments required by an image. $\mathcal{M}$ can only be verified to the same reliability as the segmented image quality measure used in $\mathcal{M}$'s evaluation.

$d = 7$, $th_{\max} = 10$,
segments = 551,
energy threshold = 15

$d = 5$, $th_{\max} = 8$,
segments = 690,
energy threshold = 13

$d = 7$, $th_{\max} = 10$,
segments = 780,
energy threshold = 17

$d = 10$, $th_{\max} = 13$,
segments = 871,
energy threshold = 24

$d = 13$, $th_{\max} = 16$,
segments = 898,
energy threshold = 30

Figure 2.17. The segmented and post-segmentation filtered images used to verify $\mathcal{M}$. The parameters used in segmentation, the energy thresholds used in post-segmentation filtering, and the number of segments after post-segmentation filtering are given below each image. (TH_3 with $w = 0.5$ and $m = .123$ was used to segment all the images, and E_{2a} was used to post-segmentation filter all the images.)

Table 2.3. Summary of the number of segments and the $\mathcal{M}$ values for each of the segmented images in Figure 2.17.

image	number of segments	$\mathcal{M}$ Values		
		all rows and columns	every 8-th row and column	every 32-nd row and column
Krista	551	21.958	21.866	22.512
Natalie	690	25.184	25.201	25.295
Girl	780	21.399	21.228	21.364
Eric	871	28.961	28.383	27.280
House	898	27.942	21.144	26.651

In this chapter we have described investigations into what type of segmentation threshold in the centroid-linkage region growing image segmentation algorithm generates the best visual quality segmented images with the least number of segments. From these investigations we have determined the characteristics of a HVS-based threshold which leads to the best visual quality segmented image, for a given number of image segments. We have also described a HVS-based method for filtering a segmented image to eliminate visually insignificant image segments. In both these techniques we have successfully exploited HVS properties to improve our image segmentation. Summarizing, the segmentation algorithm we have designed consists of two steps, illustrated in Figure 2.18:

(1) Centroid-linkage region growing using TH_3 (see Equations 2.3,2.4, and 2.6, and Figure 2.2), with $m = .123$ and $w = 0.5$. The parameters d and *thmax* are adjusted to control the number of image segments created.

(2) Post-segmentation filtering using energy measure E_{2a} (see Equation 2.10). The energy threshold is adjusted to control the number of segments eliminated from the image.

We have also, through a series of subjective tests, demonstrated clearly the superiority of image segmentation followed by post-segmentation filtering over image segmentation alone.

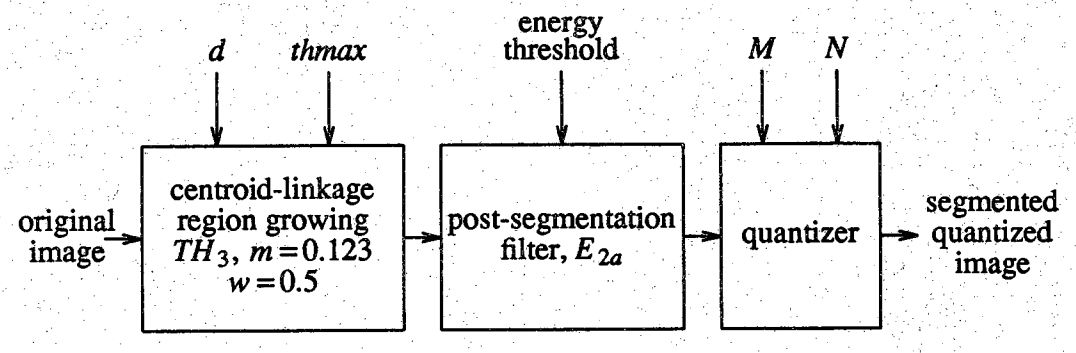


Figure 2.18. The segmentation (discussed in Chapter 2) and quantization (discussed in Chapter 3) algorithms. The parameters d and *thmax* are adjusted to control the number of segments created in the segmented image. The energy threshold is adjusted to control the number of segments eliminated from the image during post-segmentation filtering, N is the number of quantization intervals, and M is the number of gray levels in the range of the original image.

CHAPTER 3
QUANTIZATION OF SEGMENTED IMAGES

In the previous chapter we presented a human visual system (HVS) based algorithm for segmenting a gray level image. In this chapter we describe a technique for quantizing the segmented image. We show that the number of gray levels in a segmented image can be reduced significantly, with little or no degradation in the quality of the segmented image. We have found that the number of gray levels in a segmented image can typically be reduced from 256 to on the order of 20, i.e. from 8 bits per gray level to approximately 4.5 bits per gray level (a reduction of more than 40%). The quantizer we propose is designed to produce visually pleasing quantized segmented images. This is achieved by incorporating HVS properties in the process used to select the quantizer characteristics.

Other quantizers have been proposed which exploit HVS properties [29,38,97,98]. For example, quantizers which incorporate HVS properties have been proposed for use in quantization of normal (not segmented) gray level images [99-102], in differential pulse code modulation (DPCM) [103-109] and transform coding [21,22,24,110-112].

In [101] a companded quantizer was designed for gray level images that incorporates a model for HVS contrast sensitivity. The motivation was to reduce artifacts in the quantized image due to false contouring. False contours occur in an image when a smooth gray level ramp in the image is quantized and thus converted to a series of steps. In [101] a non-linear mapping, modeled after HVS contrast sensitivity, from image luminance to the perceptual quantity, brightness, is proposed. The brightness values are then uniformly quantized for minimum mean square error. An exponential probability density was assumed for the image luminance values.

In DPCM, quantizers are used for the differences between neighboring pixels. Non-uniform quantizers can be designed for these pixel differences which exploit the HVS property that sensitivity to quantization noise decreases at and adjacent to large intensity changes [105]. A function that measures intensity activity, known as a *masking function* is constructed. This masking function is incorporated into a subjective distortion measure, and a non-uniform quantizer is designed which minimizes this distortion measure. The distortion measure basically weights the quantization error at a

particular image location by the value of the masking function at that location. The result is that larger quantization errors are permitted in "busy" parts of the image, where such errors are less noticeable. In smoother parts of the image, where quantization noise is more objectionable, quantization errors are smaller.

Quantizers incorporating HVS properties have also been designed for use in transform coding techniques [21,22]. As discussed in Chapter 1, transform coding methods are typically implemented on blocks of pixels. The image is divided into blocks and the transform is applied to each block individually. The transform coefficients are then quantized. Non-uniform quantizers can be designed for the transform coefficients which exploit HVS properties based on local image characteristics. Transform coefficients which are critical to the visual quality of the image are quantized with more precision than those coefficients considered less critical.

In this chapter we design a HVS-based quantizer for the pixels in a segmented image, rather than for pixel differences or transform coefficients. The quantizer we propose is designed to produce visually pleasing quantized segmented images. This is accomplished by incorporating HVS properties relative to contrast sensitivity in the design of the quantizer. According to a model of HVS contrast sensitivity presented in Chapter 1, the HVS is most contrast sensitive in the middle of the gray level range, with the sensitivity decreasing toward the ends of the gray level range. Based on this characteristic, the quantizer we propose is non-uniform, with the spacing of the quantization thresholds varying according to the contrast sensitivity curve shown in Figure 1.3b. This will result in relatively fine quantization for mid-range gray level values, and more coarse quantization toward the extremes of the gray level range. Our quantizer does not incorporate a priori information about the image to be quantized, such as the image histogram. The only information about the image used by our quantizer is the range of gray levels in the image.

The suitability of segmented images for quantization is due to at least two factors. The first has to do with the contrast sensitivity of the eye [41,45]. HVS contrast sensitivity is a function of, among other things, spatial separation. The difference in gray level required between two test patches in order to be discernible to a human viewer varies as the spatial separation between the two patches changes. For example, it is easier to tell whether two test patches are the same gray level when they are directly adjacent to each other than when they are separated by some distance spatially. This HVS property can be exploited when quantizing a segmented image. Consider the simplified segmented image shown in Figure 3.1. Suppose that the two shaded segments have different gray levels in the original segmented image, but both gray levels fall in the same quantization interval. Then these two segments will have the same gray level after quantization. This does not cause noticeable degradation in the

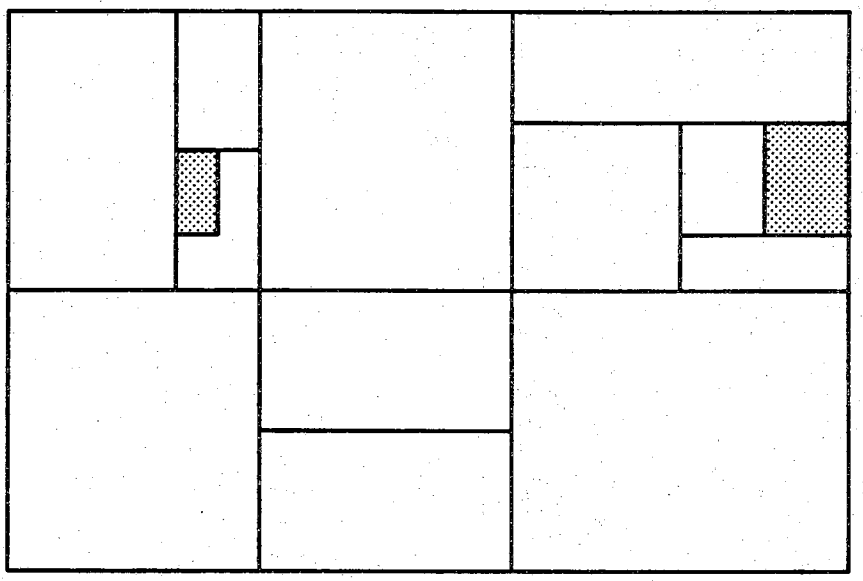


Figure 3.1. A simplified segmented image.

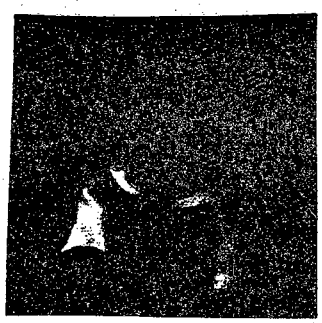
segmented image because the spatial separation of these two segments makes this change in gray level imperceptible to the human viewer.

The second factor that makes segmented images suitable for quantization has to do with the typical difference in gray level between adjacent image segments. First assume that there are an "adequate" number of quantization levels. (What constitutes "adequate" will be discussed in more detail later in this chapter.) Since segmentation divides an image into dissimilar regions, it is infrequent that two adjacent segments fall in the same quantization interval. Therefore, though two neighboring segments may both have their gray levels changed by quantization, it is unlikely that they both will be changed to the *same* gray level (and essentially be merged to form one larger segment). This is important because, assuming that an image has been segmented in such a way that each segment in the image is critical to the visual quality of the image, we would like quantization to preserve intact all the image segments. As long as quantization preserves a large number of the image segments intact, the quality of the segmented image is maintained. In summary, though quantization affects the gray level of each image segment, it preserves the contrast between segments well enough that the eye does not perceive a difference after quantization.

In Section 3.1 we present a design method for the HVS-based quantizer described above. In Section 3.2 we present the results of subjective tests comparing the performance of the proposed quantizer to a simple uniform quantizer and to a histogram-based quantizer. The tests were performed using the five segmented images shown in Figure 3.2. Histograms of these segmented images are given in Appendix D. In Section 3.3 we determine the extent of quantization possible before noticeable degradation occurs in a segmented image. Finally, Section 3.4 explores quantization of a segmented image that has been post-segmentation filtered (as described in Chapter 2), versus quantization of a segmented image that has not been post-segmentation filtered.

3.1 Human Visual System Based Quantization of Segmented Images

The design of a quantizer requires specification of quantization thresholds and quantizer output levels. A widely used approach to the specification of these parameters is classical optimum quantizer design, for example that of Max [113]. With this approach, a distortion function is defined which is a function of the quantization error. Given the quantizer input amplitude probability density, the quantizer is designed to minimize the expected value of the distortion measure. However, there is a problem with applying the methods of Max in the specific case of quantization of a segmented image. The methods of Max require specification of a distortion measure, which is difficult for images (see Section 1.2.2). Therefore, a different approach to the design of



$d = 4$, $th_{\max} = 8$, $w = 0.5$,
energy threshold = 20,
segments = 579

$d = 5$, $th_{\max} = 12$, $w = 0.5$,
energy threshold = 20,
segments = 986

$d = 13$, $th_{\max} = 20$, $w = 0.5$,
energy threshold = 20,
segments = 1084

$d = 3$, $th_{\max} = 40$, $w = 1.0$,
energy threshold = 15,
segments = 632

$d = 9$, $th_{\max} = 40$, $w = 0.5$,
energy threshold = 25,
segments = 791

Figure 3.2. Original segmented images used to compare different quantizers. The parameters used in segmentation, the energy thresholds used in post-segmentation filtering, and the number of segments in each image are given below each image. (TH_3 with $m = .123$ was used to segment all the images, and E_{2a} was used to post-segmentation filter all the images.)

a quantizer for segmented images is necessary.

We propose determination of the quantizer characteristics based on HVS contrast sensitivity. The spacing of the quantization thresholds in our quantizer will be varied according to the contrast sensitivity curve shown in Figure 1.3b. This will result in a non-uniform quantizer, with quantization thresholds densely spaced in the middle range of gray levels, and spaced further apart toward the edges of the gray level range. The quantizer design algorithm will center around determining the length of each quantization interval. The length of the intervals will be varied according to the approximation for HVS contrast sensitivity shown in Figure 3.3. The value of this approximation ranges from 4.0 to 19.8, which is a ratio of approximately 4.75. Accordingly, we will design our quantizer so that the ratio between the length of the longest quantization interval and the length of the shortest quantization interval is also approximately 4.75.

Suppose that the image to be quantized has gray levels with range M. In other words, the gray levels in the segmented image range from some gray level p, to gray level $p + M$. Also, suppose that we desire the segmented image to be quantized to N different gray levels. The values of M and N are the only input necessary for our quantizer design. Let the *unit quantization interval length*, Q, be the integer closest to M/N (Q has units of gray level). By appropriately weighting Q by a function of M and N, the quantization intervals' lengths (measured in number of gray levels) can be varied according to Figure 3.3.

There are two sets of equations which together determine the lengths of the quantization intervals, one set for N odd and one set for N even. We begin by numbering the quantization intervals, starting with 1 for the interval corresponding to the lowest input gray level values, and up to N for the interval corresponding to the highest input gray level values. It should be noted that since the quantization intervals must be integer in length, the values given by the equations below for quantization interval length are always rounded to the nearest integer.

We will first consider the case of N even. For this case, the middle two quantization intervals are specified to be $0.35Q$ gray levels in length. In other words, the length of the intervals numbered $N/2$ and $(N+2)/2$ is $0.35Q$ gray levels each. For intervals numbered between 1 and $(N-2)/2$, the length of a quantization interval is given by:

$$L_i = \frac{2.6Q}{2-N} i + (1.65 + \frac{2.6}{N-2})Q, \tag{3.1}$$

where i is the interval number, and L_i is the length of interval i, measured in number of gray levels. For intervals numbered between $(N+4)/2$ and N, the length of a quantization interval is given by:

$$L_i = \frac{2.6Q}{N-2} i + (0.35 + \frac{1.3(N+2)}{2-N})Q. \tag{3.2}$$

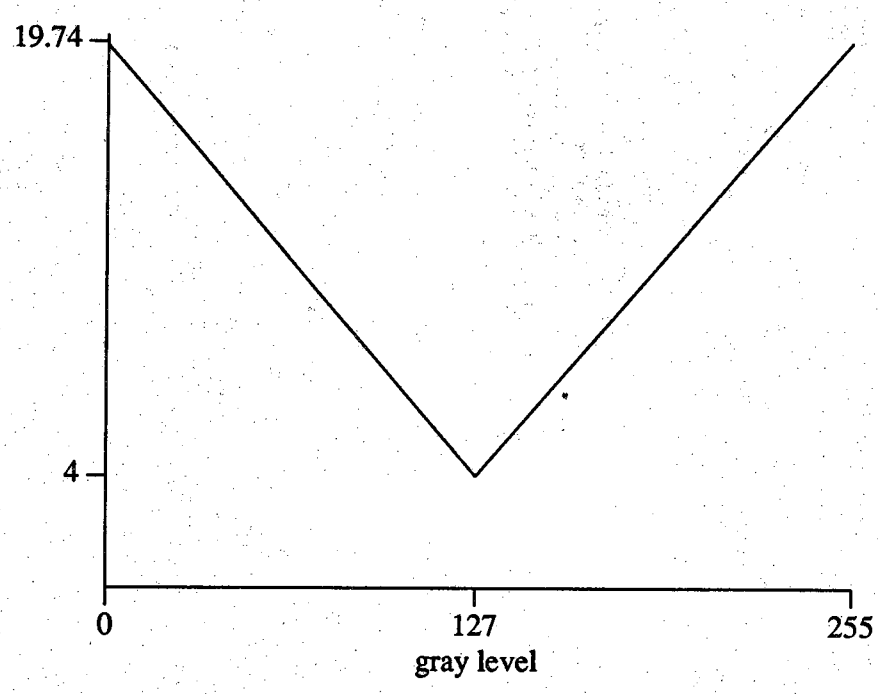


Figure 3.3. An approximation for HVS contrast sensitivity.

Next we consider the case of N odd. For this case, the middle three quantization intervals are specified to be $0.35Q$ gray levels in length. In other words, the length of the intervals numbered $(N-1)/2$, $(N+1)/2$, and $(N+3)/2$ is $0.35Q$ gray levels each. For intervals numbered between 1 and $(N-3)/2$, the length of a quantization interval is given by:

$$L_i = \frac{2.6Q}{3-N}i + (1.65 + \frac{2.6}{N-3})Q, \tag{3.3}$$

where i is the interval number, and L_i is the length of interval i, measured in number of gray levels. For intervals numbered between $(N+5)/2$ and N, the length of a quantization interval is given by:

$$L_i = \frac{2.6Q}{N-3}i + (0.35 + \frac{1.3(N+3)}{3-N})Q. \tag{3.4}$$

With these equations defined, quantization thresholds are obtained by centering the middle quantization intervals in the middle of the gray level range of the input image. For example, for N and M even, set the quantization threshold for the lower edge of quantization interval number $N/2$ to be gray level $s + (M/2) - L_{N/2}$ and the quantization threshold for the upper edge of that quantization interval to be gray level $s + (M/2) - 1$. Similarly, set the quantization threshold for the lower edge of quantization interval number $(N+2)/2$ to be gray level $s + (M/2)$ and the quantization threshold for the upper edge of that quantization interval to be gray level $s + (M/2) + L_{(N+2)/2} - 1$. By working outward and adding the quantization interval lengths given by Equations (3.1) and (3.2) (Equations (3.3) and (3.4) for N odd) to the quantization thresholds that have already been determined, the remainder of the quantization thresholds can be specified. All that remains in the design of the quantizer is to specify an output level for each of the quantization intervals. We considered two options for defining the output levels of the quantizer: the mean gray level of the pixels in each quantization interval, and the median gray level of the pixels in each quantization interval. In our experience the mean and the median were always within two or three gray levels of each other, therefore there was no noticeable difference in performance between the two. We chose to use the mean of the pixels in each of the intervals as output the levels for the quantizer. Table 3.1 shows the thresholds, output levels, and interval lengths for a typical quantizer designed using the method described above. Figure 3.4 shows a plot of this HVS-based quantizer characteristic.

Table 3.1. A HVS-based quantizer designed for the image of Figure 3.5a using the method outlined in Section 3.1 ($M = 12, N = 225$).

Bin Number	Bin Length	Gray Level Range	Output Gray Level
1	31	17-47	29
2	26	48-73	60
3	22	74-95	84
4	16	96-111	103
5	12	112-123	117
6	6	124-129	127
7	6	130-135	133
8	12	136-147	142
9	16	148-163	155
10	22	164-185	173
11	26	186-211	202
12	30	212-241	225

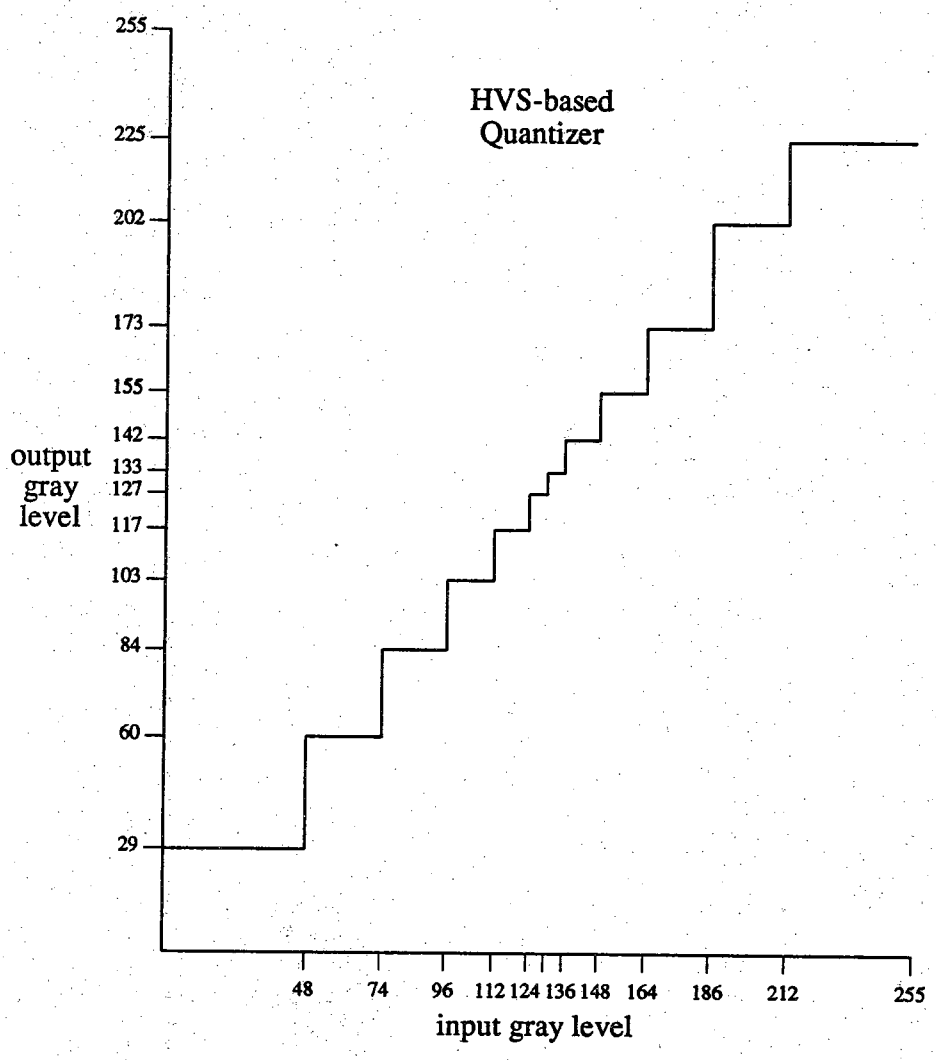


Figure 3.4. The characteristics of a HVS-based quantizer designed for the image of Figure 3.2a using the method outlined in Section 3.1.

3.2 Experimental Comparisons of Quantizers

The quantizer proposed above has been compared experimentally to both a uniform quantizer and a histogram-based quantizer. To facilitate these comparisons, the uniform quantizers and histogram based quantizers were designed for each of the segmented test images, for several different numbers of quantization levels. The quantization intervals of the uniform quantizers were uniformly spaced over the range of gray levels in the segmented images, and the output level for each interval was the average of image pixels in that interval. The histogram-based quantizers were designed based on the shapes of the histograms of the segmented images, and the output level for each interval was the average of image pixels in that interval. For M quantization intervals, the quantization thresholds were chosen by manually inspecting the image histograms and subjectively finding the $M-1$ most significant "valleys" in the histogram. The thresholds were placed in these valleys. Since these quantizers were designed by inspection, this aspect of the experiment is not precisely reproducible. However, since histogram-based quantizer design algorithms are generally heuristic in nature, the exact specifications of these quantizers is not critical in our experiment. Examples of a uniform quantizer and a histogram-based quantizer designed for the image in Figure 3.2a are shown in Figures 3.5 and 3.6.

These comparisons were accomplished through a series of psychophysical tests. As was the case with the subjective tests performed in Chapter 2, we again have three algorithms we wish to evaluate with respect to how well they preserve the quality of a standard image. Therefore, the design of these subjective tests was identical to that discussed in Chapter 2. The subjective tests were performed using eleven test subjects, and the five segmented images shown in Figure 3.2. A test subject was presented with one test image set at a time. Each test image set consisted of four images: an original segmented image (one of those shown in Figure 3.2), and three quantized versions of that image. Using the terminology of Chapter 2, the original segmented image is the "standard" image, and the three quantized versions are the "stimuli" images to be ranked. One quantized stimulus image was generated using a uniform quantizer, one was generated using a histogram-based quantizer, and the last was generated using the HVS-based quantizer described above. The four images were arranged in a square configuration, with the original segmented "standard" image in the upper left corner of the square. Each test image set was presented to a test subject twice, with the placement of the three quantized segmented "stimuli" images varied. All of the quantized segmented "stimuli" images in any one test image set were quantized to the same number of gray levels. For each of the segmented "standard" images shown in Figure 3.2, there were several different test image sets, each composed of images quantized to a different number of gray levels. For example, two test image sets were constructed from the

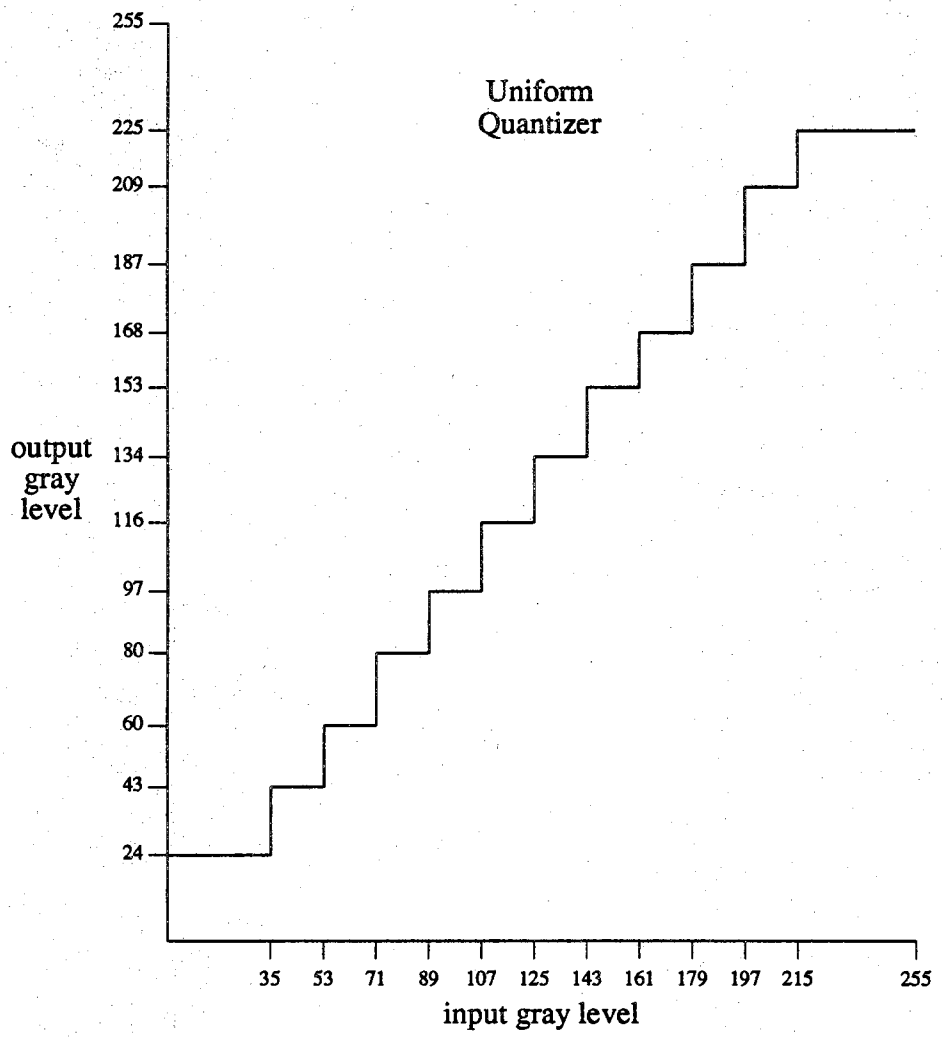


Figure 3.5. The characteristics of a uniform quantizer designed for the image of Figure 3.2a.

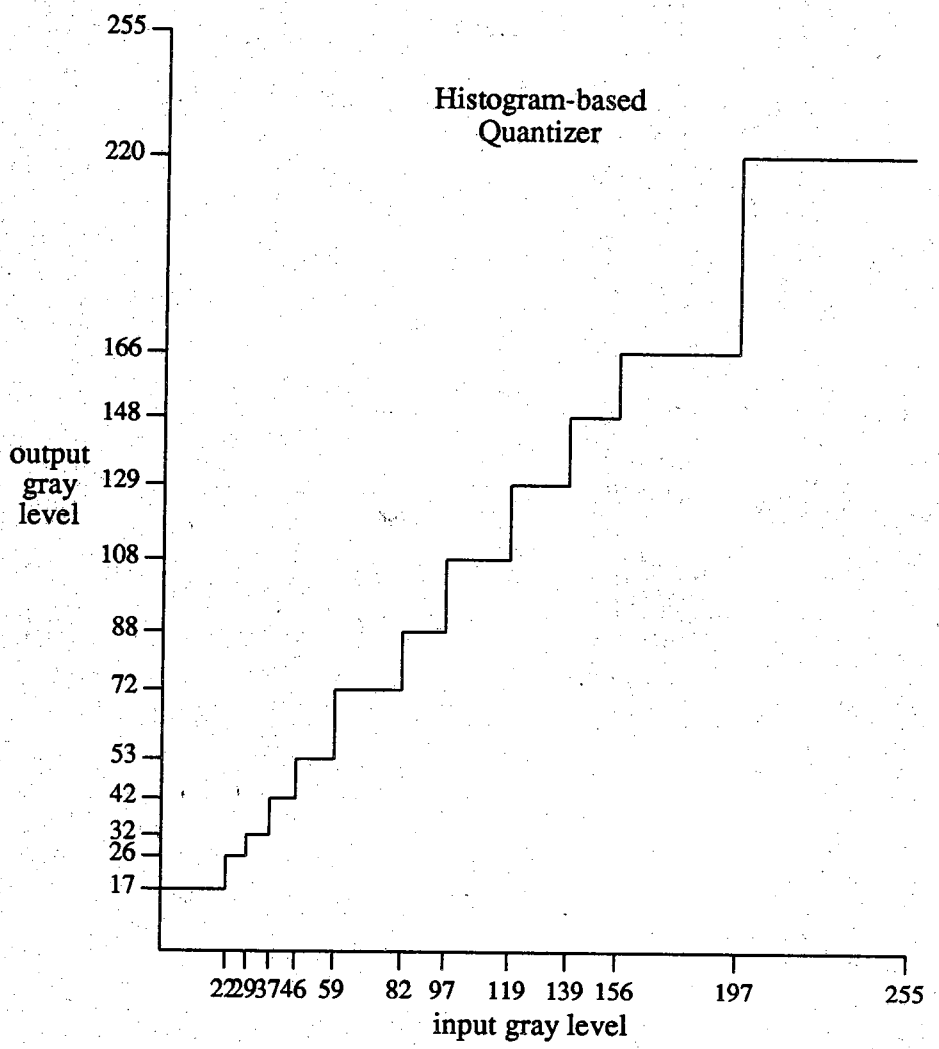


Figure 3.6. The characteristics of a histogram-based quantizer designed for the image of Figure 3.2a.

"house" test image. One set was composed of the segmented "house" image and three quantized versions of the segmented "house," all having 12 gray levels, and the other was composed of the segmented "house" image and three quantized versions of "house," all having 8 gray levels. Both of these test image sets were presented twice to each test subject, with the locations of the three quantized "stimuli" images varied. An example of a test image set is shown in Figure 3.7.

The test image sets were presented to the test subject on a DeAnza CRT monitor (manufactured by Mitsubishi Electric, model C-3910), in a darkened room. This monitor has 512×512 pixel resolution, with 256 possible gray levels. The monitor was calibrated for a linear relationship between gray level numeric value and output luminance. The calibration procedure is described in Appendix E. Each test subject was given approximately three minutes for adaption to the room's illumination (known as "dark adaption"). The test subject sat a distance of approximately six times the image height away from the screen, and was instructed to rank the three quantized segmented images in each test image set in order from the one that most closely resembled the original segmented image, to the one that least resembled the original segmented image. The subject was given 30 seconds to make this determination and an entire trial with one test subject took approximately 15 minutes. In order to compensate for any "learning" by a test subject, or any fatigue in a test subject during the 15 minute testing process, each subject viewed the test image sets in a different order. This variation also compensates for any additional dark adaption by the test subjects after the initial three minutes allowed.

The raw data from the experiment described above is summarized in Appendix B. The median rankings of the three quantized images in each test image set, over both presentations of that test image set to all the test subjects, are shown in Table 3.2. The overall median rankings (over both presentations, for all test subjects, for the various numbers of gray levels) of the three types of quantized images are shown in Table 3.3. Overall, the HVS-based quantizer was ranked third in 49 percent of the trials, the histogram-based quantizer was ranked first in 42 percent of the trials, and the uniform quantizer was ranked second in 47 percent of the trials. The slight superiority of the histogram-based quantizer over the other two quantizers can be explained by the fact that the histogram-based quantizer makes use of *significant* a priori information about the image being quantized, namely the histogram of the image. Neither the uniform nor the HVS-based quantizer makes use any such information. The histogram-based quantizer is strongly image dependent, while the other two quantizers are not. The performance of the HVS-based quantizer could be improved by incorporating information from the image histogram, and by allowing the quantizer characteristic to vary with spatial position in the image.

Figure 3.7. An example of a test image set used in the subjective tests of the quantizers.

Table 3.2. Median rankings of the three quantized images in each test image set, over both presentations of that test image set to all the test subjects.

image	number of quantization levels	Median Rankings		
		HVS quantizer ranking	Histogram quantizer ranking	Uniform quantizer ranking
Eric	12	3	1	2
	18	3	1	2
Girl	20	3	1	2
	24	3	1	2
House	8	1	2	3
	12	2	1	2
Krista	10	2	3	1
	15	2	3	2
	20	2	3	1
Natalie	12	3	1	2
	20	3	1	2

Table 3.3. Overall median rankings (over both presentations, to all test subjects, for all different number of image quantization levels) of the three quantizers for each test image.

image	Median Rankings		
	HVS quantizer ranking	Histogram quantizer ranking	Uniform quantizer ranking
Eric	3	1	2
Girl	3	1	2
House	2	2	2
Krista	2	3	2
Natalie	3	1	2

3.3 Maximum Allowable Extent of Quantization

A important question in relation to segmented image quantization is how many quantization levels are necessary to avoid visible degradation in the quantized segmented image. We have experimented with quantizing segmented images composed of between 200 and 8000 segments. From these experiments we have found that the key factor in determining the extent of quantization possible before noticeable degradation occurs in a segmented image is the percentage of segments that are merged during quantization. Since we assume that each segment in the image being quantized is visually critical, we would like for as few as possible segments to be merged during quantization. (This idea was discussed in the beginning comments of this chapter.) Therefore, an appropriate indication of whether a segmented image has been quantized to too few gray levels is the percent reduction that has occurred in the number of segments in the image.

Figures 3.8b-d through 3.12b-d show quantized versions of the segmented images of Figures 3.8a-3.12a. The quantization was performed using the HVS-based algorithm described in Section 3.2. The number of segments and gray levels in each image is given in Table 3.4. Histograms of the segmented images in Figures 3.8a-3.12a are given in Appendix D. Examining Figures 3.8-3.12, notice that as the number of quantization intervals is decreased, the number of segments in the image also decreases. As the number of image segments begins to decrease significantly, the degradation in the quantized image becomes noticeable. For example, in the case of the image in Figure 3.8, quantization to 25 gray levels (approximately 4.5 bits) reduces the number of segments in the image only from 579 to 512, a change of only 12%. Therefore there is virtually no degradation in the visual quality of the segmented image. However, further quantization down to 10 gray levels (approximately 3.5 bits) reduces the number of segments by approximately 53% and the segmented image's quality suffers noticeably. A similar progression occurs in the images in Figures 3.9-3.12. These images verify that the key factor in determining the extent of quantization possible before noticeable degradation occurs in a segmented image is the percentage of segments that are eliminated due to inadvertent merging of segments during quantization.

3.4 Quantization versus Post-Segmentation Filtering

In Chapter 2 we presented an algorithm for segmentation of a gray level image. The algorithm consisted of two steps: an initial segmentation step, and a post-segmentation filtering step. The purpose of the filtering step was to eliminate visually insignificant segments in the segmented image, so that there were no unnecessary

(a) $d = 4$, $th_{\max} = 8$, energy threshold = 20, # segments = 579

(b) 25 quant. levels, # segments = 512

(c) 18 quant. levels, # segments = 412

(d) 10 quant. levels, # segments = 275

Figure 3.8. A segmented image and three quantized versions. (a) the original segmented image. This image was generated using TH_3 with $m = .123$ and $w = 0.5$, and post-segmentation filtered using E_{2a}. (b-d) Quantized versions of the segmented image in (a). These images were quantized using the HVS-based quantizer described in Section 3.1. The parameters used in segmentation, the energy threshold used in post-segmentation filtering, the number of segments in the images, and the number of quantization levels in the quantized images are given below each image.

(a) $d = 5$, $th_{\max} = 12$, energy threshold = 20, # segments = 986

(b) 26 quant. levels, # segments = 873

(c) 20 quant. levels, # segments = 800

(d) 10 quant. levels, # segments = 476

Figure 3.9. A segmented image and three quantized versions. (a) the original segmented image. This image was generated using TH_3 with $m = .123$ and $w = 0.5$, and post-segmentation filtered using E_{2a}. (b-d) Quantized versions of the segmented image in (a). These images were quantized using the HVS-based quantizer described in Section 3.1. The parameters used in segmentation, the energy threshold used in post-segmentation filtering, the number of segments in the images, and the number of quantization levels in the quantized images are given below each image.

(a) $d = 13$, $th_{\max} = 20$,
energy threshold = 20,
segments = 1084

(b) 28 quant. levels,
segments = 1041

(c) 15 quant. levels,
segments = 972

(d) 8 quant. levels,
segments = 746

Figure 3.10. A segmented image and three quantized versions. (a) the original segmented image. This image was generated using TH_3 with $m = .123$ and $w = 0.5$, and post-segmentation filtered using E_{2a}. (b-d) Quantized versions of the segmented image in (a). These images were quantized using the HVS-based quantizer described in Section 3.1. The parameters used in segmentation, the energy threshold used in post-segmentation filtering, the number of segments in the images, and the number of quantization levels in the quantized images are given below each image.

(a) $d = 3$, $th_{\max} = 40$, energy threshold = 15, # segments = 632

(b) 25 quant. levels, # segments = 587

(c) 15 quant. levels, # segments = 514

(d) 10 quant. levels, # segments = 350

Figure 3.11. A segmented image and three quantized versions. (a) the original segmented image. This image was generated using TH_3 with $m = .123$ and $w = 1.0$, and post-segmentation filtered using E_{2a}. (b-d) Quantized versions of the segmented image in (a). These images were quantized using the HVS-based quantizer described in Section 3.1. The parameters used in segmentation, the energy threshold used in post-segmentation filtering, the number of segments in the images, and the number of quantization levels in the quantized images are given below each image.

(a) $d = 9$, $th_{\max} = 40$, energy threshold = 25, # segments = 791

(b) 25 quant. levels, # segments = 758

(c) 18 quant. levels, # segments = 722

(d) 10 quant. levels, # segments = 594

Figure 3.12. A segmented image and three quantized versions. (a) the original segmented image. This image was generated using TH_3 with $m = .123$ and $w = 0.5$, and post-segmentation filtered using E_{2a}. (b-d) Quantized versions of the segmented image in (a). These images were quantized using the HVS-based quantizer described in Section 3.1. The parameters used in segmentation, the energy threshold used in post-segmentation filtering, the number of segments in the images, and the number of quantization levels in the quantized images are given below each image.

Table 3.4. Summary of the numbers of segments and gray levels in the images in Figures 3.8 - 3.17.

Image	Gray Level Range	Number of Gray Levels	Number of Segments
3.8a	17-241	180	579
b	24-228	25	512
c	24-228	18	412
b	30-221	10	275
3.9a	4-247	201	986
b	13-237	26	873
c	14-232	20	800
d	26-215	10	476
3.10a	24-228	189	1084
b	31-222	28	1041
c	32-214	15	972
d	44-202	8	746
3.11a	27-233	142	632
b	29-237	25	587
c	29-230	15	514
d	31-226	10	350
3.12a	16-238	198	791
b	21-237	25	758
c	21-227	18	722
d	24-216	10	594
3.13a	16-242	213	2848
b	24-228	22	2439
c	24-227	22	483
3.14a	3-254	225	4094
b	20-231	15	3021
c	19-227	15	700
3.15a	21-228	204	3474
b	30-217	20	3398
c	32-218	20	1019
3.16a	26-235	172	3069
b	28-235	20	2527
c	29-233	20	563
3.17a	15-241	214	2719
b	21-227	15	2467
c	21-226	15	697

segments in the image. In this chapter we have discussed quantization of segmented images such as those produced by post-segmentation filtering. Since every segment in the image was assumed to be critical to the quality of the segmented image, the goal has been to quantize the segmented image without reducing the number of segments in the image.

Suppose instead, we apply the quantizer discussed above to the segmented image *before* post-segmentation filtering. We are now quantizing an "over-segmented" image, that is, one that contains some visually insignificant segments. Since we know that quantization of a segmented image results in the elimination of some image segments, it may be possible that quantization of such an "over-segmented" image would accomplish the task of eliminating insignificant image segments, making post-segmentation filtering unnecessary. Figures 3.13-3.17 show examples of images compared to explore this possibility. The images in Figure 3.13a-3.17a have been segmented using the algorithm of Chapter 2. These images have *not* been post-segmentation filtered. Refer to Appendix D for histograms of the segmented images in Figures 3.13a-3.17a. The images in Figure 3.13b-3.17b are quantized versions of the images in Figure 3.13a-17a (generated using the HVS-based quantizer described in this chapter). The images in Figure 3.13c-17c are post-segmentation filtered *and* quantized versions of the images in Figure 3.13a-17a. They have been filtered using the post-segmentation filtering algorithm described in Chapter 2, and quantized using the same quantizers as the images in Figure 3.13b-17b. Refer to Table 3.4 for the number of segments and gray levels in these images. Comparing Figures 3.13b-17b to 3.13c-17c we see that the segmented images that have been filtered and quantized have much fewer segments with the same subjective visual quality as the segmented images that have only been quantized.

This result is not surprising, when the operations of segmented image quantization and post-segmentation filtering are compared. When an image segment is eliminated by being merged with another segment during quantization, this merging happens without any consideration of the segment's size, or the relationships between that segment and neighboring segments. The merging is done using no information about the spatial configuration of the image segments. In contrast, when an image segment is eliminated during post-segmentation filtering, it is only after consideration of the size of the segment and the contrast of the segment with its neighbor segments. Since more complete information is used when eliminating segments during post-segmentation filtering, it is to be expected that post-segmentation filtering produces better decisions are about what segments should be eliminated. The conclusion of this investigation is that quantization does not do a good job of eliminating visually insignificant segments in a segmented image. Quantizing a segmented image cannot take the place of post-segmentation filtering the segmented image.

(a) $d = 4$, $th_{\max} = 8$,
segments = 2848

(b) 22 quant. levels,
segments = 2439

(c) energy threshold = 20,
22 quant. levels,
segments = 483

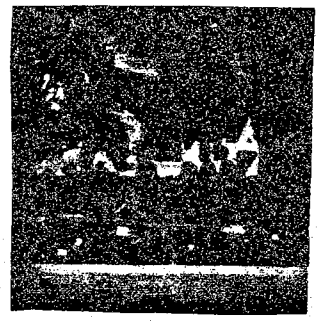
Figure 3.13. Images comparing the effect of quantization with and without preceding post-segmentation filtering. (a) The original segmented image. (generated using TH_3 with $m = .123$ and $w = 0.5$) (b) The segmented image of (a), after HVS-based quantization. (c) The segmented image of (a) after post-segmentation filtering (using E_{2a}) and HVS-based quantization. The parameters used in segmentation, the energy threshold used in post-segmentation filtering, the number of segments in the images, and the number of quantization levels in the quantized images are given below each image.

(a) $d = 5$, $th_{\max} = 12$,
segments = 4094

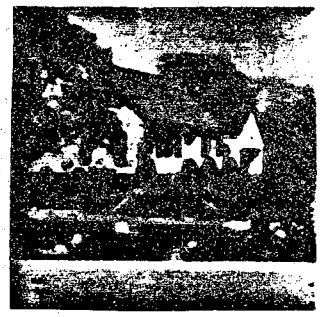
(b) 15 quant. levels,
segments = 3021

(c) energy threshold = 20,
15 quant. levels,
segments = 700

Figure 3.14. Images comparing the effect of quantization with and without preceding post-segmentation filtering. (a) The original segmented image. (generated using TH_3 with $m = .123$ and $w = 0.5$) (b) The segmented image of (a), after HVS-based quantization. c) The segmented image of (a) after post-segmentation filtering (using E_{2a}) and HVS-based quantization. The parameters used in segmentation, the energy threshold used in post-segmentation filtering, the number of segments in the images, and the number of quantization levels in the quantized images are given below each image.



(a) $d = 13$, $th_{\max} = 20$,
segments = 3474



(b) 20 quant. levels,
segments = 3398

(c) energy threshold = 20,
20 quant. levels,
segments = 1019

Figure 3.15. Images comparing the effect of quantization with and without preceding post-segmentation filtering. (a) The original segmented image. (generated using TH_3 with $m = .123$ and $w = 0.5$) (b) The segmented image of (a), after HVS-based quantization. (c) The segmented image of (a) after post-segmentation filtering (using E_{2a}) and HVS-based quantization. The parameters used in segmentation, the energy threshold used in post-segmentation filtering, the number of segments in the images, and the number of quantization levels in the quantized images are given below each image.

(a) $d = 3$,
$th_{\max} = 40$,
segments = 3069

(b) 20 quant. levels,
segments = 2527

(c) energy threshold = 15,
20 quant. levels,
segments = 563

Figure 3.16. Images comparing the effect of quantization with and without preceding post-segmentation filtering. (a) The original segmented image. (generated using TH_3 with $m = .123$ and $w = 1.0$) (b) The segmented image of (a), after HVS-based quantization. (c) The segmented image of (a) after post-segmentation filtering (using E_{2a}) and HVS-based quantization. The parameters used in segmentation, the energy threshold used in post-segmentation filtering, the number of segments in the images, and the number of quantization levels in the quantized images are given below each image.

(a) $d = 9$, $th_{\max} = 40$,
segments = 2719

(b) 15 quant. levels,
segments = 2467

(c) energy threshold = 25,
15 quant. levels,
segments = 697

Figure 3.17. Images comparing the effect of quantization with and without preceding post-segmentation filtering. (a) The original segmented image. (generated using TH_3 with $m = .123$ and $w = 0.5$) (b) The segmented image of (a), after HVS-based quantization. (c) The segmented image of (a) after post-segmentation filtering (using E_{2a}) and HVS-based quantization. The parameters used in segmentation, the energy threshold used in post-segmentation filtering, the number of segments in the images, and the number of quantization levels in the quantized images are given below each image.

In this chapter we proposed a HVS-based quantizer for segmented images and described the procedure for its design. We compared the performance of this quantizer through a series of subjective experiments to a uniform quantizer and a histogram-based quantizer. The histogram-based quantizer was ranked slightly higher than the other two quantizers. This was as expected, since the histogram-based quantizer takes advantage of a priori information about the image not used by the other two quantizers. We showed that the extent of quantization possible for a segmented image is limited by the percentage of segments the quantization operation eliminates from the segmented image. Finally, we investigated the interaction between the operations of quantizing a segmented image, and post-segmentation filtering a segmented image. These experiments demonstrated the importance of each operation. Both post-segmentation filtering and quantization successfully exploit HVS properties in generating visually pleasing segmented images with the minimum number of segments and gray levels.

CHAPTER 4
MATHEMATICAL MORPHOLOGY

Mathematical morphology is a set theoretic method for the quantitative analysis of the geometrical form of sets in a Euclidean space. The foundations of mathematical morphology were developed in 1964-1968 by G. Matheron and J. Serra at the Paris School of Mines at Fountainbleu, France [8, 114]. The word "morphology" comes from the Greek word meaning "the study of forms" [9]. Morphology has its mathematical roots in the areas of integral geometry and geometrical probabilities [8]. Specifically, morphological transformations are based on the set operations of *Minkowski set addition* and *Minkowski set subtraction*, which emerged from Minkowski's work in the study of ill-behaved sets [115-117]. By representing images as sets, Minkowski set algebra can be applied to images.

In order to use morphology, an image is treated as a set in an N-dimensional Euclidean space. This image set interacts with a *structuring element*, which is also a set in the same Euclidean space. The goal of this interaction is to transform the image set into a new form which is more expressive of some selected geometric property of the image. This new form can be used as a symbolic representation of the original image set. Also, this new form allows for quantification of the selected geometric property, and thus the quantitative analysis of the geometric form of the image can be accomplished. This two step approach of morphology is illustrated in Figure 4.1. A significant characteristic of this approach is that the image is treated as a whole entity (the image set), rather than as a collection of local details. This idea of transforming an image into a more meaningful form is basic to the philosophy of morphological image processing, and it stems from fundamental facts about image perception.

In the mind's image perception process, transformation of an image to make it more meaningful in some particular way is used extensively. According to Serra, *"For any type of perception, the mind remodels the stimulus, in order to assimilate it to its own patterns...To perceive an image is to transform it"* [8]. The work of the gestaltists in the field of psychology, especially W. Kohler [118], has found this structuring activity of the mind in even the most simple perceptive phenomena [8]. The transformational nature of the mind's visual perception process provides justification for the transformational approach of morphological image processing.

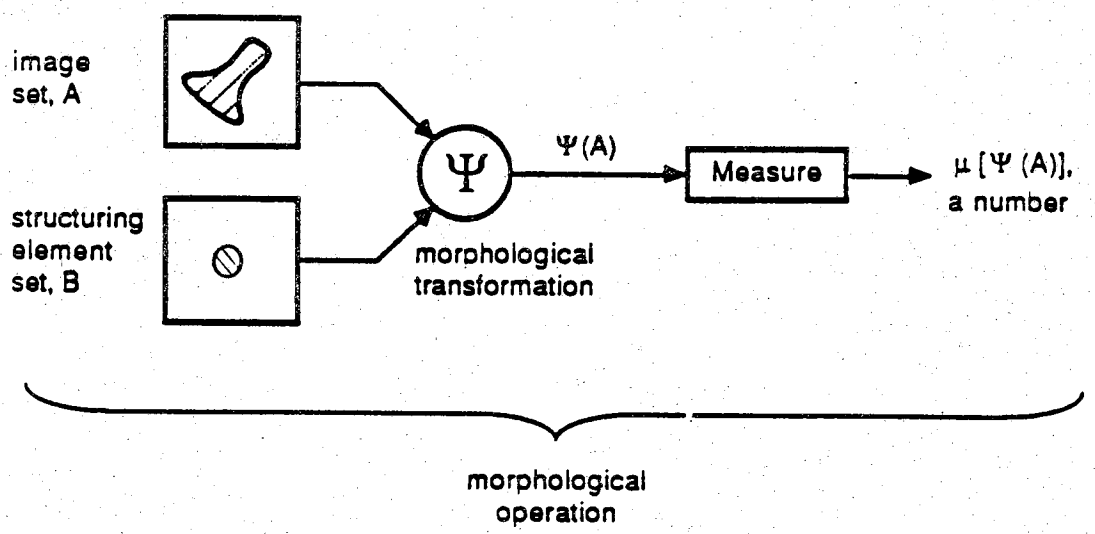


Figure 4.1. The two step morphological operation.

Another fundamental fact of image perception is that it is not purely objective; the important aspects of the geometrical structure of an image vary from observer to observer. When viewing an image *"we see only what we want to look at"* [8]. In order to extract the information from an image that is of interest to a specific observer, the morphological transformation performed on the image must somehow be adapted for the observer. Morphology incorporates this requirement through the use of the structuring element. Serra summarizes this idea: *"...(geometrical structure) does not exist in the phenomenon itself, nor in the observer, but somewhere in between the two. Mathematical morphology quantifies this intuition by introducing the concept of structuring elements"* [8]. The structuring element set is generally smaller and has a simpler shape than the image set. It acts as a kind of probe of the image; and the particular property expressed through a morphological transformation is determined by the structuring element used. For example, through the use of different structuring elements information can be extracted about the size, shape, orientation, connectivity, or smoothness of the image object.

The basic building blocks of any morphological transformation are set union and set intersection. Certain restrictions must be placed on the allowed combinations of these set operations in order for the results of a morphological transformation, $\Psi(\cdot)$, to be meaningful for image analysis. Let A represent the original image set. Then the restrictions can be expressed by the following four quantification constraints [8, 9]:

(1.) $\Psi(A)$ must be translation-invariant, i.e. $\Psi(A_z) = [\Psi(A)]_z$.

(2.) $\Psi(A)$ must be scale-invariant, i.e. for a scale parameter $\lambda>0$, $\Psi(\lambda X) = \lambda\Psi(X)$.

(3.) $\Psi(A)$ must be a function of only a bounded local area, i.e. for any bounded mask X^* within which we want to know $\Psi(A)$, there exists a bounded mask X such that $\Psi(A)\cap X^* = [\Psi(A\cap X)]\cap X^*$.

(4.) $\Psi(A)$ must be upper-semicontinuous, i.e. for an increasing set transformation $\Psi(\cdot)$, and a decreasing sequence of closed sets A_n approaching the limit A, the sequence $\Psi(A_n)$ must approach $\Psi(A)$.

Every set transformation which satisfies these four principles is known as a quantitative morphological transformation, or just a morphological transformation when there is no chance of confusion.

For a detailed theoretical discussion of mathematical morphology, the book by Serra is an excellent reference [8]. A more compact presentation of the basics of morphology can be found in [9, 119]. A slightly different approach to morphology is presented in [120]. All of the above references discuss both binary and gray scale morphology. Concise presentations on binary morphology are contained in [10] and [121], and Sternberg has written a paper dealing with only gray scale morphology [122].

4.1 Basic Morphological Set Transformations

Morphology deals with sets and set operations in N-dimensional Euclidean space. This Euclidean space can either be continuous, $\mathbf{R}^n$, or discrete, $\mathbf{Z}^n$. Erosion, dilation, opening, and closing are the basic quantitative morphological transformations. All of these transformations are based on *Minkowski set addition* and *Minkowski set subtraction*. The Minkowski set addition (MSA) of two sets, A and B, denoted $A \oplus B$, is defined as

$$A \oplus B = \{a + b: a \in A, b \in B\} = \bigcup_{b \in B} A_b. \tag{4.1}$$

Minkowski set subtraction (MSS), denoted $A \ominus B$, is the dual transformation with respect to complementation of MSA, and is defined as

$$A \ominus B = (A^c \oplus B)^c = \bigcap_{b \in B} A_b. \tag{4.2}$$

Examples of these transformations in $\mathbf{Z}^2$ are shown in Figure 4.2 [10].

Using MSA and MSS, the morphological transformations of *erosion* and *dilation* can be defined. The erosion of a set A by a set B, denoted $A \text{ e } B$, is

$$A \text{ e } B = A \ominus B^s, \tag{4.3}$$

where $B^s = \{-b: b \in B\}$ is the symmetric set of B with respect to the origin. Expressing Equation 4.3 another way,

$$A \text{ e } B = \{z: B_z \in A\} = \bigcap_{b \in B^s} A_b. \tag{4.4}$$

The dilation of A by B, denoted $A \text{ d } B$, is

$$A \text{ d } B = A \oplus B^s, \tag{4.5}$$

or equivalently

$$A \text{ d } B = \{z: B_z \cap A \neq \varnothing\} = \bigcup_{b \in B^s} A_b. \tag{4.6}$$

Notice that if $B = B^s$ then MSA is equivalent to dilation and MSS is equivalent to erosion. Figures 4.2 and 4.3 show examples of erosion and dilation in $\mathbf{Z}^2$ and $\mathbf{R}^2$ [10]. It can be seen from these examples that erosion shrinks a set, while dilation expands a set. Erosion and dilation are dual transformations with respect to complementation, and these two transformations are generally non-invertible. Also, erosion and dilation are translation invariant, and both are increasing transformations with respect to the first operand. If the second operand, B, contains the origin, then erosion is anti-extensive and dilation is extensive. Other interesting and useful properties of these two

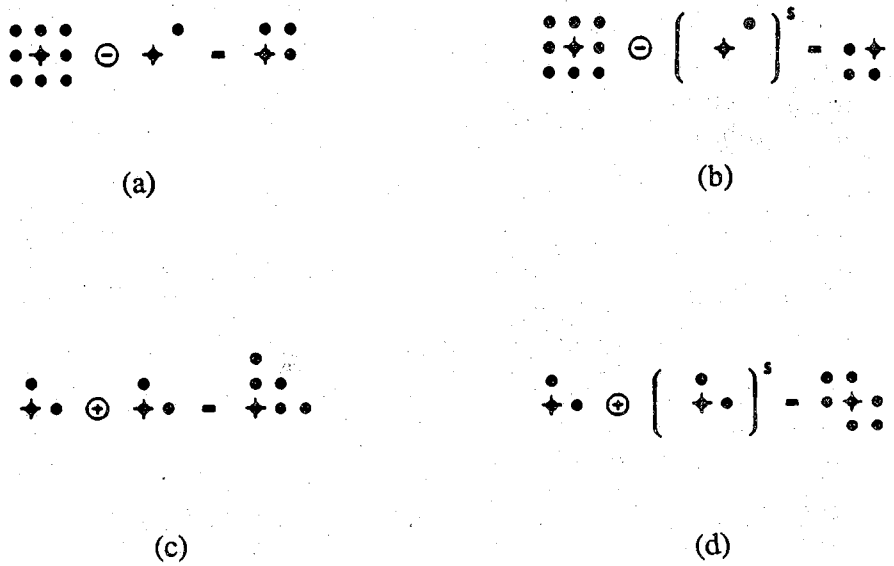


Figure 4.2. Morphological transformations of discrete sets in $\mathbf{Z}^2$. (a) Minkowski set subtraction. (b) erosion. (c) Minkowski set addition. (d) dilation. (● = *objectpoints*, + = *origin*) (from [9]).

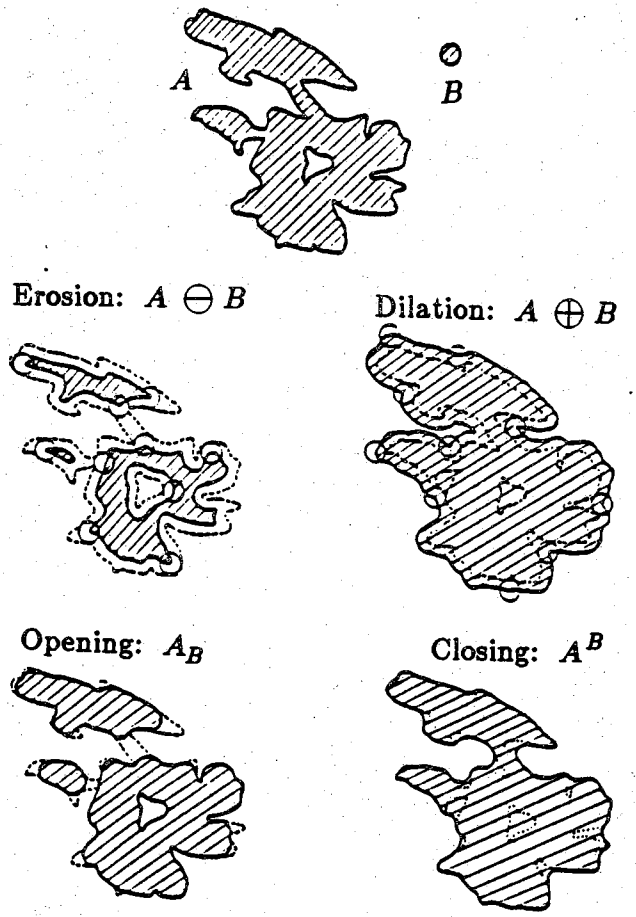


Figure 4.3. (a) Erosion, (b) dilation, (c) opening, and (d) closing of A by B in $\mathbf{R}^2$. The shaded areas correspond to the interior of the sets, the dark solid curve to the boundary of the transformed set, and the dashed curve to the boundary of the original set, A (from [9]).

morphological operators are given in [10, 119].

Based on the definitions of erosion and dilation, the quantitative morphological transformations of *opening* and *closing* can be specified. The opening of A by B, denoted A_B, is defined as

$$A_B = (A \ominus B^s) \oplus B = (A \text{ e } B) \text{ d } B^s. \tag{4.7}$$

The dual with respect to complementation of opening is closing. The closing of A by B, denoted A^B, is defined as

$$A^B = (A \oplus B^s) \ominus B = (A \text{ d } B) \text{ e } B^s. \tag{4.8}$$

Equivalent alternative definitions of opening and closing are:

$$A_B = \{a \in A: \text{for some } y \text{ , } a \in B_y \subseteq A\} \tag{4.9}$$

and

$$A^B = \{z \in \mathbf{E}: \text{for all } y \text{ such that } z \in B_y^s,\ B_y^s \cap A \neq \varnothing\}. \tag{4.10}$$

Examples of opening and closing in $\mathbf{R}^2$ are given in Figure 4.3. The examples illustrate that opening or closing A by B suppresses all details in A that are smaller than B. Opening A by B eliminates narrow peninsulas and necks in A; closing A by B fills in thin rivers and bays, and eliminates small holes in A. Opening and closing are increasing, translation invariant transformations. Opening and closing are also both idempotent transformations, in other words $(A_B)_B = A_B$, and likewise $(A^B)^B = A^B$. Other useful properties of these transformations are discussed in [8, 10].

It is important to note here that the definitions for the basic morphological transformations often vary slightly from author to author. The definitions given above agree with those of Maragos and Serra [9, 10, 119, 120]. The main difference is in where B^s is used. Despite the differences, the definitions of Maragos, Serra, and Haralick for opening and closing all agree.

All of the above discussion applies to both binary and gray scale morphology. The set transformations defined can be applied to sets in a Euclidean space of any finite dimension. The main distinction between binary and gray scale of morphology is the dimension of the Euclidean spaces in which the sets reside. Binary morphology deals with sets in two-dimensional Euclidean space, while gray scale morphology deals with sets in three-dimensional Euclidean space. In Sections 4.2 and 4.3 we will present the basics of binary morphology and gray scale morphology, respectively.

4.2 Binary Morphology

Binary morphology applies the basic morphological set transformations outlined in Section 4.1 to sets in two-dimensional Euclidean space. In order to use binary morphology to analyze a two-dimensional binary image, the image must be represented as a set in this space. The two-dimensional image set is formed by assigning all locations in the image with one binary value (e.g. "0") to be in the image set, and all locations in the image with the other binary value (in this case, "1") to be in the complement of the image set. To morphologically process a binary image, the basic quantitative morphological transformations are applied to the image set and the structuring element set in Euclidean 2-space. The first operand in these transformations is the image set and the second operand is the structuring element set. The structuring element is chosen by the user to fit the desired purpose. It can be virtually any set in the same Euclidean space as the image set. Since the images we deal with are usually sampled and therefore discrete, normally $\mathbf{E}$ will be $\mathbf{Z}^2$ (as opposed to $\mathbf{R}^2$).

4.3 Gray Scale Morphology

Gray scale morphology applies the basic morphological set transformations outlined in Section 4.1 to sets in three-dimensional Euclidean space. In order to use gray scale morphology to analyze a two-dimensional gray scale image, the image must be represented as a set in this space. As was the case with binary images, a two-dimensional gray scale image can be represented as a set in this space, though the representation process for gray scale images is not as simple as for binary images. Since the images we deal with are usually sampled and therefore discrete, normally $\mathbf{E}$ will be $\mathbf{Z}^3$; however the discussion below is applicable to either $\mathbf{Z}^3$ or $\mathbf{R}^3$.

In order to specify the basic quantitative gray scale morphological transformations, first some mathematical preliminaries must be understood. A key mathematical principle in morphology is that of the umbra. The set F in $\mathbf{Z}^3$ is an umbra if it satisfies the following property:

$$(x,y,z) \in F \Rightarrow (x,y,w) \in F \text{ for all } w \leq z. \tag{4.11}$$

Umbra can also refer to a set operation. The umbra of a set F in $\mathbf{Z}^3$ is defined to be

$$U[F] = \{(x,y,z): (x,y,a) \in F \text{ and } z \leq a\}. \tag{4.12}$$

The set operations union and intersection are defined for umbras as follows:

$$U[F] \cup U[G] = U\Big[\{(x,y,c): (x,y,a) \in F \text{ or } (x,y,b) \in G,\ c=\max(a,b)\}\Big] \tag{4.13}$$

$$U[F] \cap U[G] = U\left[\{(x,y,c): (x,y,a) \in F \text{ and } (x,y,b) \in G,\ c=\min(a,b)\}\right] \quad (4.14)$$

Obviously the union or intersection of two umbras is an umbra. Also umbras remain umbras under translation.

Another useful mathematical concept is that of the top of a set, denoted $T[F]$, which is defined

$$T[F] = \left\{(x,y,z): z = \max_{(x,y,a)\in F} a\right\}. \quad (4.15)$$

Further explanations of the concepts of top and umbra are given in [8, 120, 122].

With these preliminaries taken care of, we are ready to detail the process of representing a gray scale image as a set in Euclidean 3-space. A gray scale image is a function $f(x,y)$ on the points in Euclidean 2-space. This function on Euclidean 2-space can also be thought of as the set, F, in Euclidean 3-space, where the coordinates in the set are $[x, y, f(x,y)]$. Since $f(x,y)$ is a function, F would take the shape of a thin contour in Euclidean 3-space. Using our previous definition of umbra, the umbra of F is

$$U[F] = \{(x, y, z): z \le f(x, y)\} = U[f]. \quad (4.16)$$

Note that $U[f]$ is also a set in Euclidean 3-space. Gray scale morphological transformations on an image $f(x,y)$ are expressed as set operations on the set $U[f]$ in Euclidean 3-space . Since umbras remain umbras under union, intersection, and translation, morphological transformations on umbras always produce umbras.

As was the case for binary morphology, the basic quantitative gray scale morphological transformations of erosion, dilation, opening and closing are obtained by applying the set transformations specified in Section 4.1, this time to sets in Euclidean 3-space. This extension to three-dimensional Euclidean space is justified by a principle known as the *umbra homomorphism theorem*, which states that the operation of taking an umbra is a homomorphism from gray scale functions to binary set transformations. By using the umbra operator, gray scale morphological transformations can be expressed in terms of basic morphological set transformations. This idea can be expressed in equation form as follows:

$$U[f(i,j) \text{ d } g(i,j)] = U[f] \text{ d } U[g], \quad (4.17)$$

and

$$U[f(i,j) \text{ e } g(i,j)] = U[f] \text{ e } U[g]. \quad (4.18)$$

The left hand sides of these equations specify transformations on the gray scale representation of the image, while the right hand sides specify transformations on the set representation of the gray scale image. For proof of this theorem, see [120]. If

umbra and top are not used, gray scale morphological transformations can be expressed in terms of the function, $f(x,y)$, which defines the image. Both these modes of definition for gray scale morphological transformations will be given below.

The gray scale morphological erosion of the function f by the function g is a set operation in Euclidean 3-space defined as follows:

$$f(x,y) \text{ e } g(x,y) = T[\ U[F] \text{ e } U[G]\ ]. \tag{4.19}$$

Note that the e symbol on the left hand side of this equation represents gray scale erosion, while the e symbol on the right hand side represents basic morphological set erosion. This equation demonstrates how the umbra and top functions enable us to express gray scale morphological transformations in terms of the basic morphological set transformations that have already been defined. Using the functional representation of the image, an alternative definition of the erosion of $f(x,y)$ by $g(x,y)$, denoted $e(x,y)$, is

$$e(x,y) = \min_{(a,b)\in \mathbf{E}} [f(x-a,y-b) - g(-x,-y)]. \tag{4.20}$$

The two analogous definitions for dilation are

$$f(x,y) \text{ d } g(x,y) = T[\ U[F] \text{ d } U[G]\ ] \tag{4.21}$$

and

$$d(x,y) = \max_{(a,b)\in \mathbf{E}} [f(x-a,y-b) + g(x,y)]. \tag{4.22}$$

Following the pattern of the basic morphological set transformations, gray scale opening and closing are defined in terms of gray scale erosion and dilation:

$$f(x,y) \text{ opened by } g(x,y) = F_G = (F \ominus G) \oplus G, \tag{4.23}$$

$$f(x,y) \text{ closed by } g(x,y) = F^G = (F \oplus G) \ominus G. \tag{4.24}$$

With this background in the basics of morphology, we are prepared to apply morphology to skeletonize a binary image. The application of morphological skeletonization in image compression is described in detail in the following chapter.

CHAPTER 5
A NEW SEGMENTATION-BASED IMAGE CODING ALGORITHM

In segmentation-based image compression [2-4, 6, 80, 82, 87-89]. the image to be compressed is segmented, and information is extracted describing the shapes and interiors of the segments in the segmented image. This information is used to form the coded version of the image. For segmentation-based compression methods, the segmented version of the original image is reconstructed at the decoder. Segmentation-based image compression is obviously not a distortionless coding method.

Since in segmentation-based compression, information must be coded describing each image segment, the number of segments in an image determines, for the most part, the bit rate of the coded image. For this reason, segmentation-based compression methods are best suited for use in applications where the images can be estimated with few, large segments (for example simple "head and shoulders" images like those typically found in video-telephone or video-teleconferencing applications). Since our segmentation technique uses flat segments, this implies that our compression method is not well suited for highly textured images, since textured areas in an image would produce numerous segments in the segmented image.

In this chapter we present an image compression method which is based on this approach, and employs the image segmentation and quantization techniques discussed in Chapters 2 and 3. The compression method we propose differs in a significant way from other segmentation-based image compression methods. In the past, the shapes of the image segments were represented by the segment boundaries [6, 80, 87, 89]. With our compression technique, an alternative representation of the segments' shapes is used. Instead of segment boundaries, *morphological skeletons* are used to represent the segments.

The proposed compression technique is composed of four steps, shown in Figure 5.1. The first step is preprocessing. This is discussed in detail in Section 5.1. After preprocessing, the gray level image is segmented and quantized using the methods described in Chapters 2 and 3. The parameters necessary as input for this stage of the compression algorithm are d and $thmax$, which determine the number of segments in the segmented image, the energy threshold, which determines the number of segments removed during post-segmentation filtering, M, the number of gray levels in the range

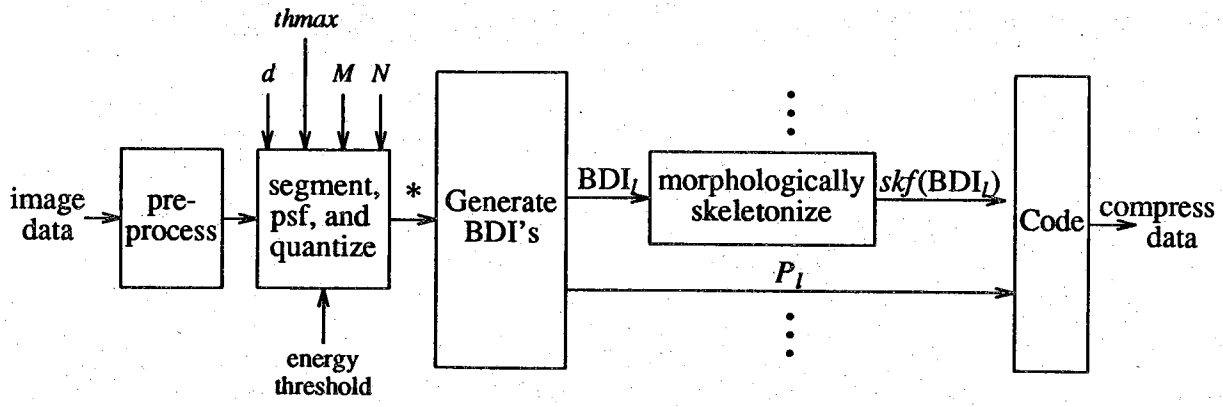


Figure 5.1. A block diagram of the encoder for a new segmentation-based compression technique for gray level images. The image at produced at "*" is the image that will be decoded. The first two blocks in this diagram are shown in more detail in Figure 2.18 ("psf" refers to post-segmentation filtering).

of the segmented image, and N, the number of gray levels desired in the quantized segmented image. These methods are briefly reviewed, and the advantages of quantization are discussed in Section 5.2. The third step in our compression algorithm is the generation of morphological skeletons to represent the image segments. A skeleton is a thin-lined caricature of the segment that summarizes its shape and conveys information about its size, orientation, and connectivity. In [10] a simple procedure is described using binary morphology to find the skeleton of a binary image. Section 5.3 describes this skeletonization technique and its application to skeletonize the image segments. The final step in our compression algorithm is the actual coding of the segments' skeletons and interiors. Three alternatives for this part of the algorithm are presented in Section 5.4. One of these alternatives involves an idea we refer to as the "minimal set of segments" to reduce the bit rate required for the coded image. The process for decoding a compressed image is described in Section 5.5.

Experiments have been performed to evaluate various aspects of the compression algorithm, and to compare different coding options. These experiments and their results are detailed in Section 5.6. The advantages and disadvantages of our morphological skeleton approach to segmentation-based image compression are also discussed. We also compare segmentation-based compression using skeletons to segmentation-based compression where segment boundaries are coded.

5.1. Preprocessing

In any segmentation-based compression algorithm, a description must be encoded for each segment in the segmented image. Thus, the number of image segments determines, for the most part, the bit rate of the coded image. Because of this, a minimum number of image segments is critical. The main purpose of preprocessing is to alter the image in such a way that fewer segments are produced by the segmenter, without degrading the visual quality of the segmented image.

One possible preprocessing operation proposed in [6] is clamping. Clamping reduces the dynamic range of the image by setting all pixels with gray level above a threshold to that threshold, and setting all pixels below a second threshold to the second threshold. This can be expressed:

$$p = \begin{cases} th_1, & p < th_1, \\ p, & th_1 \le p \le th_2 \\ th_2, & p > th_2, \end{cases} \tag{5.1}$$

where p is the gray level of a pixel in the image, and th_1 and th_2 are the two clamping

thresholds. Clamping is motivated by the contrast sensitivity of the eye, which is known to decrease as the intensity of the visual stimulus moves away from the middle range of intensity values [45]. The reasoning is that, since the eye has reduced sensitivity to differences in very high gray levels and differences in very low gray levels, variety in gray levels at these extremes of the gray level range is unnecessary.

A second possible preprocessing operation proposed in [6] is median filtering. Since the MTF of the eye indicates that the eye has reduced sensitivity to very high and very low spatial frequencies, isolated pixels may not be perceptually significant to the human viewer, and could possibly be removed from the image without degrading the subjective image quality. A two-dimensional, 3×3, separable median filter [123] is used to remove these supposedly insignificant fluctuations in the image data. A median filter moves a window along the data to be filtered and sets the output to be the median value of the data points in the window. A two-dimensional separable 3×3 median filter applies this procedure serially, to first the rows, then the columns of the image, using a 3 pixel wide window. This filter has the effect of eliminating all "spikes" in the image data which are one pixel in width in either the vertical or the horizontal direction. For example, any one pixel wide line in the image would be removed by this filter.

The result of preprocessing is a somewhat smoother image with reduced dynamic range. This should mean fewer segments in the segmented image, and thus a lower bit rate code. The effectiveness of these preprocessing operations has been evaluated experimentally, and the results of these experiments are discussed in Section 5.6.

5.2. Image Segmentation and Quantization

After the appropriate preprocessing has been performed, the next step of the compression algorithm is segmentation and quantization of the image. Image segmentation is accomplished in two steps, both of which were detailed in Chapter 2. First, a variation of centroid-linkage region growing [5] is used to form an initial segmentation of the image. There are two main reasons why centroid-linkage region growing was chosen to perform the segmentation in our compression algorithm. First, HVS properties can be readily incorporated into centroid-linkage region growing via the segmentation thresholds. This was described in Chapter 2. The second reason is that region growing segmentation techniques are guaranteed to produce disjoint segments with closed boundaries. This is important because segmentation-based compression requires a description of the shape and interior of each image segment. Such a description would be impossible if the segments overlapped or did not have closed boundaries.

After initial segmentation, post-segmentation filtering is performed on the segmented image to eliminate visually insignificant segments. Both of these operations

incorporate HVS properties. The result of the segmentation is a gray level image composed of a number of regions, each with uniform gray level.

After the image has been segmented, it is quantized using the HVS-based quantizer described in Chapter 4. By quantizing the segmented image, the number of different gray levels used to describe the segment interiors is reduced. Therefore fewer bits are required to encode the segments' gray levels. Thus, quantization of a segmented image leads to a reduction in the number of bits required to code the description of the regions in the segmented image.

The data rate can be reduced even further by utilizing quantization in another way. With the image segments assigned one of a limited number of gray levels, it is feasible for all segments with the same quantized gray level to be grouped together for transmission. In this way, it is necessary to transmit the gray level only once for each large group of segments, rather than for each individual segment. We will use this approach in coding the segmented image. The actual coding procedure is discussed in more detail in Section 5.4.

The final result of segmentation and quantization is a gray level image composed of many segments, each segment "painted in" with one of a limited number of gray levels. This is the image that will be generated at the output of the decoder.

5.3. Segment Skeletonization

Once the image has been segmented, the next step is to generate representations for the shapes and interiors of the segments in the image. In our compression algorithm, the shapes of the image segments are represented by morphological skeletons. A skeleton is a thin-lined caricature of the segment that summarizes its shape and conveys information about its size, orientation, and connectivity. Segment skeletons will be generated using the morphological skeletonization algorithm of [10]. A summary of this skeletonization process, and a description of its application to the image segments is given below.

In mathematical morphology, the form and structure of sets in Euclidean N-space are studied [8]. The basics of mathematical morphology were presented in Chapter 4. Through the application of morphological operations such as dilation, erosion, opening, and closing to a binary image, a gray level skeleton of a binary image can be generated [8, 10, 120].

Given a binary image set, A, the skeleton of A is formally defined to be the set of centers of the maximal disks inscribable inside A. A maximal disk is one that is not contained in any other disk totally included in A. The idea of the skeleton was first

introduced by Blum [124], who referred to it as a *medial axis*, or *symmetric axis* [125]. The definition of a skeleton given above is the same as Blum's definition for a medial axis. Figure 5.2 shows examples of maximal disks and skeletons of binary images [10]. It can be seen from these examples that if we draw the maximal disk at each point on the skeleton of A, then the union of these maximal disks will be exactly equal to A. The set of the centers of all maximal disks in A with radius r ($r \geq 0$), is called the r'th *skeleton subset* of A, denoted $S_r(A)$. The skeleton subsets are mutually exclusive. An example of an r'th skeleton subset is shown in Figure 5.2. These skeleton subsets can be obtained using the morphological set operations of opening and erosion. It should be noted that the morphological transformation that maps a set into its skeleton is not upper semi-continuous as defined in Chapter 4, and therefore is not a quantitative morphological operation. An example demonstrating this is shown in Figure 5.3. The skeletonizing transformation is translation and scale invariant, and lower semi-continuous.

We are interested in applying morphology to skeletonize image segments. These segments can be viewed as sets in Z^2; therefore we now limit our discussion to morphological skeletonization in Z^2. Using the notation introduced in Chapter 4, the skeleton of a set, $A \in Z^2$, can be obtained as follows:

$$S_n(A) = (A \ominus nB^s) - (A \ominus nB^s)_B, \quad n = 0, 1, 2, \cdots, N \tag{5.2}$$

$$SK(A) = \bigcup_{n=0}^{N} S_n(A), \tag{5.3}$$

where $S_n(A)$ denotes the n'th skeleton subset of A, $SK(A)$ denotes the binary morphological skeleton of A, B is the structuring element used to perform the skeletonization, and

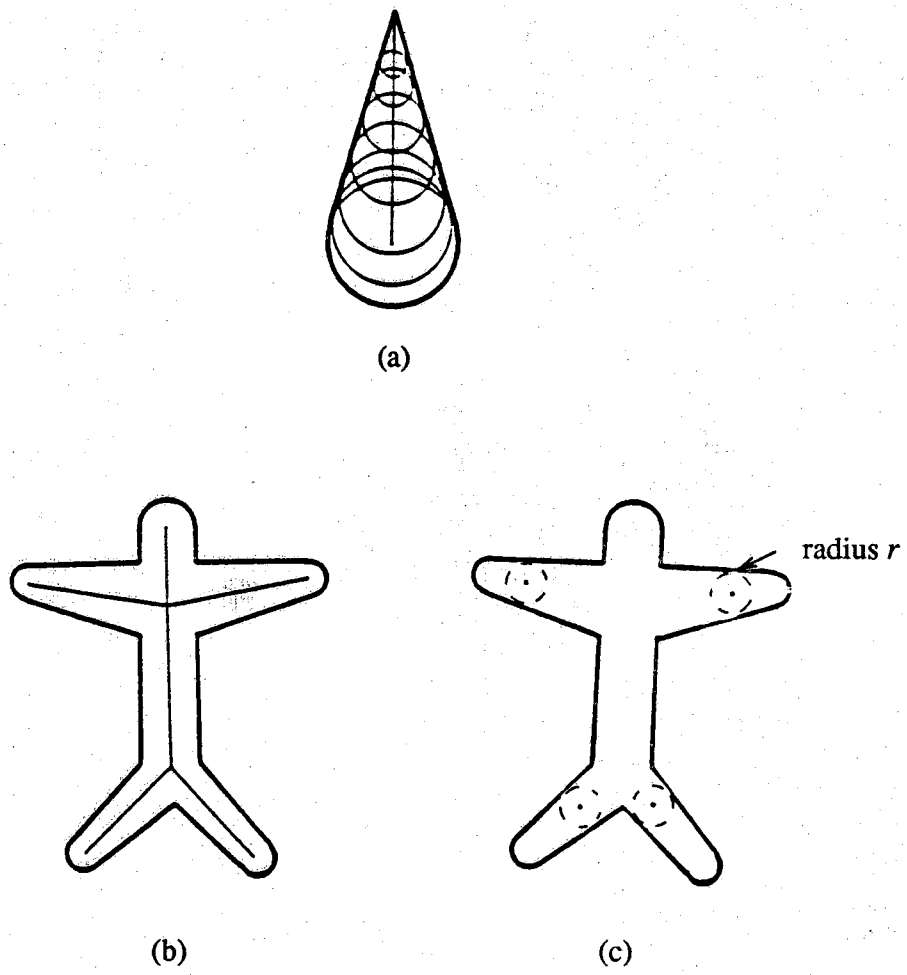
$$nB = B \oplus B \oplus \cdots \oplus B \quad (n \text{ times}).$$

N is the maximum n after which a further erosion of A by B results in the empty set. B is the structuring element, and can be chosen as desired. For skeletonization of the image segments, B is chosen to be a 3×3 pixel square, with the center of B in the middle of the square. Notice that, since $S_0(A) = (A \ominus 0B^s) - (A \ominus 0B^s)_B = A - A_B$, the 0'th skeleton subset simply contains all features of A that are smaller than B.

An alternate, more compact representation of the information in the skeleton is the gray level *skeleton function*, defined as the following two-dimensional discrete image array:

$$[skf(A)](i,j) = \begin{cases} n+1, & (i,j) \in S_n(A) \\ 0, & (i,j) \notin SK(A), \end{cases} \tag{5.4}$$

where i and j index the rows and columns of the image. The gray level of a point on the skeleton function indicates the skeleton subset to which that skeleton point belongs, i.e.



(a)

(b)

(c)

Figure 5.2. Examples of skeletons and maximal discs of sets in $\mathbf{R}^2$. (a) the set of maximal disks and their centers for a cone-shaped set, A. Since the skeletons shown are binary, they represent $SK(A)$ (from [9]) (b) $SK(A)$ for a stickman-shaped set, A (from [9]). (c) The r'th skeleton subset of the $SK(A)$ shown in (b) These points on $SK(A)$ would have gray level $r+1$ on $[skf(A)](i,j)$.

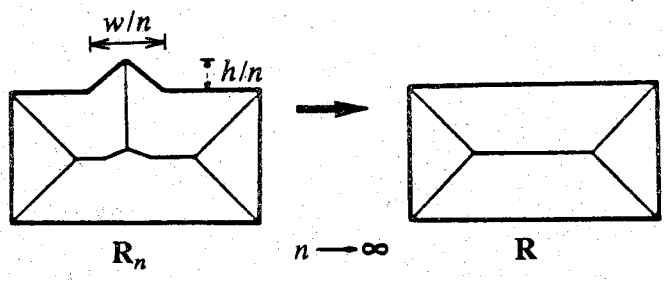


Figure 5.3. A series of sets in $\mathbf{R}^2$ and their morphological skeletons illustrating that morphological skeletonization is not a continuous transformation (from [9]).

radius of the maximal disc centered at that point. In Section 5.6 we will give examples of binary images and their corresponding gray level skeleton functions. An important property of the morphological skeleton as defined above, is that it is uniquely invertible [10]. If the structuring element is known, a binary image can be perfectly and uniquely reconstructed from its morphological skeleton. The inversion process is described in Section 5.5.

Once the skeleton described above has been generated, further processing of this skeleton yields the *globally minimal* skeleton [10]. The aim of the processing is to remove redundant skeleton points from the skeleton, and thus produce the globally minimal skeleton. The motivation for this is that, since the globally minimal skeleton has fewer points it will require fewer bits to encode. The following algorithm is used to determine what points should be removed from a morphological skeleton in order to generate the globally minimal skeleton [10].

Assume $A \in Z^2$ is the original binary image, and $S_n(A)$, $n = 0, 1, \ldots N$ are the skeleton subsets of A with respect to the structuring element B. The following steps are repeated for $n = 0, 1, \ldots N$. Define $k_n(i, j)$ to be the binary characteristic function of the set $nB \in Z^2$. In other words,

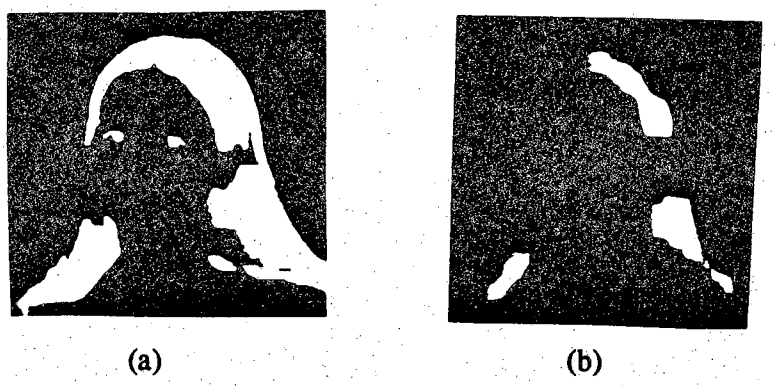
$$k_n(i, j) = \begin{cases} 1, & (i, j) \in nB \\ 0, & (i, j) \notin nB. \end{cases} \tag{5.5}$$

Then generate a *pseudo-graytone function* as follows:

$$[pgf(A)](i,j) = \sum_{n=0}^{N} \sum_{(s,t)\in S_n(A)} k_n(i-s, j-t). \tag{5.6}$$

The region of support of $[pgf(A)](i,j)$ is the same as the region of support of A, and $[pgf(A)](i,j) \geq 1$ at every point in A. An example of a binary image and its pseudo-graytone function is given in Figure 5.4.

Whether or not a particular point, $(s,t) \in S_n(A)$ can be removed from the skeleton is determined as follows. If $[pgf(A)](i,j) \geq 2$ at every point in the region of support of $k_n(i-s, j-t)$, then (s,t) can be removed from the skeleton. When a point (s,t) is removed from the skeleton, then the shifted characteristic function $k_n(i-s, j-t)$ is algebraically subtracted from $[pgf(A)](i,j)$ before the algorithm proceeds to test the next skeleton point for removal. If (s,t) is not removed from the skeleton, $[pgf(A)](i,j)$ is not changed and the algorithm proceeds to test the next skeleton point for removal. This algorithm is illustrated with the flowchart in Figure 5.5. After all skeleton points in every skeleton subset have been tested, and the redundant points removed, the skeleton points that remain form the globally minimal skeleton function. For further details on this minimal skeleton algorithm and for a description of fast algorithms for morphological skeletonization, refer to [10]. Hereafter, when we refer to a skeleton function we



(a) (b)

Figure 5.4. (a) A binary image and, (b) its pseudo-graytone function. The gray level values of the pseudo-graytone function have been scaled for illustrative purposes.

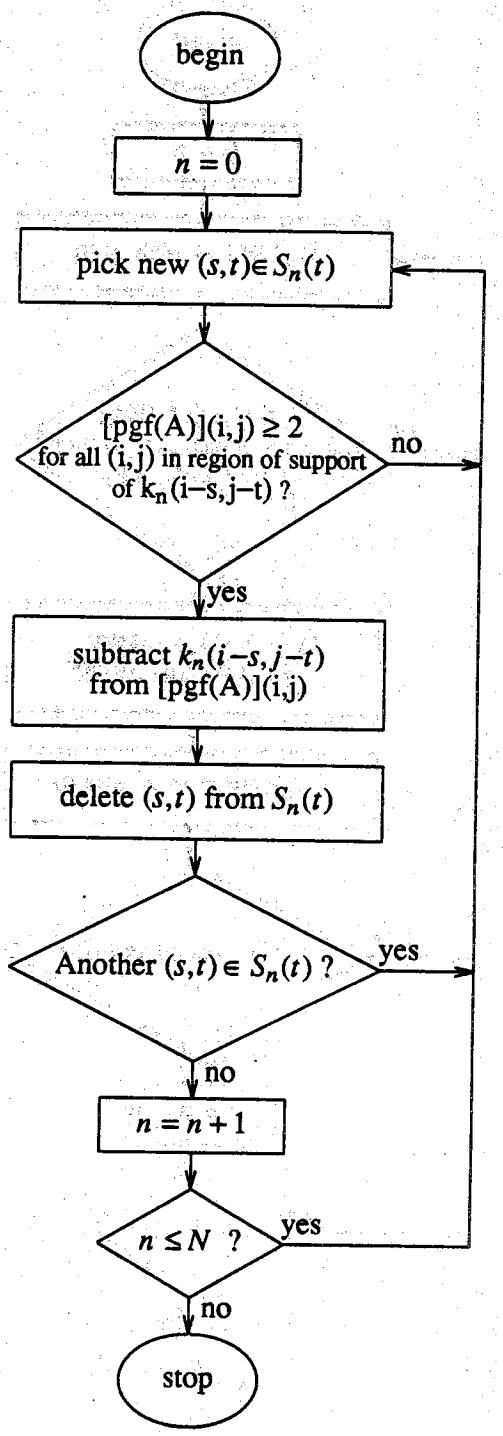


Figure 5.5. The algorithm to generate a globally minimal skeleton.

will mean the globally minimal skeleton function.

5.3.1. Skeletonization of the Image Segments

We wish to apply the skeletonization and minimal skeleton procedures described above to generate morphological skeletons for the segments in our segmented quantized image. In order to do this, the segmented quantized gray level image is decomposed into a series of binary images known as binary decomposition images (BDI). In this decomposition, any pixel in the segmented image belonging to a region with a particular gray level is set to "one" in the corresponding BDI, while all other pixels in that BDI are set to "zero." The BDI's are analogous to indicator sets for particular gray level segments in the segmented image. For example, one BDI may consist of all pixels from the segmented image that were contained in a segment with gray level 30. A BDI is generated for each different gray level in the segmented image, so the number of BDI's is equal to the number of gray levels in the segmented quantized image. For a segmented quantized image with M gray levels, we obtain a series of BDI's, each with an associated gray level, P_l:

$$(\mathrm{BDI}_l, P_l), \quad l = 0, 1, 2, \cdots, M-1. \tag{5.7}$$

The skeletonization procedures summarized above are used to generate a skeleton for each BDI. Equations 5.2 and 5.4 are used to transform each BDI into a gray level skeleton function; and the minimal skeleton algorithm is used to prune each skeleton. Since all the image segments are mutually exclusive, the net effect of skeletonizing a BDI is the skeletonization of all the image segments in that BDI. For a segmented quantized image with M gray levels, the end result of skeletonization is a set of gray level skeleton functions, each representing one BDI, and a set of associated gray levels:

$$([skf(\mathrm{BDI}_l)](i,j), P_l), \quad l = 0, 1, 2, \cdots, M-1. \tag{5.8}$$

These pairs of gray level skeleton functions and associated gray levels form a compact and uniquely reversible representation of the segmented image for coding purposes. Note that there are two categories of gray levels in the representation of Equation 5.8. First are the *associated gray levels*, P_l, $l = 0, 1,...M-1$. These indicate the gray level of the segments represented by the particular skeleton function. Since there is one skeleton for each BDI, there is also one associated gray level for each BDI, and thus one associated gray level for each skeleton. To avoid confusion, this first class of gray levels will always be referred to as associated gray levels. Second are the gray levels along each skeleton function. Each BDI has a skeleton, and each of these skeletons is composed of many different gray levels (recall that a *binary* BDI produces a *gray level* skeleton). The gray levels along a skeleton simply indicate to which skeleton subset a

particular skeleton point belongs.

5.4. Coding of the Skeletons and Associated Gray Levels

The last step in the compression process is to actually encode the ($[skf(\mathrm{BDI}_l)](i,j)$, P_l) pairs using a source coding technique. We have considered three possible schemes for this encoding. In the first approach, $[skf(\mathrm{BDI}_l)](i,j)$ is coded in its entirety for $M-1$ of the BDI's, where M is the total number of BDI's. The result is a complete skeleton encoded for a subset of the image segments. In the second approach, the 0'th skeleton subset of every $[skf(\mathrm{BDI}_l)](i,j)$ is deleted before encoding, and a $[skf(\mathrm{BDI}_l)](i,j)$ is encoded for every BDI. In other words, reduced forms of $[skf(\mathrm{BDI}_l)](i,j)$, $l = 0, 1,...M-1$ are encoded. The net effect of this approach is a skeleton coded for every image segment; however with the 0'th skeleton subset of each skeleton omitted. In the third approach, a "minimal set of segments" is selected from the image, and only skeletons for segments in this group are encoded. A $[skf(\mathrm{BDI}_l)](i,j)$ is encoded for every BDI, however, as with the second approach, the $[skf(\mathrm{BDI}_l)](i,j)$ are in a reduced form. In this case, complete skeletons are coded, but only for a subset of the image segments. These approaches are described in detail in the remainder of this section, and each is experimentally evaluated in Section 5.6.

All of the encoding techniques we have proposed above require coding of a gray level skeleton for each BDI. In [10] the authors investigated several different possible techniques for encoding a gray level skeleton, using a variety of source coding techniques. They found that the best compression was achieved by coding the shape of the gray level skeleton (i.e. $SK(A)$)in the form of a binary image, and then coding the gray levels along the skeleton function ($[skf(A)](i,j)$) using a Huffman code. They proposed using a form of runlength coding proposed by Elias [126] to code the binary image describing the skeleton function's shape. We will also use these methods to code our skeleton functions.

Runlength coding is a technique designed to work well on sparse binary signals, for example an image made up of mostly zeros, with a few ones. The image rows are catenated together to form a vector, and all runs of consecutive 0's are found. The lengths of these runs, separated by a symbol (referred to as a "comma") to mark the end of a run (i.e., the presence of a 1), completely describe the original image. The runlengths and commas are then coded using a source coding technique such as the one described in [126]. This technique involves using n symbols in an n-ary arithmetic system to represent the runlengths, and an $n+1$'th symbol to represent a comma, and is described in detail in Appendix C.

Huffman coding [127] is a variable length coding technique where the codewords are chosen based on the probability distribution of the source symbols. The idea is to assign short codewords to source symbols that occur frequently, and longer codewords to source symbols which occur less frequently.

The remainder of this section is devoted to describing the three techniques we have proposed for coding the set of ($[skf(\mathrm{BDI}_l)](i,j)$, P_l) pairs which represent the image.

5.4.1. Approach 1: Coding Complete Skeletons

The obvious approach to coding the skeleton functions and associated gray levels is to simply apply the coding techniques described above to code the skeleton function and associated gray level for every BDI. However, since the BDI's are mutually exclusive, and $\bigcup_{l=0}^{l=M-1} \mathrm{BDI}_l$ covers the entire image, it is actually only necessary to code the skeleton functions for $M-1$ of the BDI's. The shape of the missing BDI can be implied from the coded BDI's. So with this coding technique the runlength Elias method mentioned above is used to code the shape of each of the $M-1$ of the skeleton functions, and a Huffman code is used to code the gray levels along each skeleton function.

In Chapter 2 we saw that a segmented quantized image typically has on the order of 20 gray levels; and for the images we are using these 20 gray levels could be any of the 256. Since there is one skeleton function and associated gray level for each gray level in the segmented quantized image, there are also typically on the order of only twenty associated gray levels to code (one for each skeleton function). Since no two associated gray levels are the same, each P_l can be coded directly, using eight bits each. With this coding method, an *exact* duplicate of the segmented quantized image is decoded. The decoding method is described in Section 5.5.1.

5.4.2. Approach 2: Coding Without 0'th Skeleton Subsets

In the discussion in Section 5.3 relative to morphological skeletonization we saw that the 0'th skeleton subset of a BDI's skeleton consists of all the features of the BDI which are smaller than the structuring element. An example of the 0'th skeleton subset of a set is shown in Figure 5.6 In our experiments applying the skeletonization procedure to BDI's we have found that typically there are nearly as many skeleton points in the 0'th subset of a BDI's skeleton as there are in all the other skeleton subsets of the BDI combined. This means that the 0'th skeleton subsets contribute a disproportionate

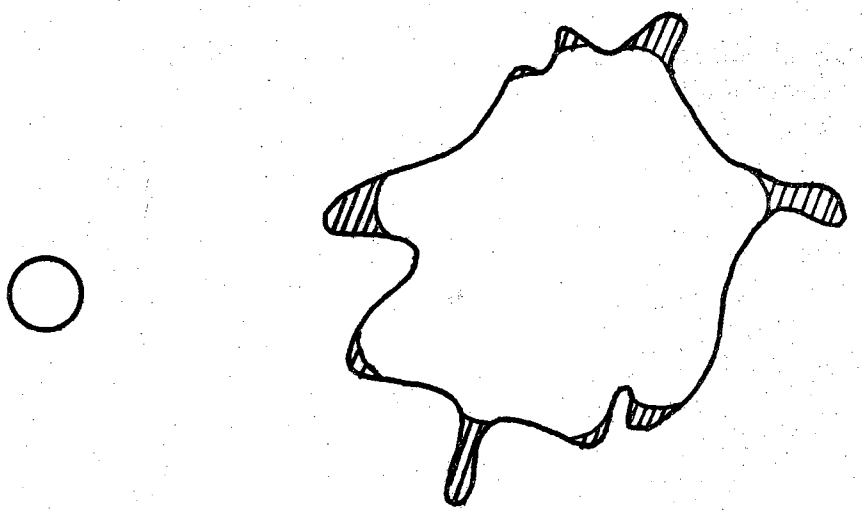


Figure 5.6. A structuring element (on the left), an image set (on the right), and the set's 0'th skeleton subset (the shaded portion of the set on the right).

amount to the total cost of coding a skeleton. For this reason, we propose omitting the 0'th skeleton subsets from the BDI skeleton functions when coding. We will simply delete the 0'th skeleton subsets from all $[skf(\mathrm{BDI}_l)](i,j)$, and then these reduced skeleton functions, denoted $[skf^*(\mathrm{BDI}_l)](i,j)$, will be coded as described in the previous section. We will still use a runlength Elias code for the reduced skeleton functions' shapes, and a Huffman code for the gray levels along the reduced skeleton functions. The P_l's can again be coded directly with eight bits each.

The cost of omitting the 0'th skeleton subsets is increased distortion in the decoded image. When the 0'th skeleton subset is deleted from $[skf(\mathrm{BDI}_l)](i,j)$, this means that certain pixels in BDI_l have no representatives in the skeleton function. Therefore, when BDI_l is reconstructed from the reduced skeleton function, these pixels cannot be reconstructed. All the features of BDI_l that were smaller than the structuring element used for skeletonization will be lost. Specifically, rather than reconstructing BDI_l, we actually reconstruct $(\mathrm{BDI}_l)_B$, the opening of BDI_l by B.

In Section 5.6 we show examples illustrating the unreconstructed pixel problem and the effect on image reconstruction of deleting the 0'th skeleton subsets from the $[skf(\mathrm{BDI}_l)](i,j)$. In Section 5.5.2 we describe the post-processing technique we propose to "fill in" the unreconstructed pixels.

5.4.3. Approach 3: Coding the Minimal Set of Segments

Consider the information represented when a skeleton is coded for *every* segment in a segmented image. Given skeletons for two neighboring segments, information about the shape of the segments' common boundary is represented in both skeletons. This means that when both of the skeletons are coded, redundant information about the segmented image is coded. This observation is the motivation behind the coding technique we propose in this section.

Consider the simplified segmented image shown in Figure 5.7. It is possible to select a subset of the segments in this image, from which the shapes of all the other image segments can be implied. The set of segments $\mathcal{A}_1 = \{a, b, c, f, g, i, j, k, m, o, q\}$ is one such subset. Together, the segments in $\mathcal{A}_1$ imply completely the shapes of the segments in the set $\mathcal{B}_1 = \{d, e, h, l, n, p\}$. We refer to the subset $\mathcal{A}_1$ as a *minimal set of segments* for the image in Figure 5.7. The minimal set of segments is not necessarily unique . There is usually more than one such set for any given segmented image. Another minimal set of segments for the segmented image of Figure 5.7 is $\mathcal{A}_2 = \{a, c, d, e, f, h, i, k, l, m, n, n, q\}$. These segments, together, completely imply the shapes of the segments in the set $\mathcal{B}_2 = \{b, g, j, o\}$. Both these minimal sets of segments

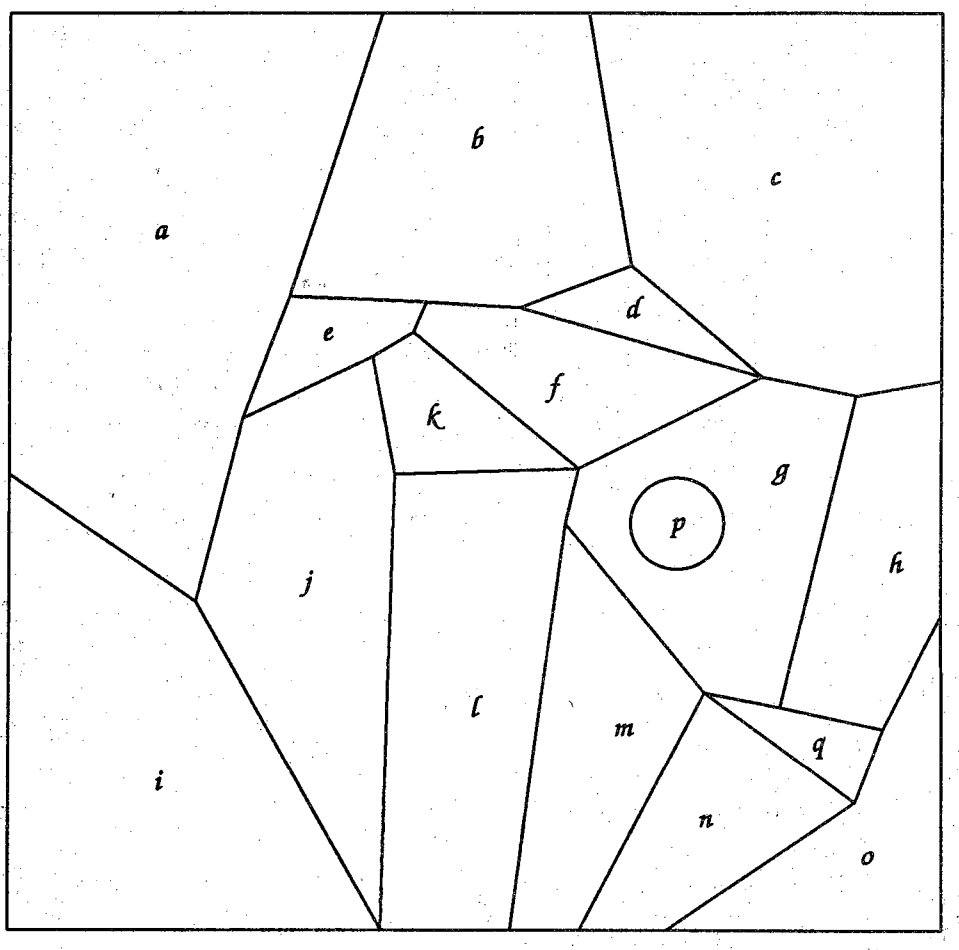


Figure 5.7. A simplified segmented image.

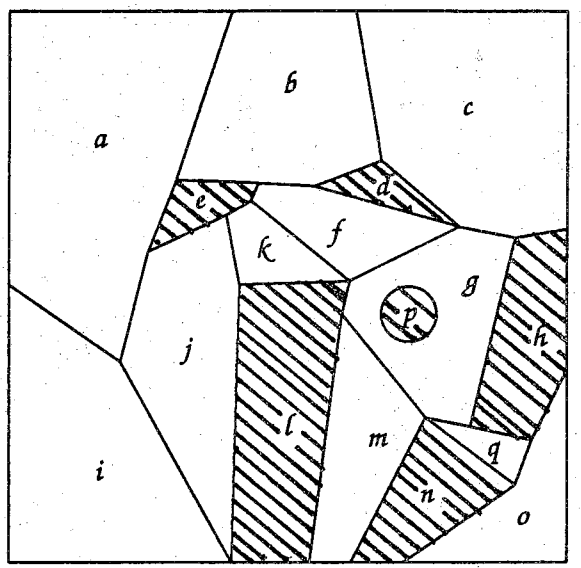
are demonstrated in Figure 5.8.

We can apply this idea of a minimal set of segments to the coding of a segmented image. Generate a minimal set of segments, call it $\mathcal{A}$, for the segmented image being coded. Since the shapes of all the segments not in $\mathcal{A}$ are implied by the shapes of the segments in $\mathcal{A}$, it is only necessary to code skeleton functions for the image segments in $\mathcal{A}$, Therefore, before skeletonization of the BDI's, all segments not in $\mathcal{A}$ can be deleted from the BDI's. We will refer to these reduced BDI's as $\mathrm{BDI}^{\dagger}{}_l$. The reduced BDI's will be composed of fewer segments, and when skeletonized will have fewer skeleton points. Thus the number of skeleton points in $[skf(\mathrm{BDI}^{\dagger}{}_l)](i,j)$, $l = 0, 1, \cdots M-1$, will be less than the number of skeleton points in $[skf(\mathrm{BDI}_l)](i,j)$, $l = 0, 1, \cdots M-1$, and therefore $[skf(\mathrm{BDI}^{\dagger}{}_l)](i,j)$, $l = 0, 1, \cdots M-1$, should require fewer bits to code.

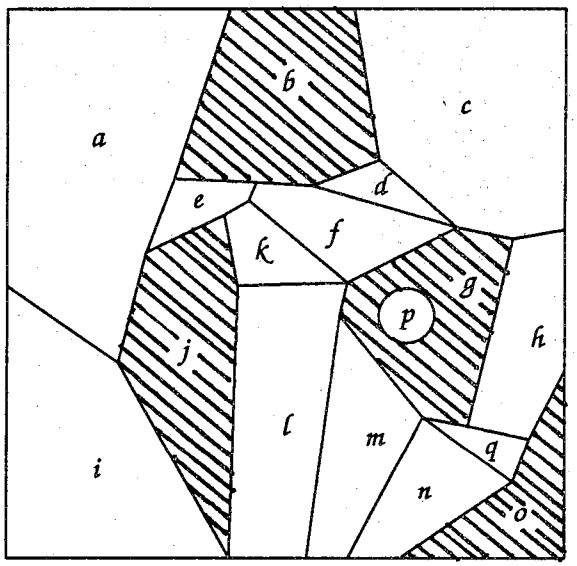
A minimal set of segments is found for a segmented image by applying the following algorithm to the segmented image. In the description of the algorithm we refer to the minimal set of segments as $\mathcal{A}$. The complement of the set $\mathcal{A}$ is $\mathcal{B}$. Then $\mathcal{A} \bigcup \mathcal{B}$ is equal to the set of all image segments. The algorithm begins at the segment at the upper left corner of the segmented image. This image segment is assigned to $\mathcal{B}$, and all image segments bordering the segment are assigned to $\mathcal{A}$. The algorithm proceeds to scan the image in a raster fashion until a segment which has not yet been assigned to $\mathcal{A}$ or $\mathcal{B}$ is encountered, and the process is repeated. The unassigned segment is assigned to $\mathcal{B}$ and all segments bordering that segment are assigned to $\mathcal{A}$. The raster scan of the image and the assigning of segments to set $\mathcal{A}$ or $\mathcal{B}$ is repeated until all image segments have been assigned. This algorithm is not necessarily an optimal algorithm. It maybe possible to apply graph theoretic concepts to develop an optimal algorithm for finding a minimal set of segments.

Figure 5.9 shows examples of actual segmented images, and binary images showing the minimal set of segments found for these images. The numbers of segments in the images and in the minimal sets of segments are given in the figure. The number of segments in a minimal set can be as little as 60% of the segments in the original segmented image. This means that we can reduce the number of segment skeleton functions that are encoded by as much as 40% by only coding skeleton functions for a minimal set of segments.

Since the segments in $\mathcal{B}$ are not included in any BDI, in order to reconstruct these segments at the decoder, we must encode the gray level of each segment in $\mathcal{B}$. The gray levels of the segments in $\mathcal{B}$ will be coded in raster scan order. Since there will be numerous segments in $\mathcal{B}$, we propose using a Huffman code for the gray levels of these segments. The same Huffman code can also be used for coding the associated gray levels, P_l. As with the two previous coding techniques, we will also use a Huffman code for coding the gray levels along the skeleton functions. With this coding method, an



(a)



(b)

Figure 5.8. Two different minimal sets of segments for the segmented image shown in Figure 5.7. The minimal set of segments consists of all the white segments. The shaded segments' shapes are implied by the minimal set of segments.

Figure 5.9. (a-b) Minimal sets of segments for two actual segmented images. The segmented images are shown on the left, and the minimal set of segments found for each of the segmented images are given by the binary images on the right. The white pixels are in segments in the minimal set of segments and the black pixels are in segments whose shapes are implied by the minimal set of segments. There are 473 segments in segmented quantized Krista and 275 segments in Krista's minimal set of segments. There are 769 segments in segmented quantized House and 480 segments in House's minimal set of segments.

exact duplicate of the segmented image is decoded. The decoding method is described in Section 5.5.3.

One point must be made about this approach to the coding of the skeleton functions. When only coding skeleton functions for a minimal set of segments, the 0'th skeleton subsets of $[skf(\mathrm{BDI}^{\dagger}_{l})](i,j)$, $l = 0, 1, \cdots M-1$, *cannot* be deleted. If these skeleton subsets were deleted, then when the segmented image was reconstructed at the decoder, it would be impossible to distinguish between image pixels not reconstructed due to missing 0'th skeleton subset points, and image pixels not reconstructed because they were in implied image segments. This ambiguity would make reconstruction of a reasonable approximation of the segmented image impossible.

5.5. Image Reconstruction

The reconstruction process for each of the three coding methods described above is slightly different. The basic outline, however, is the same for all three. The first step is to recreate the skeleton function shapes from their Elias runlength code, and then fill in the gray levels along each skeleton function from the Huffman coded versions. Once the skeleton functions are known, a morphological process is used to "grow" back the BDI's from the skeleton functions. Finally, each BDI is then "painted in" with its associated gray level value and the BDI's are combined to form the reconstructed segmented quantized image. The basic reconstruction process is illustrated in Figure 5.10. The details of decoding for each of the three coding techniques are given below.

5.5.1. Approach 1: Reconstruction From Complete Skeletons

The process by which the original quantized segmented gray level image is reconstructed from the coded version as defined in Section 5.4.1 is described here. The reconstruction process begins by decoding the shape of each skeleton function, $[skf(\mathrm{BDI}_{l})](i,j)$, from its runlength code description, and decoding the gray levels along each skeleton function from their Huffman coded versions. This information is then combined to form the set of skeleton functions, $[skf(\mathrm{BDI}_{l})](i,j)$, $l = 0, 1, \cdots M-1$, which represent the BDI's.

Using Equation 5.4, each skeleton function, $[skf(\mathrm{BDI}_{l})](i,j)$, can be transformed to a set of skeleton subsets, $S_n(\mathrm{BDI}_{l})$, $n = 0, 1, ...N_l$, where $N_l + 1$ is the number of skeleton subsets in $[skf(\mathrm{BDI}_{l})](i,j)$. This transformation is performed for $l = 0, 1, \cdots M-1$. Next, the following morphological operation is used to perfectly reconstruct BDI_{l} from its skeleton subsets:

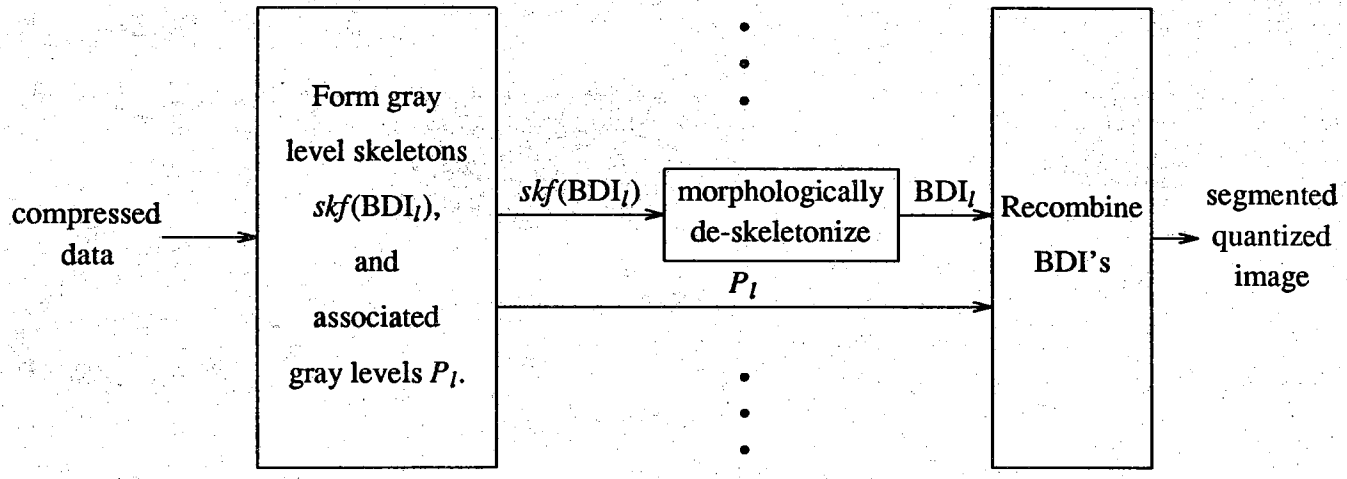


Figure 5.10. The general decoding process.

$$\mathrm{BDI}_l = \bigcup_{n=0}^{N_l} [S_n(\mathrm{BDI}_l) \ominus nB]. \tag{5.9}$$

This reconstruction is also performed for $l = 0, 1, \cdots M-1$.

After BDI_l has been reconstructed from $[skf(\mathrm{BDI}_l)](i,j)$, BDI_l is "painted in" with its associated gray level, P_l. Recall that the P_l's had been coded directly, using eight bits each. Again, this is done for $l = 0, 1, \cdots M-1$. The reconstructed segmented quantized gray level image is then formed by simply taking the union of the mutually exclusive reconstructed "painted in" BDI's. An exact replica of the original segmented quantized image is reconstructed by this process.

5.5.2. Approach 2: Reconstruction Without 0'th Skeleton Subsets

The process for reconstruction of a segmented quantized gray level image from the coded version described in Section 5.4.2 is very similar to the reconstruction process described in Section 5.5.1. As before, the reconstruction process begins by decoding the shape of each reduced skeleton function from its runlength code description, and decoding the gray levels along each reduced skeleton function from its Huffman coded version. This information is then combined to form the set of reduced skeleton functions, $[skf^*(\mathrm{BDI}_l)](i,j)$, $l = 0, 1, \cdots M-1$, which represent the BDI's. Recall that we refer to these as reduced skeleton functions because the 0'th skeleton subset is missing from each of them.

Using Equation 5.4, each reduced skeleton function, $[skf^*(\mathrm{BDI}_l)](i,j)$, can be transformed to a set of skeleton subsets, $S_n(\mathrm{BDI}_l)$, $n = 1, 1, \ldots N_l$, where N_l is the number of skeleton subsets in $[skf(\mathrm{BDI}_l)](i,j)$. This transformation is performed for $l = 0, 1, \cdots M-1$. Notice again, because we deleted the 0'th skeleton subsets before encoding the skeleton functions, there are no 0'th skeleton subsets. Next, the following morphological operation is used to perfectly reconstruct $(\mathrm{BDI}_l)_B$ from the skeleton subsets:

$$(\mathrm{BDI}_l)_B = \bigcup_{n=1}^{N_l} [S_n(\mathrm{BDI}_l) \ominus nB]. \tag{5.10}$$

As was discussed in Section 5.4.2, since the 0'th skeleton subsets were deleted from the BDI skeleton functions, all features in the BDI's that were smaller than the structuring element used in skeletonization have been lost. From a reduced skeleton function, we reconstruct $(\mathrm{BDI}_l)_B$ rather than BDI_l. Equation 5.10 is applied for $l = 0, 1, \cdots M-1$.

After $(\mathrm{BDI}_l)_B$ has been reconstructed from its reduced skeleton function,$(\mathrm{BDI}_l)_B$ is "painted in" with its associated gray level, P_l. Recall that the P_l's

had been coded directly, using eight bits each. Once more, this is done for $l = 0, 1, \cdots M-1$. As in the previous section, the reconstructed quantized segmented image is formed by taking the union over $l = 0, 1, \cdots M-1$, of the mutually exclusive reconstructed "painted in" $(\mathrm{BDI}_l)_B$. However, in this case , since some BDI points were not reconstructed from the reduced skeletons, there will be "holes" in the reconstructed image. Some image pixels will not yet have been assigned a gray level in the reconstruction process.

Post-processing is necessary to "fill in" these unreconstructed pixels. We propose an averaging filter to accomplish this. The filter scans the reconstructed segmented image in a raster fashion until it encounters an unreconstructed pixel. At each unreconstructed pixel the filter calculates the average of the *known* pixels in an eight-neighborhood of the unknown pixel. This average value is assigned to the unreconstructed pixel. When no pixels in an eight-neighborhood of an unreconstructed pixel are known, that pixel cannot be filled in. This occurs when blocks of size 3×3 pixels or larger of unreconstructed pixels exist in the image. Therefore, in order to fill in all the reconstructed pixels, multiple passes of the averaging filter are sometimes necessary. Since the filtering operation only changes unreconstructed pixels, multiple passes of the filter do not effect the known pixels in the image. This postprocessing operation completes the reconstruction process.

5.5.3. Approach 3: Reconstruction From the Minimal Set of Segments

The process for reconstruction of a segmented quantized gray level image from the coded version described in Section 5.4.3 is very similar to the reconstruction process already described in Section 5.5.1. The reconstruction process begins by decoding the shape of each reduced skeleton function from its runlength code description, and decoding the gray levels along each reduced skeleton function from its Huffman coded version. This information is then combined to form the set of skeleton functions, $[skf(\mathrm{BDI}^{\dagger}{}_l)](i,j)$, $l = 0, 1, \cdots M-1$, which represent the reduced BDI's.

Using Equation 5.4, each skeleton function, can be transformed to a set of skeleton subsets, $S_n(\mathrm{BDI}^{\dagger}{}_l)$, $n = 0, 1,...N_l$, where N_l is the number of skeleton subsets in $[skf(\mathrm{BDI}^{\dagger}{}_l)](i,j)$. This transformation is performed for $l = 0, 1, \cdots M-1$. Next, using Equation 5.9, $\mathrm{BDI}^{\dagger}{}_l$ can be perfectly reconstructed from the skeleton subsets, $S_n(\mathrm{BDI}^{\dagger}{}_l)$, $n = 0, 1,...N_l$. This reconstruction is performed for $l = 0, 1, \cdots M-1$.

Once $\mathrm{BDI}^{\dagger}{}_l$ has been reconstructed from its skeleton function, it is "painted in" with its associated gray level, P_l. Recall that the associated gray levels had been coded using a Huffman code. This is done for $l = 0, 1, \cdots M-1$. As in the previous section, the reconstructed quantized segmented image is formed by taking the union over

$l = 0, 1, \cdots M-1$, of the mutually exclusive reconstructed "painted in" $BDI^{\dagger}_{l}$. In this case when the BDI's are overlayed, since skeleton functions were only coded for image segments in $\mathcal{A}$, segments not in $\mathcal{A}$ will not have been assigned a gray level yet. But the gray levels of these segments were coded. Recall that the gray levels of all the segments not in $\mathcal{A}$ were encoded in raster scan order using a Huffman code. These gray levels can be decoded and used to fill in the unfinished image segments. This completes the decoding process. An exact replica of the original segmented image is created with this process.

5.6. Experimental Results

In this section we discuss experiments performed to evaluate various aspects of the coding algorithms presented in this chapter. The experiments were performed using the 256×256 pixel, 256 gray level test images shown in Figure 5.11. In the discussion in this section, we will refer to the image on the left in Figure 5.11 as Krista, and the image on the right as House. Segmented versions of Krista and House are shown in Figure 5.12. These images were segmented using the technique describe in Chapter 2, and the exact parameters used in the segmentation algorithm are given in the figures. Information about the number of gray levels and number of segments in the images in Figures 5.11-5.17 is summarized in Table 5.1. In the experiments discussed in this section, it will sometimes be necessary to compare the "quality" of images. Since the image are being generated for viewing by humans, we would like a quality measure that has good correspondence with human judgement of image quality. However, as was discussed in Chapter 1, no such quantitative measure is known. Therefore, it becomes necessary to compare images based on subjective visual quality. In the experiments we have performed, the visual quality of the images was determined by the authors' careful, but nonetheless, subjective evaluations of the images.

In Section 5.6.1 we examine the preprocessing procedures described in Section 5.1. The three coding approaches proposed in Section 5.4 are demonstrated in Sections 5.6.2 - 5.6.4, and are then compared in Section 5.6.5.

5.6.1. Preprocessing

Two preprocessing techniques were presented in Section 5.1. The goal of these techniques was to alter the image so that fewer segments are produced by the image segmenter, without degrading the subjective visual quality of the segmented image. Experiments to evaluate the effectiveness of these techniques are discussed in this

Figure 5.11. Two test images to be compressed. Each image is 256×256 pixels, with 256 gray levels.

$m = .123, w = 0.5$
$d = 7, th_{max} = 10,$
segments = 1925

$m = .123, w = 0.5$
$d = 13, th_{max} = 16,$
segments = 3806

Figure 5.12 Segmented versions of the test images in Figure 5.11. The parameters used in segmentation threshold TH_3, and the number of segments in the images are given below each image.

section.

The first experiment was to compare the results of segmenting an image with and without the clamping operation proposed in Section 5.1. The test images shown in Figure 5.11 were clamped according to Equation 5.1. The clamped images are shown in Figure 5.13. The clamping thresholds were chosen in order to reduce the number of gray levels in the images by a factor of two, thereby reducing by one the number of bits required to represent the gray levels. The specific threshold values were subjectively chosen for each image to be the best to achieve this, and are given in the figure. These images were then segmented using the same parameters as those used to generate the images of Figure 5.12, and the clamped and segmented images are shown in Figure 5.14. Information about the number of segments and gray levels in all the images is summarized in Table 5.1. Comparing the images in Figure 5.11 to the images in Figure 5.13, it can be seen that the clamping operation very noticeably degrades the subjective quality of all the images, and hence the segmented and clamped images of Figure 5.14 are also of lower quality than the images that were not clamped before segmenting, shown in Figure 5.12. These experiments indicate that, while clamping does reduce the number of segments in the segmented image slightly, clamping also reduces the visual quality of the segmented image noticeably. This result is contrary to the goals we stated for preprocessing. Therefore, clamping is not an effective preprocessing operation. A possible explanation for the seeming failure of the contrast sensitivity model, is that the model does not hold for the *very* highest and *very* lowest intensities. However, since the compression algorithm we have proposed includes quantization of the segmented image, it is not important to reduce the number of bits required to represent the gray levels in the image at the preprocessing stage. This will be achieved when the segmented image is requantized and therefore clamping is not necessary.

The second experiment performed in relation to preprocessing was to evaluate the effectiveness of median filtering an image before segmentation. The test images shown in Figure 5.11 were median filtered using the two-dimensional, separable 3×3 median filter described in Section 5.1. The median filtered images are shown in Figure 5.15, and the median filtered and segmented images are shown in Figure 5.16. Again, the segmented images were generated using the same segmentation threshold parameters used to generate the images of Figure 5.12, and information about the number of segments and gray levels in each image is summarized in Table 5.1. Comparing the images in Figure 5.11 to the images in Figure 5.15, it can be seen that the median filtering operation removes some important image features, for example around the eyes in the Krista image. This performance is not surprising, given the median filter's definition. Median filters are typically used to remove "spiky" noise from an image. When such noise is present in an image, image details tend to be somewhat obscured to begin with, and when the filter removes the noise, the apparent quality of the image

Figure 5.13. The images in Figure 5.11, clamped to gray levels 50-177 and 66-193, respectively. Both images have 128 gray levels after clamping.

Figure 5.14. Segmented versions of the clamped images in Figure 5.13. These images were segmented using the same segmentation thresholds as the images in Figure 5.12.

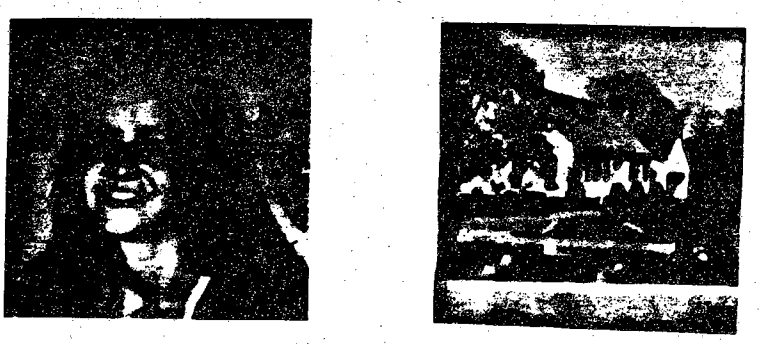


Figure 5.15. Median filtered versions of the test images in Figure 5.11. These images were filtered using a two-dimensional, 3×3, separable median filter.

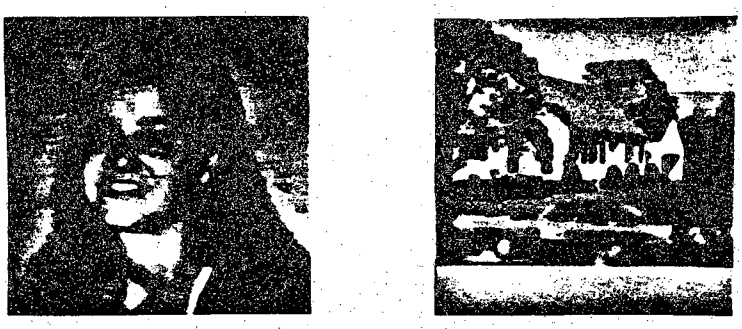


Figure 5.16. Segmented versions of the median filtered images in Figure 5.15. These images were segmented using the same segmentation thresholds as the images in Figure 5.12.

improves. Since our original image does not have much of any type of noise, image details are very visible in the original image. Therefore, it is objectionable when the filter removes these image details.

Since the median filtered images are missing important image features, the segmented versions in Figure 5.16 of the median filtered images are also missing the same important image features. This is not in keeping with the goal we stated in Section 5.1 of only removing insignificant image fluctuations. Therefore, median filtering is not an effective preprocessing operation.

From these experiments evaluating the preprocessing operations proposed in Section 5.1, we have determined that neither of the operations are appropriate for preprocessing an image before segmentation.

5.6.2. Approach 1: Coding Complete Skeletons

In this section we present the results of compressing the two segmented, post-segmentation filtered, and quantized test images shown in Figure 5.17, using the techniques outlined in Sections 5.4.1 and 5.5.1. The images in Figure 5.17 were segmented using the same threshold parameters as the image in Figure 5.12, post-segmentation filtered using E_{2a} described in Chapter 2, and quantized using the HVS-based technique outlined in Chapter 3. Information about the number of segments and gray levels in the images is summarized in Table 5.1.

The first step in this coding method is to decompose the segmented and quantized image into a series of BDI's. Examples of BDI's from each of the test images are shown in Figure 5.18. Since the Krista image in Figure 5.17 has 19 gray levels, there will be a total of 19 BDI's for Krista. Using similar reasoning, House will have 17 BDI's. In the second step of the coding algorithm, $M-1$ (M is 19 for Krista, and 17 for House) of the BDI's are skeletonized using the morphological procedure described in Section 5.3. Figure 5.19 shows the globally minimal gray level morphological skeleton functions, $[skf(\mathrm{BDI}_l)](i,j)$, of the binary BDI's shown in Figure 5.18. The final step in the coding algorithm is to generate the Elias runlength code representing the $M-1$ skeleton functions, and the Huffman code for the gray levels along the skeleton functions. Table 5.2 gives, for Krista, the number of points in each of the 19 BDI's, the number of points in each skeleton function, and the number of bits required for the Elias runlength code (with m=3) for each $[skf(\mathrm{BDI}_l)](i,j)$. Table 5.2 also gives similar information for House. Adding up the bits in the third column of Table 5.2, we see that Krista requires 109,184 bits to code the shapes of 18 skeleton functions. This is the sum of the number of bits required to code the 18 smallest of the 19 skeletons described in Table 5.2. Recall that we only need to code 18 of the 19 skeletons, and the shape of the

energy threshold = 15,
19 quant.levels,
segments = 473

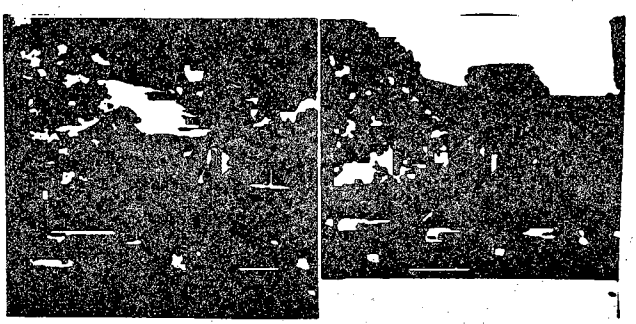
energy threshold = 30,
17 quant.levels,
segments = 769

Figure 5.17. Post-segmentation filtered and quantized versions of the segmented images in Figure 5.12. Both images were post-segmentation filtered using E_{2a}, and the images were quantized using the HVS-based quantizer described in Chapter 3. The energy thresholds used in post-segmentation filtering, the number of quantization levels for each image, and the number of segments in each image are given below each image.

Table 5.1. Summary of the numbers of segments and gray levels in the images in Figures 5.11-5.17.

Image	Gray Level Range	Number of Gray Levels	Number of Segments
5.11 Krista	14-249	234	NA
House	13-235	223	NA
5.12 Krista	19-239	193	1925
House	19-228	209	3806
5.13 Krista	50-177	128	NA
House	66-193	128	NA
5.14 Krista	50-177	122	1573
House	66-193	128	3125
5.15 Krista	15-245	230	NA
House	17-233	217	NA
5.16 Krista	19-238	188	1192
House	22-223	192	1438
5.17 Krista	25-232	19	473
House	32-217	17	769

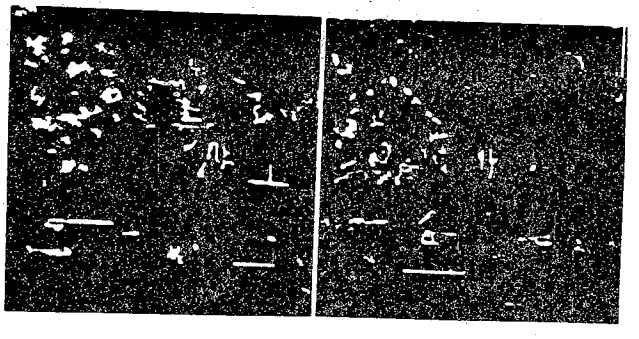
(a)



(b)

Figure 5.18. Sample BDI's from each of the images in Figure 5.17. (a) Two BDI's from Krista, (b) Two BDI's from House.

(a)



(b)

Figure 5.19. (a-b) Globally minimal morphological skeletons corresponding to the BDI's of Figure 5.18.

Table 5.2. Summary of BDI skeleton coding information for Krista and House images for the coding method (Approach 1) of Sections 5.4.1 and 5.5.1.

	Krista			House		
l	# of points in BDI_l	# of points in *skf* (BDI_l)	# of bits for Elias runlength code	# of points in BDI_l	# of points in *skf* (BDI_l)	# of bits for Elias runlength code
1	252	30	422	5165	980	6292
2	92	45	414	20477	1186	7978
3	192	89	804	6011	2283	13578
4	985	313	2436	4031	1588	9682
5	1794	543	3564	9063	2866	16716
6	3297	1044	6888	3232	1540	9344
7	4169	1512	9798	2754	1642	9784
8	3969	934	6454	1655	1336	8022
9	7599	1725	11600	1693	1079	6784
10	3425	1307	8922	1737	1159	7178
11	8292	1634	10618	1158	859	5834
12	4506	1214	8258	1470	993	6408
13	1952	891	6390	1351	1036	7026
14	3190	1184	8542	1728	962	6672
15	1326	912	6912	1381	924	7130
16	2588	1166	8498	639	363	2680
17	1893	1094	8038	1991	433	3090
18	2832	973	7444	-	-	-
19	13183	534	4782	-	-	-

19'th BDI is implied from the shapes of the other 18. In the case of Krista we do not code a skeleton for BDI_9, and for House, we do not code a skeleton for BDI_5. This number does not include a nominal number of overhead bits to signify the end of the code for one skeleton function and the beginning of the code for the next skeleton function. Adding up the 16 smallest numbers of the 17 numbers in the sixth column of Table 5.2 we see that 117,482 bits are required to code the shapes of the skeleton functions for House.

This coding technique also requires a Huffman code for the gray levels along the skeleton functions. The source symbol frequencies for Krista, and the Huffman code designed based on these frequencies is given in Table 5.3. Similar information for House is also given in Table 5.3. These source symbol frequencies are the frequencies of the gray levels along $M-1$ of the skeleton functions. Multiplying the length of each Huffman codeword by the number of times the source symbol associated with that codeword occurs and summing the results, (e.g. $(12292 \times 1) + (1827 \times 2) + (536 \times 3) +$ ...) we see that for Krista, this portion of the code requires 22,232 bits. A similar calculation results in 22,389 bits for this portion of the code for House. These bit requirement calculations do not include a nominal number of overhead bits required to transmit the Huffman codebook.

Finally, the coding technique requires eight bits to code each of the M associated gray levels. For Krista this requires 19×8 bits, or 152 bits. Similarly, House requires 17×8, or 136 bits to code its associated gray levels. Adding up the three numbers calculated for each of the images, we find that Krista requires approximately 109,184 + 22,232 + 152 = 131,568 bits, or 2.00 bpp for this coding method. Similarly, House requires approximately 117,482 + 22.389 + 136 = 140,007 bits, or 2.14 bpp for this coding method.

As detailed in Section 5.5.1, the decoding process for this coding method consists of perfectly reconstructing the 18 BDI's from the skeleton functions and then combining them to form the reconstructed image. A perfect replica of the original *segmented quantized* image is reconstructed by the decoding process. The decoding process is the exact inverse of the coding process for this coding method.

5.6.3. Approach 2: Coding Without 0'th Skeleton Subsets

In this section we present the results of compressing the two segmented and quantized test images shown in Figure 5.17, using the techniques outlined in Section 5.4.2 and 5.5.2.

Table 5.3. The source symbol frequencies and the Huffman code designed for the gray levels along the skeleton functions for the Krista and House images, using the coding technique (Approach 1) described in Section 5.4.1.

	Krista		House	
source symbol	symbol frequency *	Huffman codeword	symbol frequency **	Huffman codeword
1	12292	1	16028	1
2	1827	01	1612	01
3	536	000	361	000
4	266	00111	184	0011
5	140	00100	46	001010
6	69	0011011	29	0010111
7	36	0010101	18	0010011
8	35	0010100	16	0010001
9	27	00110100	1	00101100100
10	43	0011000	6	001001010
11	20	00101111	2	00101100111
12	17	00101110	2	00101100110
13	14	001101010	3	0010110001
14	44	0011001	3	0010110000
15	38	0010110	3	0010010111
16	10	0011010111	3	0010010110
17	3	00110101101	15	00101101
18	2	00110101100	2	00101100101
25	0	-	9	00100001
27	0	-	9	00100000
28	0	-	11	00100100

* $\times$ 15149

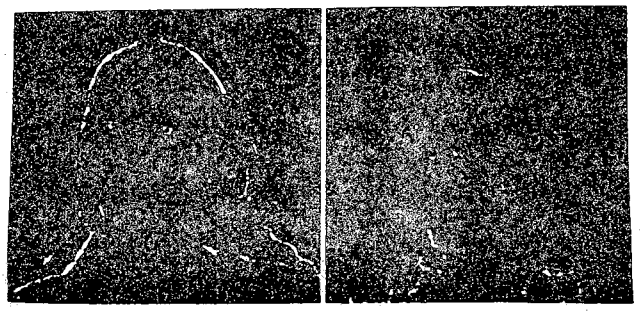
** $\times$ 18363

As in the previous section, the first step in this coding method is to decompose the segmented and quantized image into a series of BDI's. These are the same BDI's as in the previous section, and examples are given in Figure 5.18. The next step in the coding algorithm is to skeletonize all the BDI's using the morphological procedure described in Section 5.3. After the BDI's are skeletonized, the 0'th skeleton subsets of all the BDI's are discarded, as described in Section 5.4.2. Figure 5.20 shows the reduced gray level morphological skeleton functions, $[skf^*(\mathrm{BDI}_l)](i,j)$, of the binary BDI's shown in Figure 5.18. By comparing the reduced skeletons of Figure 5.20 to the complete skeletons of Figure 5.19, one can see approximately how many skeleton points are eliminated when the 0'th skeleton subset is omitted. This information can also be deduced by comparing the number of points in $[skf^*(\mathrm{BDI}_l)](i,j)$, given in Table 5.4, to the number of points in $[skf(\mathrm{BDI}_l)](i,j)$, given in Table 5.2.

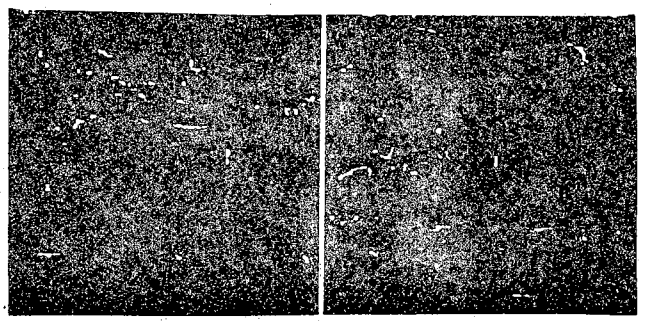
The next step in the coding algorithm is to generate the Elias runlength code representing each of the reduced skeleton functions, and the Huffman code for the gray levels along the reduced skeleton functions. Table 5.4 gives, for Krista and for House, the number of points in $[skf^*(\mathrm{BDI}_l)](i,j)$, and the number of bits required to Elias runlength code $[skf^*(\mathrm{BDI}_l)](i,j)$, for $l = 0, 1, \cdots M-1$. Adding up the bits in the second column of Table 5.4, we see that Krista requires 35,264 bits for this portion of the code. Again, this number does not include a nominal number of overhead bits to signify the end of the code for one reduced skeleton function and the beginning of the code for the next reduced skeleton function. By a similar calculation on the fourth column of Table 5.4 we see that the number of bits required for House for this stage of the coding is 25,768 bits.

As in the previous section, this coding technique also requires a Huffman code for the gray levels along the reduced skeleton functions. The source symbol frequencies for Krista and for House, and Huffman codes based on each of these distributions are given in Table 5.5. Multiplying the length of each Huffman codeword by the number of times the source symbol associated with that codeword occurs and summing the results, (e.g. $(2084 \times 1) + (613 \times 2) + (305 \times 4) + 7{,}826$ bits. A similar calculation (e.g. $(1902 \times 1) + (423 \times 2) + (205 \times 3) + \ldots$) results in 4,857 bits for this portion of the code for House. These bit requirement calculations do not include a nominal number of overhead bits required to transmit the Huffman codebook.

Finally, the coding technique requires eight bits to code each of the M associated gray levels. The bit requirements for this portion of the code are identical to the requirements of the previous section. Adding up the three numbers of bits calculated for each of the images, we find that Krista requires approximately $35{,}264 + 7{,}826 + 152 = 43{,}242$ bits, or 0.66 bpp for this coding method, and House requires approximately $25{,}768 + 4{,}852 + 136 = 30{,}761$ bits, or 0.47 bpp for this coding method.



(a)



(b)

Figure 5.20. (a-b) The reduced morphological skeletons formed by discarding the 0'th skeleton subsets of the skeletons shown in Figure 5.19.

Table 5.4. Summary of BDI skeleton coding information for Krista and House images for the coding method (Approach 2) of Sections 5.4.2 and 5.5.2.

	Krista		House	
l	number of points in $skf^*(\mathrm{BDI}_l)$	number of bits for Elias runlength code	number of points in $skf^*(\mathrm{BDI}_l)$	number of bits for Elias runlength code
1	30	30	300	782
2	11	20	285	861
3	24	66	380	1874
4	89	88	227	1082
5	137	406	430	2529
6	270	915	164	1097
7	317	1190	153	1443
8	259	750	59	991
9	469	1167	92	784
10	255	1119	88	964
11	272	1077	50	575
12	203	914	82	754
13	109	751	62	530
14	187	1059	117	729
15	78	682	60	776
16	200	986	53	94
17	105	691	163	395
18	175	597	-	-
19	406	534	-	-

Table 5.5. The source symbol frequencies and the Huffman code designed for the gray levels along the skeleton functions for the Krista and House images, using the coding technique (Approach 2) described in Sections 5.4.2 and 5.5.2.

	Krista		House	
source symbol	symbol frequency *	Huffman codeword	symbol frequency **	Huffman codeword
2	2084	1	1902	1
3	613	00	423	00
4	305	0111	205	010
5	168	0100	65	01110
6	87	01011	39	011111
7	57	011001	26	011010
8	49	011000	22	011001
9	31	0110101	2	0110110111
10	43	010100	9	0110000
11	20	01101101	4	011110001
12	17	01101100	3	011011001
13	14	01101000	4	011110000
14	44	010101	9	01111011
15	38	0110111	3	011011000
16	10	011010011	3	011011010
17	3	0110100101	15	0110111
18	2	0110100100	2	0110110110
25	0	-	9	01111010
27	0	-	9	01111001
28	0	-	11	0110001

* $\times$ 3585

** $\times$ 2765

The first step in decoding the segmented quantized image from the code described above is to reconstruct the reduced skeleton functions, $[skf^*(\mathrm{BDI}_l)](i,j)$, as described in Section 5.5.2. Then using the morphological process of Equation 5.10, the set $(\mathrm{BDI}_l)_B$, $l = 0, 1, \ldots M-1$ is reconstructed. Figure 5.21 show these versions of the BDI's from Figure 5.18. From a close comparison of Figures 5.21 and 5.18 the "unreconstructed pixel" problem is apparent. It can be seen that the fine details of the BDI's in Figure 5.18 are not recreated in the images in Figure 5.21. Reiterating, this is due to the fact that the 0'th skeleton subsets of the skeleton functions were not encoded.

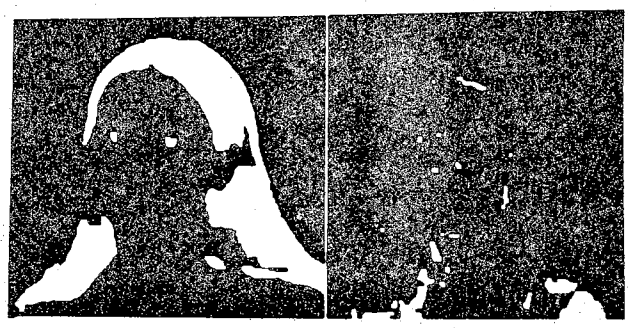
The next step in decoding is to "paint in" each reconstructed BDI with its associated gray level, P_l, and then combine these images as described in Section 5.5.2, to begin to form the reconstructed image. Figure 5.22 shows the reconstructed images at this point in the algorithm. In Figure 5.22 the unreconstructed pixel is again apparent. We can see in both images that many image pixels have not been assigned a gray level (these pixels appear black in Figure 5.22). In order to demonstrate the problem more clearly, Figure 5.23 shows in black all the unreconstructed pixels for each test image. The caption of Figure 5.23 tells exactly how many unreconstructed pixels appear in each test image.

The final step to complete the decoding is post-processing to "fill in" these unreconstructed pixels. An averaging filter to accomplish this is described in detail in Section 5.5.2. Figure 5.24 shows the images resulting from applying this filter to the images of Figure 5.22. Comparing the reconstructed image of Figure 5.24 the the original segmented quantized images of Figure 5.17, distortion in the reconstructed images is quite apparent. This coding technique does not perfectly reconstruct the segmented quantized images of Figure 5.17.

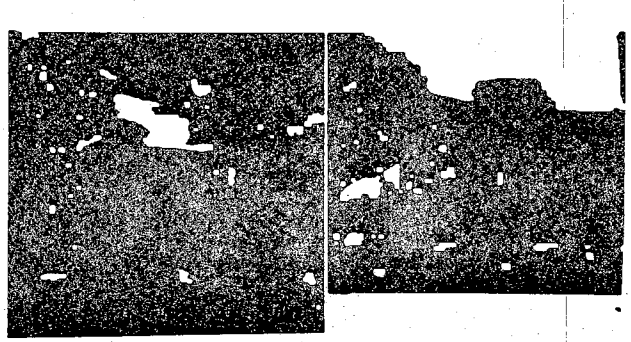
5.6.4. Approach 3: Coding the Minimal Set of Segments

In this section we present the results of compressing the two segmented and quantized test images shown in Figure 5.17, using the techniques outlined in Section 5.4.3 and 5.5.3.

With this coding method, the first step after segmenting and quantizing the image is to find a minimal set of segments for the segmented and quantized image. Using the algorithm described in Section 5.4.3, this was done for each of the images in Figure 5.17. The minimal set of segments for each test image is illustrated in Figure 5.25. The caption of Figure 5.25 gives the number of segments in the minimal set of segments found for each of the test images. The segments in white in Figure 5.25 are the segments that will be in the reduced BDI's, and eventually have their skeleton functions encoded. The image segments in black in Figure 5.25 will not have skeleton functions



(a)



(b)

Figure 5.21. (a-b) The BDI's reconstructed from the reduced skeletons of Figure 5.20.

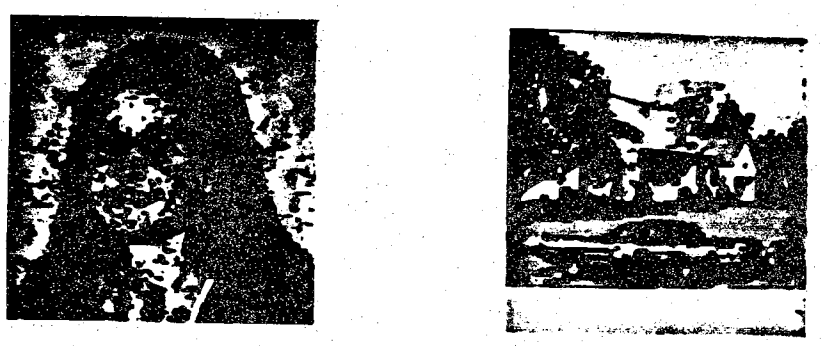


Figure 5.22. The segmented images reconstructed from reduced morphological skeletons like those shown in Figure 5.20 (that is, skeletons missing their 0'th skeleton subsets). The black pixels are unreconstructed pixels.

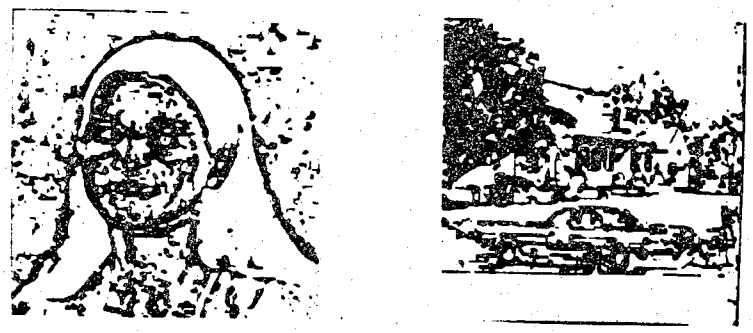


Figure 5.23. Binary images demonstrating the unreconstructed pixels in the segmented images of Figure 5.22. White pixels will be reconstructed from reduced skeletons, black pixels will not. There are 13,548 (21%) unreconstructed pixels in Krista, and 18,464 (28%) unreconstructed pixels in House.

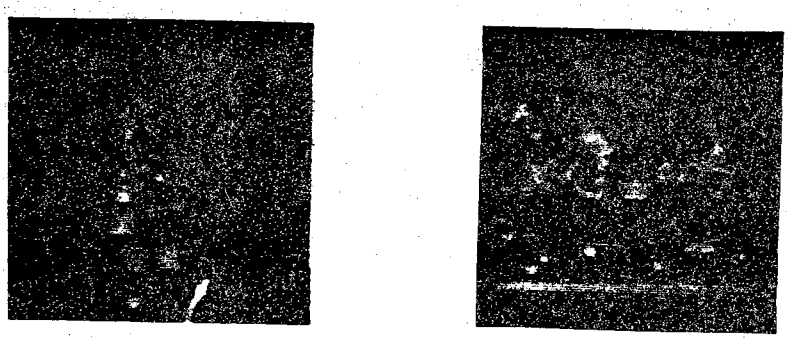


Figure 5.24. The reconstructed segmented images of Figure 5.22 after the averaging filter designed to fill in the unreconstructed pixels. The Krista image required five passes of the filter, the House image required ten passes of the filter.

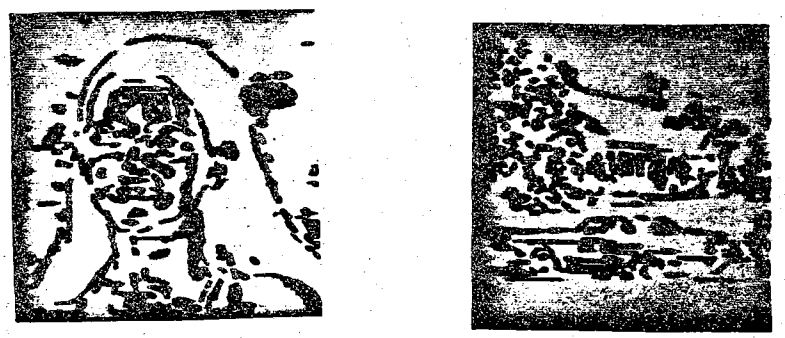


Figure 5.25. Minimal sets of segments for the segmented quantized images of Figure 5.17. The white pixels are in segments in the minimal set of segments and the black pixels are in segments whose shapes are implied by the minimal set of segments. There are 473 segments in segmented quantized Krista and 275 segments in Krista's minimal set of segments. There are 769 segments in segmented quantized House and 480 segments in House's minimal set of segments.

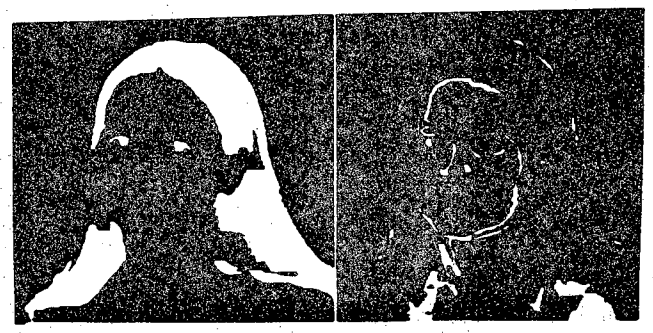
coded.

After the segments not in the minimal set have been set to zero in the segmented quantized image, the next step is to generate the series of reduced of BDI's, $BDI^{\dagger}_{l}$, $l = 0, 1, ...M-1$. Examples of reduced BDI's for each of the test images are shown in Figure 5.26. These reduced BDI's correspond to the complete BDI's of Figure 5.18. Notice that the reduced BDI's in Figure 5.26 have much fewer segments than the complete BDI's in Figure 5.18. Table 5.6 gives, for Krista and House, the number of points in $BDI^{\dagger}_{l}$ for $l = 0, 1, ...M-1$. The next step in the coding algorithm is to skeletonize the reduced BDI's using the morphological procedure described in Section 5.3. Figure 5.27 shows the gray level morphological skeletons, $[skf(BDI^{\dagger}_{l})](i,j)$, of the reduced BDI's shown in Figure 5.26. Comparing the numbers in Table 5.2 to those in Table 5.6 it can be seen that the skeleton functions for the reduced BDI's have significantly fewer points than the skeleton functions for the complete BDI's. This can also be seen by comparing the skeletons in Figure 5.27 to those in Figure 5.19.

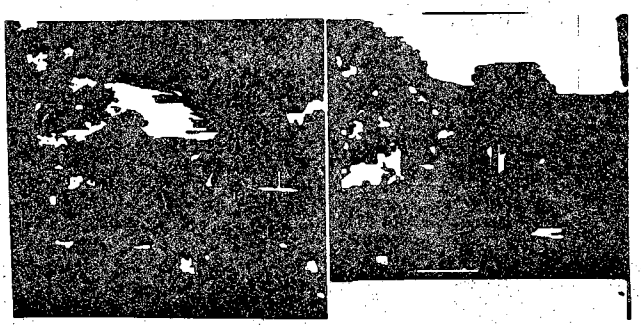
The next step in the coding algorithm is to generate the Elias runlength code representing each of the skeleton functions, and the Huffman code for the gray levels along the skeleton functions. Table 5.6 gives, for Krista and for House, the number of points in each of the reduced BDI's, the number of points in each skeleton function, and the number of bits required for the Elias runlength code for each $[skf(BDI^{\dagger}_{l})](i,j)$. Adding up the bits in the third column of Table 5.6, we see that Krista requires 93,424 bits for this portion of the code. Once more, this number does not include a nominal number of overhead bits to signify the end of the code for one skeleton function and the beginning of the code for the next skeleton function. The number of bits required for House for this stage of the coding (the sum of the numbers in the sixth column of Table 5.6) is 104,922 bits.

As with the other two techniques, this coding technique requires a Huffman code for the gray levels along the skeleton functions. The source symbol frequencies for Krista and for House, and the Huffman codes designed based on these distributions are given in Table 5.7. Multiplying the length of each Huffman codeword by the number of times the source symbol associated with that codeword occurs and summing the results, (e.g. $(10{,}064 \times 1) + (1631 \times 2) + (521 \times 3) + 19{,}969$ bits. A similar calculation (e.g. $(13{,}969 \times 1) + (1{,}477 \times 2) + (385 \times 3) + ...$) results in 21,930 bits for this portion of the code for House. These bit requirement calculations do not include a nominal number of overhead bits required to transmit the Huffman codebook.

Finally, this coding technique requires a second Huffman code to be used for the gray levels of the image segments not in the minimal set of segments, along with the associated gray levels. The frequencies of these source symbols, and the Huffman code designed for Krista is shown in Table 5.8. The same information for House is given in

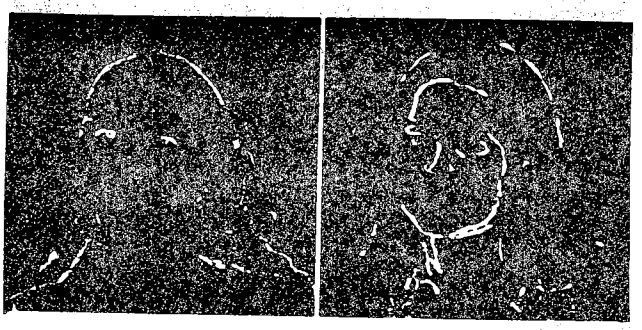


(a)

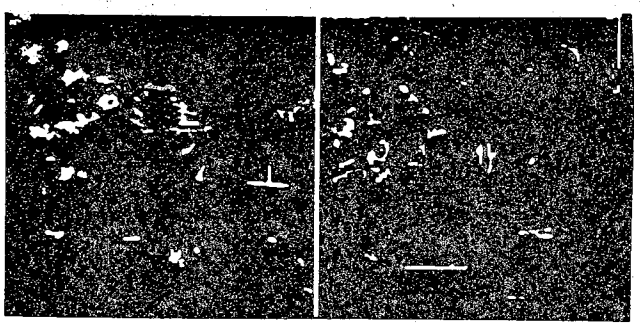


(b)

Figure 5.26. (a-b) The BDI's of Figure 5.18, with segments not in the minimal set removed.



(a)



(b)

Figure 5.27. (a-b) Globally minimal morphological skeletons corresponding to the reduced BDI's of Figure 5.26.

Table 5.6. Summary of BDI skeleton coding information for Krista and House images for the coding method (Approach 3) of Sections 5.4.3 and 5.5.3.

	Krista			House		
l	number of points in $BDI^{\dagger}_{l}$	number of points in $skf(BDI^{\dagger}_{l})$	number of bits for Elias runlength code	number of points in $BDI^{\dagger}_{l}$	number of points in $skf(BDI^{\dagger}_{l})$	number of bits for Elias runlength code
1	252	30	422	4646	782	5096
2	57	20	182	19710	861	6008
3	130	66	616	5460	1874	11324
4	223	88	670	3161	1082	6872
5	1513	406	2716	8600	2529	14900
6	2926	915	5964	2570	1097	6800
7	3420	1190	7768	2509	1443	8566
8	3651	750	5216	1233	991	6106
9	5480	1167	8298	1333	784	5182
10	3054	1119	7706	1494	964	5968
11	7584	1077	7054	831	575	4118
12	3963	914	6246	1163	754	4854
13	1645	751	5410	719	530	4002
14	2849	1059	7794	1117	729	5278
15	1078	682	5426	1097	776	6252
16	2388	986	7294	129	94	720
17	1092	691	5206	1832	395	2876
18	2411	597	4664	-	-	-
19	13183	534	4782	-	-	-

Table 5.7. The source symbol frequencies and the Huffman code designed for the gray levels along the skeleton functions for the Krista and House images, using the coding technique (Approach 3) described in Sections 5.4.3 and 5.5.3.

	Krista		House	
source symbol	symbol frequency *	Huffman codeword	symbol frequency **	Huffman codeword
1	10064	1	13969	1
2	1631	01	1477	01
3	521	000	385	000
4	265	00111	197	0010
5	156	00100	62	001110
6	81	001010	39	0011111
7	53	0011001	26	0011010
8	49	0011000	22	0011001
9	31	00110101	2	00110110110
10	43	0010110	9	00110000
11	20	001101101	4	0011110001
12	17	001101100	3	0011011001
13	14	001101000	4	0011110000
14	44	0010111	9	001111011
15	38	00110111	3	0011011000
16	10	0011010011	3	0011011010
17	3	00110100101	15	00110111
18	2	00110100100	2	0011011011
25	0	-	9	001111011
27	0	-	9	001111001
28	0	-	11	00110001

* $\times$ 13042

** $\times$ 16260

Table 5.8. The source symbol frequencies and the Huffman code designed for the gray levels of the segments not in the minimal set of segments and the associated gray levels for the Krista image, using the coding technique (Approach 3) described in Sections 5.4.3 and 5.5.3.

	Krista	
source symbol	symbol frequency *	Huffman codeword
42	8	11011
59	18	1111
75	10	0000
86	13	1000
95	12	0101
104	8	11010
110	14	1100
120	19	001
125	12	0110
132	25	101
139	12	0111
145	16	1110
156	5	01000
164	10	0001
175	13	1001
187	3	010010
202	5	010011

* $\times$ 186

Table 5.9. Multiplying the codeword lengths by the number of times the source symbol associated with that codeword occurs, and summing the results, we calculate that Krista requires 805 bits for this portion of the code. A similar calculation results in 1,211 bits for this portion of the code for House. Adding up the three numbers of bits calculated for each of the images, we find that Krista requires approximately 93,424 + 19.969 + 805 = 114,125 bits for this coding method, and House requires approximately 104,922 + 21,930 + 1,211 = 127,451 bits for this coding method. These numbers provide a bit rate of 1.74 bpp for Krista and 1.94 bpp for House. All the bit rates discussed in this section and the preceding two sections are summarized in Table 5.10.

The first step in decoding the segmented quantized image from the code described above is to reconstruct the skeleton functions, $[skf(\mathrm{BDI}^{\dagger}{}_{l})](i,j)$, as described in Section 5.5.2. Then using the morphological process of Equation 5.10, the set $\mathrm{BDI}^{\dagger}{}_{l}$, $l = 0, 1, \ldots M-1$ is exactly reconstructed. These reduced BDI's will be identical to those shown in Figure 5.26.

The next step in decoding is to "paint in" each reconstructed reduced BDI with its associated gray level, P_l, and then combine these images as described in Section 5.5.2, to begin to form the reconstructed image. Recall that the P_l values were encoded using a Huffman code. Figure 5.28 shows the reconstructed segmented quantized images at this point in the reconstruction algorithm. We can see in both images in Figure 5.28 that the image segments not in the minimal set of segments have not been assigned a gray level yet (these segments appear black in Figure 5.28).

The final step to complete the decoding is post-processing to "fill in" the unreconstructed segments with their gray level values, which were encoded in raster scan order, using a Huffman code, as described previously. A perfect replica of the original segmented quantized image is reconstructed by this decoding process.

5.6.5. Comparisons

We have proposed three techniques for coding a segmented quantized gray level image using morphological skeletons. The bits required and the resulting bit rates for the three methods are summarized in Table 5.10. We will refer to the method from Sections 5.4.1 and 5.5.1 as "Approach 1," the method from Sections 5.4.2 and 5.5.2 as "Approach 2," and the method from Sections 5.4.3 and 5.5.3 as "Approach 3," These results show that the lowest bit rate is attained using Approach 2. However, this method introduced substantial distortion in the decoded segmented image.

Approach 1 and Approach 3 both resulted in perfect recreation of the segmented quantized image at the decoder output. Of these two methods, the Approach 3 had a

Table 5.9. The source symbol frequencies and the Huffman code designed for the gray levels of the segments not in the minimal set of segments and the associated gray levels for the House image, using the coding technique (Approach 3) described in Sections 5.4.3 and 5.5.3.

	House	
source symbol	symbol frequency *	Huffman codeword
32	5	00010
52	19	1001
66	10	00011
82	13	11101
93	31	001
102	17	0000
111	19	1010
117	13	11110
122	13	11111
129	20	1011
135	12	11100
143	17	0110
153	17	0111
166	22	1100
177	24	1101
199	18	1000
217	32	010

* $\times 285$

Table 5.10. Summary of coding requirements for Krista (Figure 5.17), House (Figure 5.17) , and Krista2 (Figure 5.29).

image	approach number	bits for skeleton shapes	bits for skeleton gray levels	bits for associated gray levels	total bits	bitrate (bpp)
Krista	1	109,184	22,232	152	131,568	2.00
	2	35,264	7,826	152	4,3242	0.66
	3	93,424	19,969	805 *	114,198	1.74
House	1	117,482	22,389	136	140,007	2.14
	2	25,768	4,852	136	30,761	0.47
	3	104,922	21,930	1,211 *	128,063	1.95
Krista2	3	32,630	10,678	161 *	43,469	0.66

* Also includes bits to code gray levels for segments not in the minimal set.

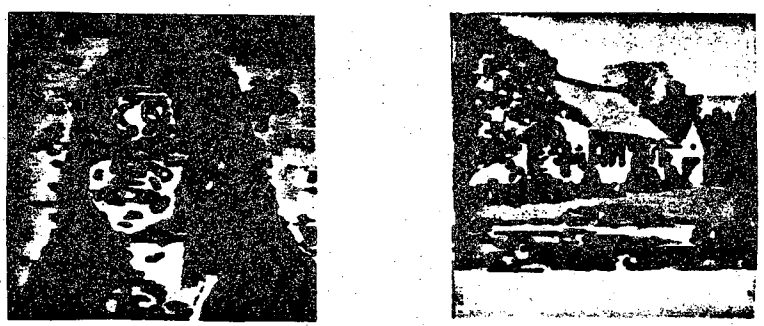


Figure 5.28. The segmented quantized images reconstructed from reduced BDI's like those shown in Figure 5.27 (that is, BDI's without segments not in the minimal set). The black pixels are in segments which are not in the minimal set of segments.

lower data rate. Recall that Approach 3 made use of the minimal set of segments. By using this idea, we achieved a reduction in the data rate of between nine and thirteen percent.

5.6.6. A Low Bit Rate Example

We have applied our new compression method, using Approach 3, to compress a more coarsely segmented and quantized image. The segmented quantized image we have compressed, referred to as Krista2, is shown in Figure 5.29. This image is composed of 79 segments, with 9 gray levels, and was segmented and quantized using the methods described in Chapters 2 and 3. The parameters used in those algorithms are given in the figure. Figure 5.30 shows the minimal set of segments that was found for this image. There are 35 segments in the minimal set, and 45 segments have their shapes implied by the minimal set. Table 5.11 gives the coding information for Krista2. Adding up the bits required for each skeleton function we see that this image requires 32,630 bits to code the shapes of the skeleton functions. A Huffman code was designed for the gray levels along the skeleton functions, and these gray levels were found to require 10,678 bits to encode. Finally, a second Huffman code was designed for the gray levels of the segments not in the minimal set, along with the associated gray levels. These were found to require 161 bits for coding. Adding we find that the image of Figure 5.29 takes 43,469 bits to code, for a bit rate of 0.66 bpp. The bit requirements for Krista2 are summarized in Table 5.10. This example illustrates that by relaxing the visual quality requirements on the segmented quantized image, the bit rate can be lowered substantially.

5.7 Comparison to Boundary Coding

As mentioned above, one significant difference between the compression method we propose and other segmentation-based image compression methods is the method for coding the segment shapes. Where we propose coding segment skeletons, other have coded segment boundaries to represent the segment shapes. In this section we will compare the bit rates achievable with both these methods applied to the same segmented quantized images. In the previous section we calculated the bit rates for the images in Figure 5.17 using our new compression technique. Now we will estimate the bit rates for these images using a boundary coding segmentation-based compression technique.

$m = .123, w = 0.5,$
$d = 22, th_{\max} = 25,$
energy threshold = 20,
9 quant. levels,
segments = 79

Figure 5.29. The segmented quantized image Krista2 to be compressed. The image was segmented using TH_3 with the parameters given below the image, and quantized to nine gray levels using the HVS-based quantizer described in Chapter 3.

Figure 5.30. The minimal set of segments for the segmented quantized image of Figure 5.29. The white pixels are in segments in the minimal set of segments and the black pixels are in segments whose shapes are implied by the minimal set of segments. There are 82 segments in segmented quantized Krista2 and 35 segments in Krista2's minimal set of segments.

Table 5.11. Summary of BDI skeleton coding information for Krista2 (Figure 5.29) for the coding method (Approach 3) of Sections 5.4.3 and 5.5.3.

l	number of points in $BDI^{\dagger}_{l}$	number of points in $skf\,(BDI^{\dagger}_{l})$	number of bits for Elias runlength code
1	324	43	474
2	412	115	816
3	2353	180	1462
4	3650	472	3442
5	21826	1011	6826
6	885	212	1518
7	5934	981	7140
8	4276	890	6320
9	16581	490	4632

For boundary coding methods, two pieces of information are coded to describe the segmented image: a binary image of the boundaries of the image segments, and the gray level of every segment in the image. Figure 5.31 shows the binary edge images for the segmented quantized images of Figure 5.17. These images were coded using the same runlength Elias method as was used previously in this chapter to code skeletons. The Krista image required 77,092 bits to represent the segment boundaries, and the House image required 86,834 bits to represent the segment boundaries. A Huffman code was designed to code the segment gray levels for each image. The Krista image has 473 segments and was found to require a total of 1,859 bits to code the gray levels of those segments. The House image has 769 segments and was found to require a total of 3,068 bits to code the gray levels of those segments. Adding, Krista requires a total of 78,951 bits, for a bit rate of 1.20 bpp, and House requires a total of 89,902 bits, for a bit rate of 1.37 bpp. These bit rates are approximately thirty percent lower than the bit rates we obtained using skeleton coding segmentation-based compression (1.74 bpp for Krista and 1.95 bpp for House). The data rates are summarized in Table 5.12.

There have been other methods proposed to code the boundaries using fewer bits than above. It has been found that by estimating the segment boundaries using line segments and arcs, bit rates in the neighborhood of 1.2 bits per contour point are achievable for the boundary image [2]. Using this result, since the edge image for Krista has 16,566 points, it may be possible to code the segment boundaries with as few as 19,879 bits, resulting in an overall bit rate of 0.33 bpp. Similarly, the edge image for House has 19,850 points, requiring 23,820 bits, resulting in an overall bit rate of 0.41 bpp. The data rates are also summarized in Table 5.12.

5.8 Conclusions

These results indicate that, using the present methods, data rates in the neighborhood of 1.5 to 2.0 bpp are attainable with the compression method we have proposed. These rates are somewhat higher than those achieved by coding boundaries rather than skeletons. This result may be due in part to the significant efforts that have been devoted in the past to efficient schemes for coding boundary images. Similar long term efforts have not been spent on the problem of coding morphological skeletons. Such efforts would almost certainly lead to a more efficient method for coding our skeletons than the one we have used.

Though morphological skeletons may result in a higher data rate in a segmentation-based compression scheme, there are also certain advantages to using morphological skeletons. One advantage is that the skeleton method for segmentation-based compression is a more parallel approach than boundary coding. This allows for a

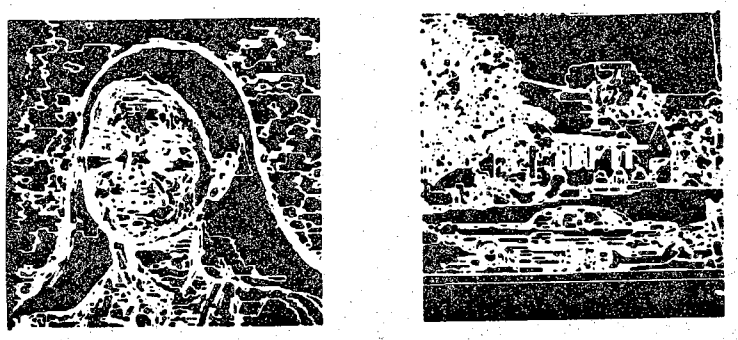


Figure 5.31. The binary edge images of the segmented quantized images of Figure 5.17.

Table 5.12. Summary of bit requirements for Krista (Figure 5.17), and House (Figure 5.17), for boundary segmentation-based compression.

image	method	bits for boundary image	bits for segment gray levels	total bits	bitrate (bpp)
Krista	runlength	77,092	1,859	78,951	1.20
	line and arc	19,879	1,859	21,738	0.33
House	runlength	86,834	3,068	89,902	1.37
	line and arc	23,820	3,068	26,888	0.41

faster implementation of the coding algorithm. The algorithm for finding and coding the segment boundaries is not conducive to being done in a parallel fashion. The skeletonization of the BDI's, and the coding of each skeleton are perfectly suited for parallel implementation, both at the coder and at the decoder.

Another advantage of the skeleton method for segmentation-based compression is that it allows more readily for data rate/image quality trade-offs. After an image has been segmented, the data rate can be varied by varying the number of 0'th skeleton subsets that get coded. For example, to get a low data rate, do not code any 0'th skeleton subsets. Our experiments have shown the data rate is typically reduced by a factor of three to four by not coding any 0'th skeleton subsets. The data rate can be gradually increased by increasing the number of 0'th skeleton subset points encoded. The data rate can also be varied to a lesser extent by varying the number of quantization levels in the segmented image quantizer. This will change slightly the number of segments in the image, and the number of bits required to code the associated gray levels. It will also effect the number of bits needed to code the gray levels of the segments not in the minimal set of segments. In contrast, the only plausible way to vary the data rate for boundary segmentation-based image compression is to completely re-segment the image.

One more advantage of skeleton segmentation-based compression algorithms is that they are well-suited to allow progressive reconstruction of the image at the decoder. With skeleton techniques, the image is represented in a hierarchical fashion. The higher-order skeleton subsets will to reconstruct a coarse estimate of the image, that is an estimate of the image composed of large, "blobby" segments. The lower the order of the skeleton subset, the finer the image detail represented by that subset. Therefore, the lower order skeleton subsets can be gradually included in the image reconstruction to progressively add finer detail in the image. This type of progression is not possible with boundary segmentation-based compression techniques.

In this investigation into skeleton segmentation-based image compression we have found that the data rate possible with these techniques is presently higher than that for boundary segmentation- based compression techniques. However, there are many important advantages of skeleton over boundary coding that may, for some applications, offset the disadvantage of a somewhat higher data rate.

CONCLUSIONS

In this thesis we proposed a new segmentation-based gray level image compression technique, which achieves data rates in the neighborhood of 1.5 t0 2.0 bpp. The formulation of this technique required investigations into image segmentation and quantization, and included the application of mathematical morphology in a new way to compress gray level images.

Our compression technique is different in several key ways from other segmentation-based image compression schemes. First, we employ an improved version of a previously proposed image segmentation technique (centroid-linkage region growing). This improved segmentation method takes advantage of HVS properties to achieve visually pleasing image segmentation. A second difference in our compression technique is that we propose quantization of the segmented image to reduce the number of gray levels in the segmented image. This results in reduction in the bit rate required for the image. Our compression technique also employs a new representation for the image segment shapes. Other segmentation-based gray level image compression techniques have typically represented the image segments by encoding the segment boundaries. We use skeletons generated using mathematical morphology to represent the segment shapes. This application of morphology is also new in the sense that we use morphological skeletons in the compression of gray level images, while others have only used morphological skeletons for binary image compression.

Our research has resulted in contributions in the areas of image segmentation, quantization, and compression. We systematically designed a version of the centroid-linkage region growing algorithm which incorporates HVS properties to produce visually pleasing segmented images. This design entailed investigations and comparisons of several different segmentation thresholds. We also investigated a method for filtering segmented images to remove visually insignificant segments. This required comparisons and evaluations of numerous different measures for the energy in an image segment. We then evaluated the interactions between the steps in our algorithms through subjective tests.

We proposed the quantization of segmented images, and showed that quantization can be done to reduce by a factor of 2 the number of bits required to code the gray levels in the segmented image, with little or no degradation in the quality of the segmented

image. We proposed a HVS-based quantizer, and compared this quantizer through subjective tests to several other quantizers. We also investigated the interactions between our segmentation and quantization algorithms.

Finally, we applied our results in segmentation and quantization to a new image compression technique. This technique employed morphological skeletons in a new way for gray level image compression. We investigated the appropriateness of two preprocessing steps which had been previously proposed by other researchers. We proposed the idea of the "minimal set of segments," which reduced the data rates achieved by our compression technique by approximately 13%. Finally, we compared our compression technique to other segmentation-based image compression methods.

There are many aspects of the work presented here that offer avenues for further research. In relation to image segmentation, investigations into different criteria for segmentation are possible. For example, others have used polynomial models for the image segments. It may be possible to use this type of segmentation criteria while also capitalizing on HVS properties. Further work needs to be done to verify and calibrate the measure proposed in Section 2.4 for the number of segments required by an image. This would require the use of some type of quantitative measure for segmented image quality.

The performance of the HVS-based quantizer proposed in Chapter 3 could possibly be improved by incorporating a priori information relative to the image being quantized, perhaps in the form of the image histogram. It may also be appropriate to spatially vary the quantizer characteristics, according to some local image characteristic.

Further work is also possible to improve the data rate achieved by the compression technique proposed. For example, perhaps a better technique for coding the skeletons of the image segments could be found. In relation to this, one possible improvement has to do with the manner in which the skeletons for the BDI's are generated. Suppose the BDI's were given labels from 0 to M-1. Using the binary representation for these labels, the BDI's could be grouped in a bit-plane fashion. For example, all BDI's having labels with the most significant bit a "1" would be logically OR'ed to form a binary image, and a morphological skeleton could be generated for this binary image. This method would result in $\log_2 M$ morphological skeletons (one for each bit required to represent the labels), rather than M skeletons (one for each BDI). Since fewer skeletons would need to be coded, the bit rate required may also be reduced.

The compression technique we proposed makes use of the idea of a minimal set of segments. Recall that this minimal set is not unique. Since a skeleton is coded for each segment in the minimal set, and each segment skeleton requires a different number of bits to code, we would like to choose the minimal set of segments which requires the overall fewest bits to code, i.e. the *optimal* minimal set of segments. Further research

could lead to a technique to find the optimal minimal set of segments.

Research such as that described above would certainly reduce the data rate achieved by our compression method. We feel that by implementing the improvements discussed above, data rates in the neighborhood of 0.4 to 0.6 bpp could be achieved. Such a data rate would make our compression technique comparable with other segmentation-based image compression schemes.

Another area for research to improve the proposed compression technique is in post-processing. The decoded image is a segmented quantized version of the original image. Such images have certain types of distortion, the most significant being false contouring in areas of gradual change in gray level. The visual quality of the decoded image could almost certainly be improved with some type of post-processing filtering operation. The post-processing operation should preserve unchanged high contrast edges, and smooth low contrast edges.

LIST OF REFERENCES

LIST OF REFERENCES

1. W. F. Schreiber, "Picture coding," *Proceedings of the IEEE*, vol. 55, pp. 320-330, March 1967.

2. M. Kunt, A. Ikonomopolous, and M. Kocher, "Second generation image-coding techniques," *Proceedings of the IEEE*, vol. 73, pp. 549-574, April 1985.

3. S. A. Rajala, H. A. Peterson, and E. J. Delp, "The use of mathematical morphology for encoding graytone images," *Proceedings of the 1988 IEEE International Conference on Circuits and Systems*, pp. 2807-2811, Helsinki, Finland, May 1988.

4. E. Hanna, D. Pearson , and J. Robinson, "Low data-rate coding using image primitives," *Proceedings of SPIE-Image Coding*, vol. 594, 1985.

5. R. M. Haralick and L. G. Shapiro, "Image segmentation techniques," *Computer Vision, Graphics, and Image Processing*, vol. 29, pp. 100-132, 1985.

6. S. A. Rajala, M. R. Civanlar, and W. M. Lee, "A second generation image coding technique using human visual system based segmentation," *Proceedings of the 1987 IEEE Conference on Acoustics, Speech, and Signal Processing*, pp. 1362-1365, April 1987.

7. M. Eden and M. Kocher, "On the performance of a contour coding algorithm in the context of image coding. Part II: Coding and contour graphs," *Signal Processing*, vol. 8, May 1985.

8. J. Serra, *Image Analysis and Mathematical Morphology,* Academic Press, New York, 1982.

9. P. A. Maragos, *A unified theory of translation-invariant systems with applications to morphological analysis and coding of images,* Ph.D. dissertation, School of Electrical Engineering Georgia Institute of Technology, Atlanta, Georgia, July 1985.

10. P. A. Maragos and R. W. Schafer, "Morphological skeleton representation and coding of binary images," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-34, pp. 1228-1244, October 1986.

11. A. N. Netravali and J. O. Limb, "Picture coding: A review," *Proceedings of the IEEE*, vol. 68, pp. 366-406, March 1980.

12. H. Armbruster and G. Arndt, "Broadband Communication And Its Realization With Broadband ISDN," *IEEE Communications Magazine*, vol. 25-11, pp. 8-19, November 1987.

13. A. Habibi, "Survey of adaptive image coding techniques," *IEEE Transactions on Communications*, vol. COM-25, pp. 1275-1284, November 1977.

14. W.-H. Chen and C. H. Smith, "Adaptive coding of monochrome and color images," *IEEE Transactions on Communications*, vol. COM-25, pp. 1285-1292, November 1977.

15. W. Zschunke, "DPCM picture coding with adaptive prediction," *IEEE Transactions on Communications*, vol. COM-25, pp. 1295-1302, November 1977.

16. A. Habibi, "An adaptive strategy for hybrid image coding," *IEEE Transactions on Communications*, vol. COM-29, pp. 1736-1740, December 1981.

17. N. F. Maxemchuck and J. A. Stuller, "An adaptive intraframe DPCM codec based upon nonstationary image model," *Bell System Technical Journal*, vol. 58, pp. 1395-1413, July-August 1979.

18. J. E. Abate, "Linear and adaptive delta modulation," *Proceedings of the IEEE* , vol. 55, pp. 298-308, March 1967.

19. P. J. Ready and D. J. Spencer, "Block adaptive DPCM transmission of images," *NTC'75 Conference Record*, vol. 2, pp. 22-10 to 22-17.

20. W. Kaminski and E. F. Brown, "An edge adaptive three-bit ten-level differential PCM coder for television," *IEEE Transactions on Communication Technology*, vol. COM-19, pp. 944-947, December 1971.

21. J. I. Gimlett, "Use of activity classes in adaptive transform image coding," *IEEE Transactions on Communications*, vol. COM-23, pp. 785-786, July, 1975.

22. C. Reader, "Intraframe and interframe adaptive transform coding," *Proceedings of SPIE*, vol. 66, pp. 108-118, August 1975.

23. E. J. Delp and O. R. Mitchell, "Image compression using block truncation coding," *IEEE Transactions on Communications*, vol. COM-27, pp. 1335-1342, September 1979.

24. P. A. Wintz, "Transform picture coding," *Proceedings of the IEEE*, vol. 60, pp. 809-820, July 1972.

25. A. N. Netravali and J. D. Robbins, "Motion-compensated television coding - Part I," *Bell System Technical Journal*, pp. 631-670, March 1979.

26. T. R. Natarajan and N. Ahmed, "On interframe transform coding," *IEEE Transactions on Communications*, vol. COM-25, pp. 1323-1329, November 1977.

27. J. A. Roese, W. K. Pratt, and G. S. Robinson, "Interframe cosine transform coding," *IEEE Transactions on Communications*, vol. COM-25, pp. 1329-1339, November 1977.

28. J. A. Stuller and B. Kurz, "Interframe sequential picture coding," *IEEE Transactions on Communications*, vol. COM-25, pp. 485-495, May 1977.

29. H. G. Musmann, P. Pirsch, and H.-J. Grallert, "Advances in picture coding," *Proceedings of the IEEE*, vol. 73, pp. 523-548, April 1985.

30. J. O. Limb, C. B. Rubinstein, and J. E. Thompson, "Digital coding of color video signals - A review," *IEEE Transactions on Communications*, vol. COM-25, pp. 1349-1385, November 1977.

31. W. F. Schreiber and R. R. Buckley, "A two-channel picture coding system: II-Adaptive companding and color coding," *IEEE Transactions on Communications*, vol. COM-29, pp. 1849-1858, December 1981.

32. N. S. Jayant and P. Noll, *Digital Coding of Waveforms: Principles and Applications to Speech and Video,* Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1984.

33. E. J. Delp, R. L. Kashyap, and O. R. Mitchell, "Image data compression using autoregressive time series models," *Pattern Recognition*, vol. 11, pp. 313-323, 1979.

34. J. K. Yan and D. J. Sakrison, "Encoding of images based on two-component source model ," *IEEE Transactions on Communications*, vol. COM-25, pp. 1315-1323 , November 1977.

35. A. Rosenfeld, *Image Modeling,* Academic Press, New York, 1981.

36. B. R. Hunt, "Nonstationary statistical image models (and their application to image data compression)," *Computer Vision, Graphics, and Image Processing*, vol. 12, pp. 173-186, February 1980.

37. J. O. Limb, "Distortion criteria of the human viewer," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-9, pp. 778-793, December 1979.

38. A. K. Jain, "Image data compression: A review," *Proceedings of the IEEE*, vol. 69, pp. 349-389, March 1981.

39. J. L. Mannos and D. J. Sakrison, "The effects of a visual fidelity criterion on the encoding of images," *IEEE Transactions on Information Theory*, vol. IT-20, pp. 525-536, July 1974.

40. D. J. Granrath, "The role of human visual models in image processing," *Proceedings of the IEEE*, vol. 69, pp. 552-561, May 1981.

41. D. J. Sakrison, "On the role of the observer and a distortion measure in image transmission," *IEEE Transactions on Communications*, vol. COM-25, pp. 1251-1267, November 1977.

42. Z. L. Budrikis, "Visual fidelity criterion and modelling," *Proceedings of the IEEE* , vol. 60, pp. 771-779, July 1972.

43. C. E. Shannon, "Coding theorem for a discrete source with a fidelity criterion," in *Information and Decision Processes* , ed. R. E. Machol, pp. 93-126, McGraw-Hill, New York, 1960.

44. D. Marr, *Vision,* W. H. Freeman and Company, New York, 1982.

45. T. N. Cornsweet, *Visual Perception,* Academic Press, New York, 1970.

46. S. S. Stevens, *Handbook of Experimental Psychology,* Wiley, New York, 1951.

47. J. J. DePalma and E. M. Lowry, "Sine-wave response of the human visual system. II. Sine-wave and square-wave contrast sensitivity," *Journal of the Optical Society of America*, vol. 52, pp. 328-335, March 1962.

48. F. L. van Nes, J. J. Koenderink, H. Nas, and M. A. Bouman, "Spatiotemporal modulation transfer in the human eye," *Journal of the Optical Society of America*, vol. 57, pp. 1082-1088, September 1967.

49. D. H. Kelly, "J stimulus patterns for visual research," *Journal of the Optical Society of America*, vol. 50, pp. 1115-1116, November 1960.

50. J. G. Robson, "Spatial and temporal contrast-sensitivity functions of the visual system," *Journal of the Optical Society of America*, vol. 56, pp. 1141-1142, August 1966.

51. D. H. Kelly, "Motion and vision. I. Stabilized images of stationary gratings," *Journal of the Optical Society of America*, vol. 69, pp. 1266-1274, September 1979.

52. D. H. Kelly, "Motion and vision. II. Stabilized spatio-temporal threshold surface," *Journal of the Optical Society of America*, vol. 69, pp. 1340-1349, October 1979.

53. A. J. Seyler and Z. L. Budrikis, "Detail perception after scene changes in television image presentations," *IEEE Transactions on Information Theory*, vol. IT-11, pp. 31-43, January 1965.

54. R. M. Gray, "Vector quantization," *IEEE Acoustics, Speech, and Signal Processing Magazine*, pp. 4-29, April 1984.

55. C. E. Shannon, "A mathematical theory of communication," *Bell System Technical Journal*, vol. 27, pp. 379-423, 623-656, 1948.

56. J. Y. Y. Huang and P. M. Schultheiss, "Block quantization of correlated gaussian random variables," *IEEE Transactions on Communication Systems*, vol. CS-11, pp. 289-296, September 1963.

57. J. B. O'Neal, Jr., "Entropy coding in speech and television differential PCM systems," *IEEE Transactions on Information Theory*, vol. IT-17, pp. 758-761, November 1971.

58. E. Cinlar, *Introduction to Stochastic Processes*, Prentice-Hall, Inc., Englewood Cliffs, New Jersey, 1975.

59. W. K. Pratt, *Image Transmission Techniques*, Academic Press, New York, 1979.

60. C. C. Cutler (Guest Editor), "Special Issue on Redundancy Reduction," *Proceedings of the IEEE*, vol. 55, March 1967.

61. T. S. Huang and O. J. Tretiak, *Picture Bandwidth Compression*, Gordon and Breach, New York, 1972.

62. A. Habibi (Guest Editor) , "Special Issue on Image Bandwidth Compression," *IEEE Transactions on Communications*, vol. COM-25, November 1977.

63. A. Habibi and A. N. Netravali (Guest Editors), "Special Issue on Picture Communication Systems," *IEEE Transactions on Communications*, vol. COM-29, December 1981.

64. C. C. Cutler, *Differential quantization of communication signals*, U.S.Patent 2,605,361, July 29, 1952.

65. F. deJager, "Delta modulation, a method of PCM transmission using a 7-unit code," *Philips Research Reports*, pp. 442-466, December 1952.

66. C. L. Song, J. Garodnick, and D. L. Schilling, "A variable step size robust delta-modulator," *IEEE Transactions on Communication Technology*, vol. COM-19, pp. 1033-1099, December 1971.

67. A. K. Jain, "A fast Karhunen-Loeve transform for finite discrete images," *Proceedings of the National Electronics Conference*, Chicago, IL, October 1974.

68. G. B. Anderson and T. S. Huang, "Piecewise Fourier transformation for picture bandwidth compression," *IEEE Transactions on Communication Technology*, vol. COM-19, pp. 133-140, April 1971.

69. S. C. Knauer, "Real-time video compression algorithm for Hadamard transform processing," *Proceedings of SPIE*, vol. 66, pp. 58-69, August 1975.

70. T. Fukinuki and M. Miyata, "Intraframe image coding by cascaded Hadamard transforms," *IEEE Transactions on Communications*, vol. COM-21, pp. 175-180, March 1973.

71. O. K. Ersoy and C. H. Chen, "Transform-coding of images with reduced complexity," *Computer Vision, Graphics, and Image Processing*, vol. 42, pp. 19-31.

72. R. M. Haralick, N. Griswold, and N. Kattiyabulwanich, "A fast two-dimensional Karhunen-Loeve transform," *Proceedings of SPIE*, vol. 66, pp. 144-159, August 1975.

73. L. Ehrman, "Analysis of some redundancy removal bandwidth compression techniques," *Proceedings of the IEEE*, vol. 55, pp. 278-287, March 1967.

74. L. D. Davisson, "Data compression using straight line interpolation," *IEEE Transactions on Information Theory*, vol. IT-14, pp. 390-394, May 1968.

75. D. Gabor and P. C. J. Hill, "Television band compression by contour interpolation," *Proceedings of the IEE*, vol. 108B, pp. 303-313, May 1961.

76. T. S. Huang, "Coding of two-tone images," *IEEE Transactions on Communications*, vol. COM-25, pp. 1406-1424, November 1977.

77. J. W. Modestino and V. Bhaskaran, "Robust two-dimensional tree encoding of images," *IEEE Transactions on Communications*, vol. COM-29, pp. 1786-1798, December 1981.

78. J. Capon, "A probabilistic model for run-length coding of pictures," *IRE Transactions on Information Theory*, vol. IT-5, pp. 157-163, December 1959.

79. S. W. Golomb, "Run length encodings," *IRE Transactions on Information Theory*, vol. IT-12, pp. 399-401, July 1966.

80. L. C. Wilkins and P. A. Wintz, "A contour tracing algorithm for data compression for two-dimensional data," Technical Report TR-EE 69-14, Purdue University School of Electrical Engineering, West Lafayette, Indiana, 1969.

81. W. F. Schreiber, C. F. Knapp, and N. D. Kay, "Synthetic highs, an experimental TV bandwidth reduction system," *Journal of Society of Motion Picture and Television Engineers*, vol. 68, pp. 525-537, August 1959.

82. W. F. Schreiber, T. S. Huang, and O. J. Tretiak, "Contour coding of images," in *Picture Bandwidth Compression*, ed. O. J. Tretiak, pp. 443-448, Gordon and Breach, New York, 1972.

83. S. G. Mallat, "A theory for multiresolutional signal decomposition: The wavelet representation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-11, pp. 674-693, July 1989.

84. P. J. Burt and E. H. Adelson, "The Laplacian pyramid as a compact image code," *IEEE Transactions on Communications*, vol. COM-31, pp. 532-540 , April 1983.

85. J. W. Woods and S. D. O'Neill, "Subband coding of images," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, pp. 1278-1288, October 1986.

86. D. E. Troxel *et al.*, "A two-channel picture coding system: I- Real-time implementation," *IEEE Transactions on Communications*, vol. COM-29, pp. 1841-1848, December 1981.

87. M.J. Biggar, O.J. Morris, and A.G.Constantinides, "Segmented-image coding: Performance comparison with the discrete cosine transform," *IEE Proceedings*, vol. 135, Pt. F, NO. 2, April 1988.

88. M. Kocher and M. Kunt, "A contour-texture approach to picture coding," *Proceedings of the 1982 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 436-440, Paris, France, May 1982.

89. D. N. Graham, "Image transmission by two-dimensional contour coding," *Proceedings of the IEEE*, vol. 55, pp. 336-346, March 1967.

90. D. H. Ballard and C. M. Brown, *Computer Vision,* Prentice-Hall, Englewood Cliffs, 1982.

91. T. Asano and N. Yokoya, "Image schema for low-level computer vision," *Pattern Recognition*, vol. 14, pp. 1-10, 1982.

92. C. Brice and C. Fennema, "Scene analysis using regions," *Artificial Intelligence*, vol. 1, pp. 205-226, 1970.

93. C. K. Chow and T. Kaneko, "Boundary detection of radiographic images by a thresholding method," in *Frontiers of Pattern Recognition*, ed. S. Watanabe, pp. 61-82, Academic Press, New York, 1972.

94. Y. Fukada, "Spatial clustering procedures for region analysis," *Pattern Recognition*, vol. 12, pp. 395-403, 1980.

95. H. A. Peterson, S. A. Rajala, and E. J. Delp, "Image segmentation using human visual system properties with applications in image compression," *Proceedings of the 1989 SPIE/SPSE Symposium on Electronic Imaging*, pp. 155-163, Los Angeles, California, January 1989 .

96. E. H. Adelson and J. R. Bergen, "Spatiotemporal energy models for the perception of motion," *Journal of the Optical Society of America*, vol. 2, pp. 284-299, February 1985.

97. A. N. Netravali and B. Prasada, "Adaptive quantization of picture signals using spatial masking," *Proceedings of the IEEE*, vol. 65, pp. 536-548, April 1977.

98. D. K. Sharma, "Design of absolutely optimal quantizers for a wide class of distortion measures," *IEEE Transactions on Information Theory*, vol. IT-24, pp. 693-702, November 1978.

99. H. G. Musmann, "Predictive image coding," in *Image Transmission Techniques*, ed. W. K. Pratt, Academic Press, New York, 1979.

100. M. I. Sezan, K. L. Yip, and S. J. Daly, "Uniform perceptual quantization: Applications to digital radiography," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-17, pp. 622-634, July/August 1979.

101. F. Kretz , "Subjectively optimal quantization of pictures," *IEEE Transactions on Communications*, vol. COM-23, pp. 1288-1292 , November 1975.

102. R. Schafer, "Design of adaptive and nonadaptive quantizers using subjective criteria," *Signal Processing*, vol. 5, pp. 333-345, July 1983.

103. J. O. Limb and C. B. Rubinstein, "On the design of quantizers for DPCM coders: A functional relationship between visibility, probability,and masking," *IEEE Transactions on Communications*, vol. COM-26, pp. 573-578, May 1978.

104. P. Pirsch, "Design of DPCM quantizers for video signals using subjective tests," *IEEE Transactions on Communications*, vol. COM-29, pp. 990-1000, July 1981.

105. A. N. Netravali, "On quantizers for DPCM coding of picture signals," *IEEE Transactions on Information Theory*, vol. IT-23, pp. 360-370, May 1977.

106. J. O. Limb, "Source-receiver encoding of television signals," *Proceedings of the IEEE*, vol. 55, pp. 364-380, March 1967.

107. W. Thoma, "Optimizing the DPCM for video signals using a model of the human visual system," *Proceedings of the IEEE International Zurich Seminar on Digital Communications*, pp. C3(1)-C3(7), March 1974.

108. D. K. Sharma and A. N. Netravali, "Design of quantizers for DPCM coding of picture signals," *IEEE Transactions on Communications*, vol. COM-25, pp. 1267-1274 , November 1977.

109. J. C. Candy and R. H. Bosworth, "Methods for designing differential quantizers based on subjective evaluations of edge busyness," *Bell System Technical Journal*, vol. 51, pp. 1495-1516, September 1972.

110. K. N. Ngan, K. S. Leong, and H. Singh, "Cosine transform coding incorporating human visual system model," *SPIE - Visual Communication & Image Processing*, vol. 707, pp. 165-171, 1986.

111. Norman B. Nill, "A visual model weighted cosine transform for image compression and quality assessment," *IEEE Transactions on Communication*, vol. COM-33, pp. 551-557, June 1985.

112. F. W. Mounts, A. N. Netravali, and B. Prasada, "Design of quantizers for real time Hadamard transform coding of pictures," *Bell System Technical Journal*, vol. 56, pp. 21-48, January 1977.

113. J. Max, "Quantizing for minimum distortion," *IRE Transactions on Information Theory*, vol. IT-6, pp. 7-12, March 1960.

114. G. Matheron, *Random Sets and Integral Geometry*, Wiley, New York, 1975.

115. H. Minkowski, "Volumen und Oberflache," *Math. Annalen.*, vol. 57, pp. 447-495, 1903.

116. H. Hadwiger, "Minkwoskishe Addition und Subtraction beliebiger Punktmengen und die Theoreme von Erhard Schmidt," *Math. Zeitschrift*, vol. 53, pp. 210-218, 1950.

117. H. Hadwiger, *Vorlesungen uber Inhalt, Oberflache und Isoperimetrie*, Springer Verlag, Berlin, 1957.

118. W. Kohler, *Gestalt Psychology*, Liveright Publ. Corp., New York, (1947)1970.

119. J. Serra, "Introduction to mathematical morphology," *Computer Vision, Graphics, and Image Processing*, vol. 35, pp. 285-303, September 1986.

120. R. M. Haralick, S. R. Sternberg, and Xinhua Zhuang, "Image analysis using mathematical morphology," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-9, pp. 523-550, July 1987.

121. P. A. Maragos and R. W. Schafer, "Morphological filters- Part I: Their set-theoretic analysis and relations to linear shift-invariant filters," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. ASSP-35, pp. 1153-1169, August 1987.

122. S. R. Sternberg, "Grayscale morphology," *Computer Vision, Graphics, and Image Processing*, vol. 35, pp. 333-355, September 1986.

123. G. R. Arce, N. C. Gallagher, and T. A. Nodes, "Median filters: Theory for one- and two-dimensional filters," in *Advances in Computer Vision and Image Processing*, vol. 2, ed. T. S. Huang, pp. 89-166, Greenwich, CT, JAI Press, 1986.

124. H. Blum, "A transformation for extracting new descriptors of shape," in *Models for the Perception of Speech and Visual Forms*, ed. W. Wathen-Dunn, M. I. T. Press, Cambridge, MA, 1967.

125. H. Blum, "Biological shape and visual sciences (Part I)," *Journal of Theoretical Biology*, vol. 38, pp. 205-287, 1973.

126. P. Elias, "Predictive coding - Part I," *IRE Transactions on Information Theory*, vol. IT-2, pp. 16-33, March 1955.

127. D. A. Huffman, "A method for the construction of minimum redundancy codes," *Proceedings of the IEEE*, vol. 40, pp. 1098-1101, September 1952.

APPENDICES

APPENDIX A.

Table A1. Number of times each test image set received each ranking in the experiment described in Chapter 2. There were ten test subjects, each of whom ranked each test image set twice, for a total of 20 rankings for each test image set. ("psf" refers to post-segmentation filtering).

image	appx. no. segments	coarse segmentation, no psf			medium segmentation, moderate psf			fine segmentation, extensive psf		
		rank 1	rank 2	rank 3	rank 1	rank 2	rank 3	rank 1	rank 2	rank 3
Airpl	2688	1	0	19	0	19	1	19	1	0
	2911	0	0	20	3	17	0	17	3	0
	2684	1	1	18	0	19	1	19	0	1
Eric	1916	4	4	12	4	10	6	12	6	2
	2301	1	0	19	5	14	1	14	6	0
	1434	0	0	20	6	14	0	14	6	0
Girl	1273	0	0	20	4	16	0	16	4	0
	1652	0	0	20	4	16	0	16	4	0
	1064	0	0	20	0	20	0	20	0	0
House	3504	1	7	12	1	12	7	18	0	2
	2774	1	2	17	11	7	2	8	11	1
	2834	1	2	17	5	14	1	14	4	2
	2314	2	1	17	3	16	1	15	3	2
Krista	685	0	5	15	15	3	2	5	12	3
	948	0	0	20	14	6	0	6	14	0
Natalie	714	0	0	20	2	18	0	18	2	0
	855	0	0	20	4	16	0	16	4	0
	1076	0	0	20	2	18	0	18	2	0

APPENDIX B.

Table B1. Number of times each test image set received each ranking in the experiment described in Chapter 3. There were eleven test subjects, each of whom viewed each test image set twice, for a total of 22 rankings for each test image set.

image	number of quant levels	HVS-based quantizer			Histogram quantizer			Uniform quantizer		
		rank 1	rank 2	rank 3	rank 1	rank 2	rank 3	rank 1	rank 2	rank 3
Eric	12	0	5	17	22	0	0	0	17	5
	18	0	8	14	21	0	1	1	14	7
Girl	20	1	8	13	12	7	3	9	7	6
	24	1	5	16	12	8	2	9	9	4
House	8	12	8	2	5	9	8	5	5	12
	12	4	9	9	13	5	4	5	8	9
Krista	10	9	10	3	1	7	14	12	5	5
	15	10	5	7	7	3	12	5	14	3
	20	10	12	0	0	3	19	12	7	3
Natalie	12	1	3	18	18	3	1	3	16	3
	20	1	2	19	14	8	0	7	12	3

APPENDIX C.

In this appendix we describe a technique proposed by Elias for coding a sparse binary image. Suppose a binary image is mostly 0's with only a few 1's. The rows of the $N \times N$ image are concatenated together to form a vector of N^2 gray level values, and all runs of consecutive 0's in this vector are found. The lengths of these runs, separated by a symbol (referred to as a "comma") to mark the end of a run (i.e., the presence of a 1), completely describe the original binary image. Elias has proposed coding these runlengths (viewed as decimal numbers) using an n-ary arithmetic system, and using an $n+1$'th symbol to represent a comma. For example, for $n=3$ the runlengths are represented in a ternary system. The comma requires an additional symbol, for a total of four symbols. These four symbols are represented using a two bit code. One possible choice to represent the four symbols is: 00=*comma*, 01=*0*, 10=*1*, 11=*2*.

Consider the following 40 bit binary sequence:

0000110000000000010001000000000100000000

The runlengths for this sequence are: 4, 0, 11, 3, 9, 8, and the ternary representations for these runlengths are: 11, 0, 102, 10, 100, 22. Finally, using the representation described above, the Elias code representation for the original binary sequence is:

1010 00 01 00 100111 00 1001 00 100101 00 1111

We have represented the original 40 bit sequence using 36 bits.

APPENDIX D.

Histograms for the images in Figures 2.3 and 3.2.

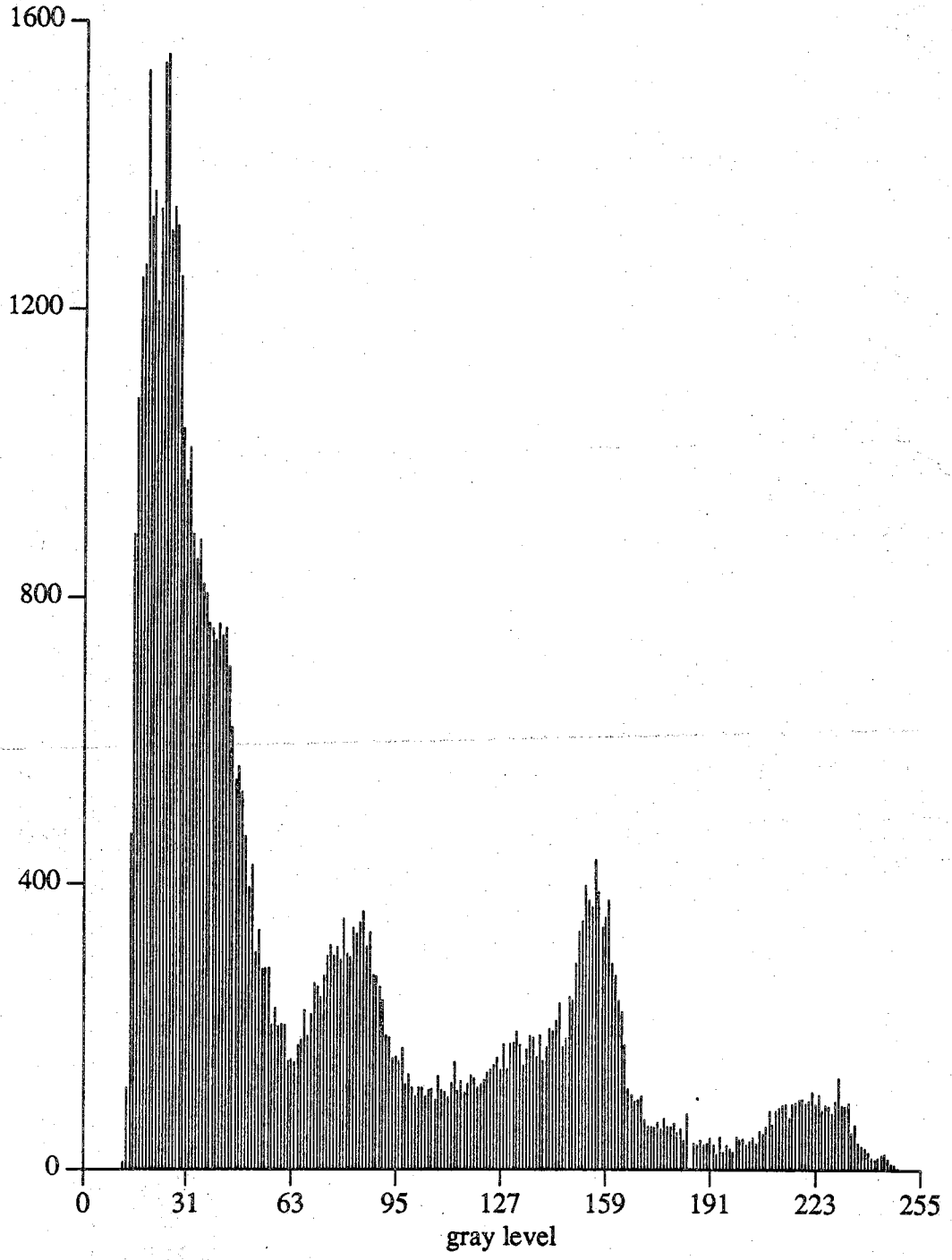


Figure D1. Histogram for the Girl image in Figure 2.3b.

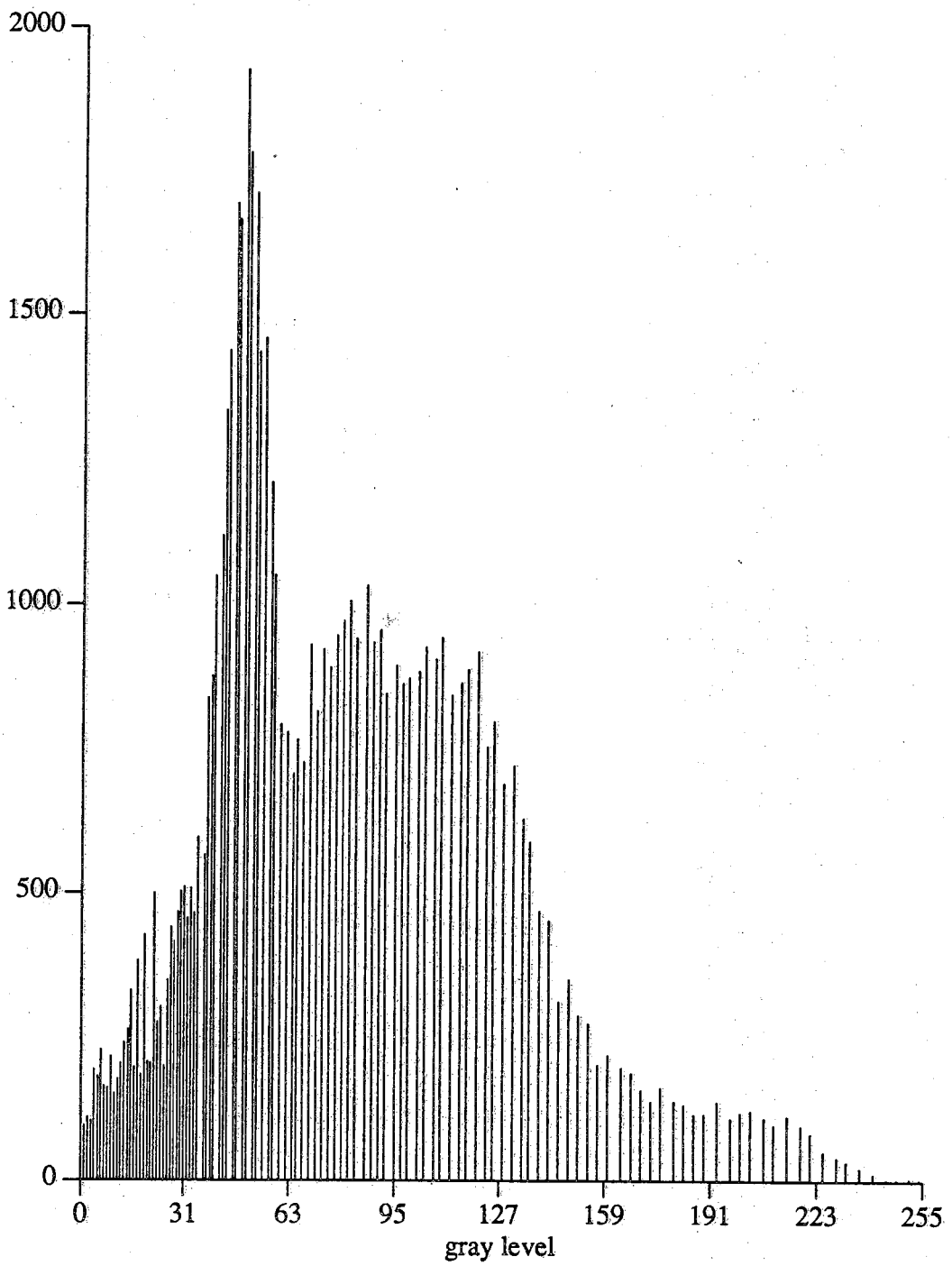


Figure D2. Histogram for the Eric image in Figure 2.3e.

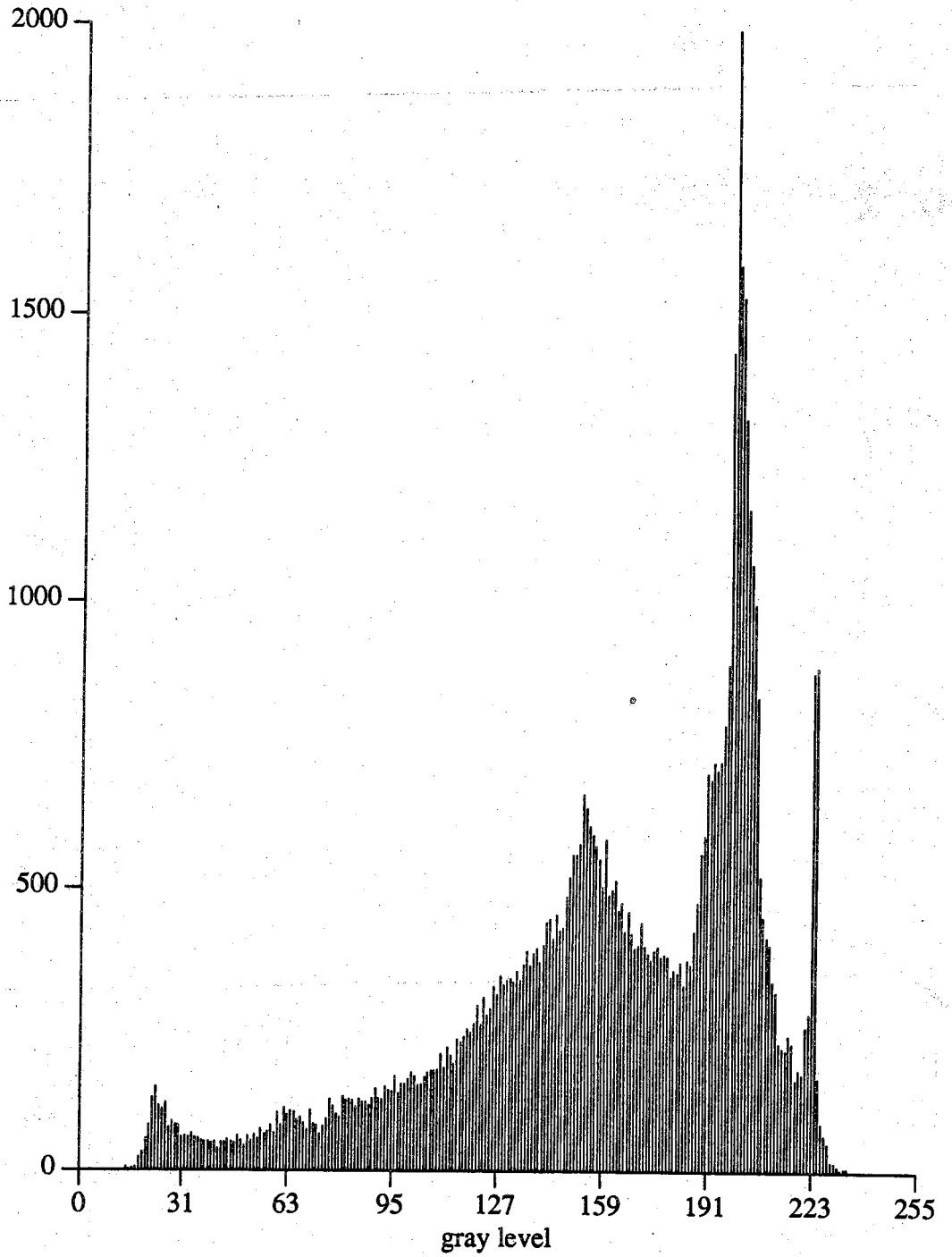


Figure D3. Histogram for the Natalie image in Figure 2.3a.

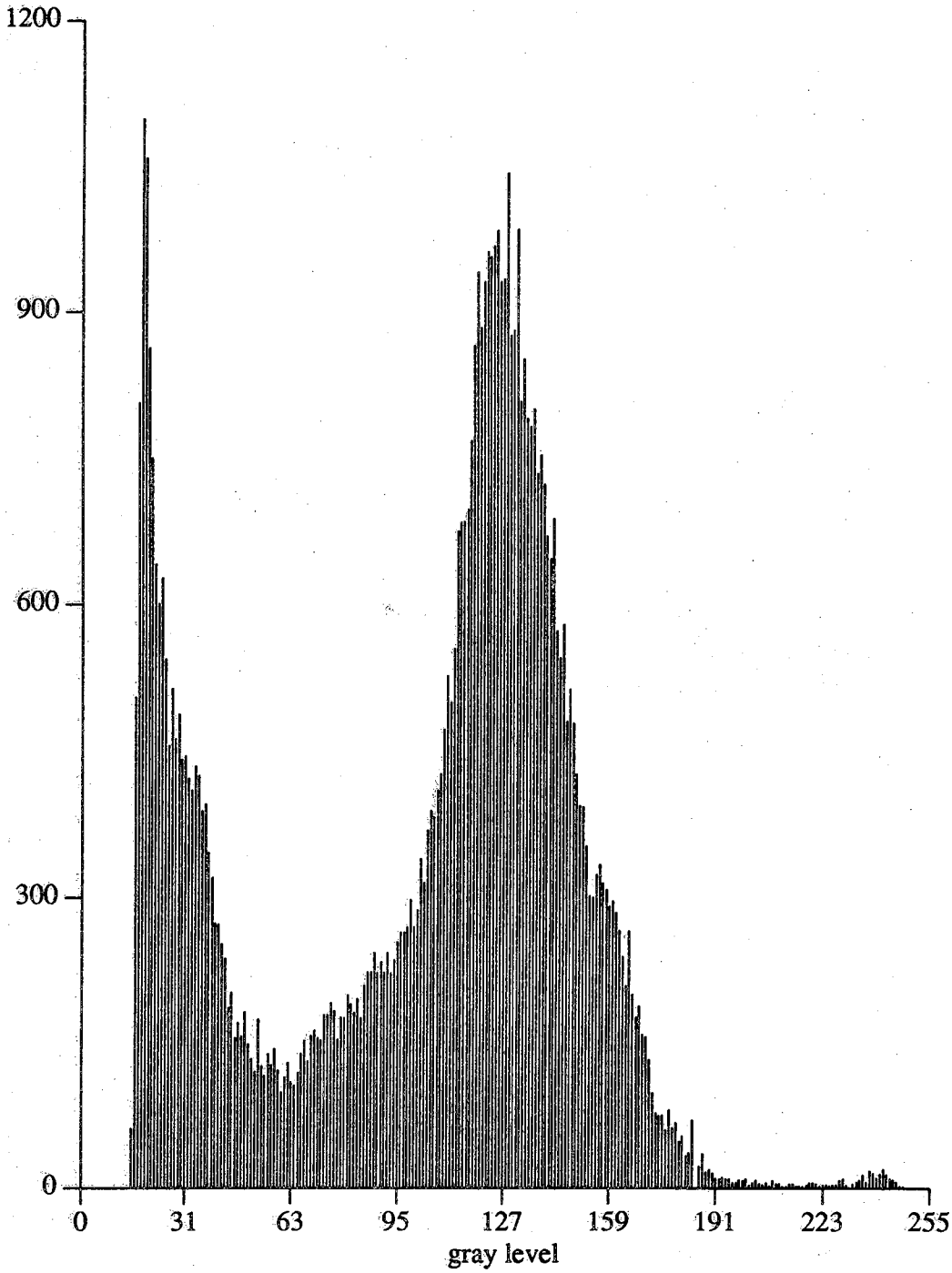


Figure D4. Histogram for the Krista image in Figure 2.3d.

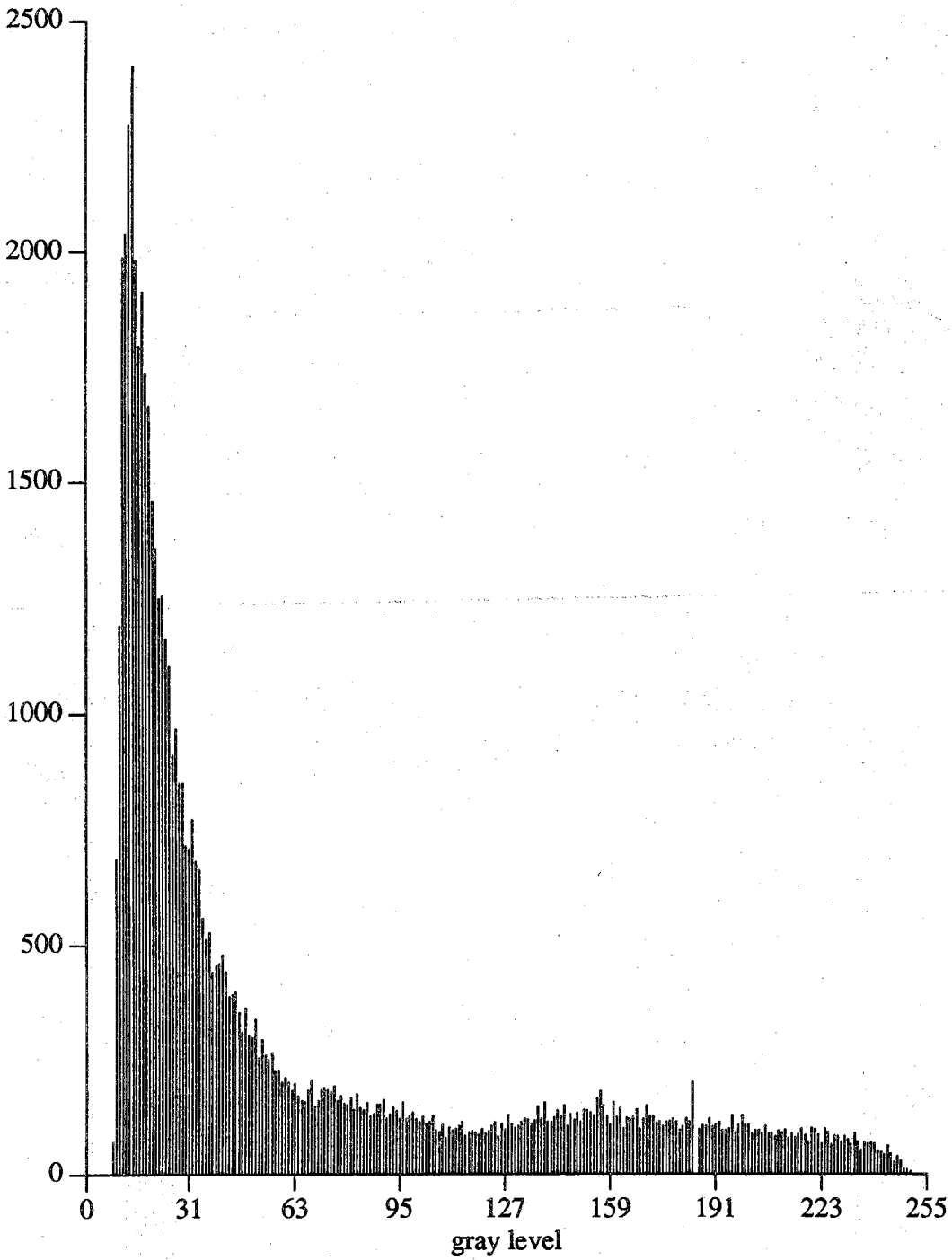


Figure D5. Histogram for the House image in Figure 2.3c.

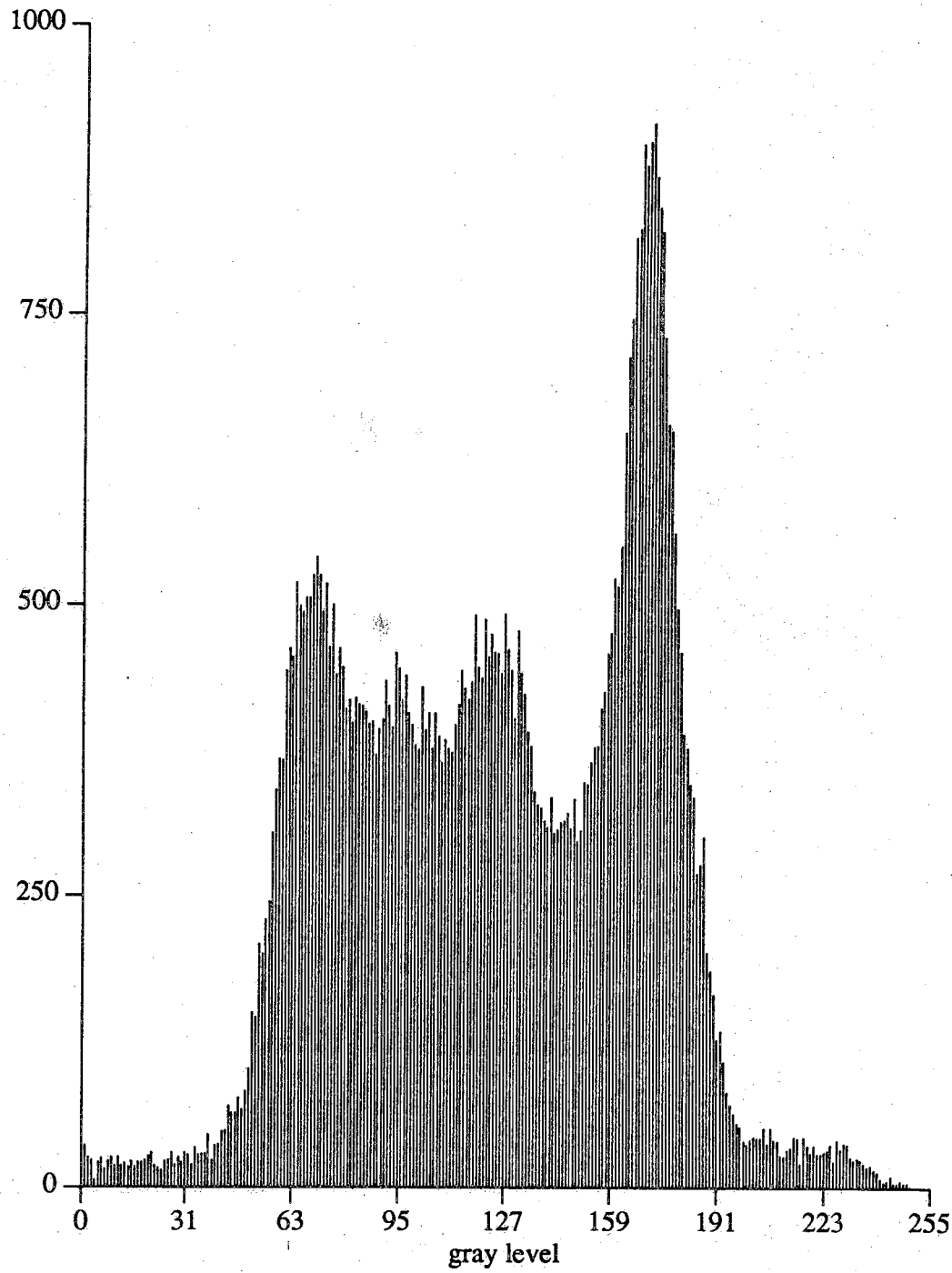


Figure D6. Histogram for the Airpl image in Figure 2.3f.

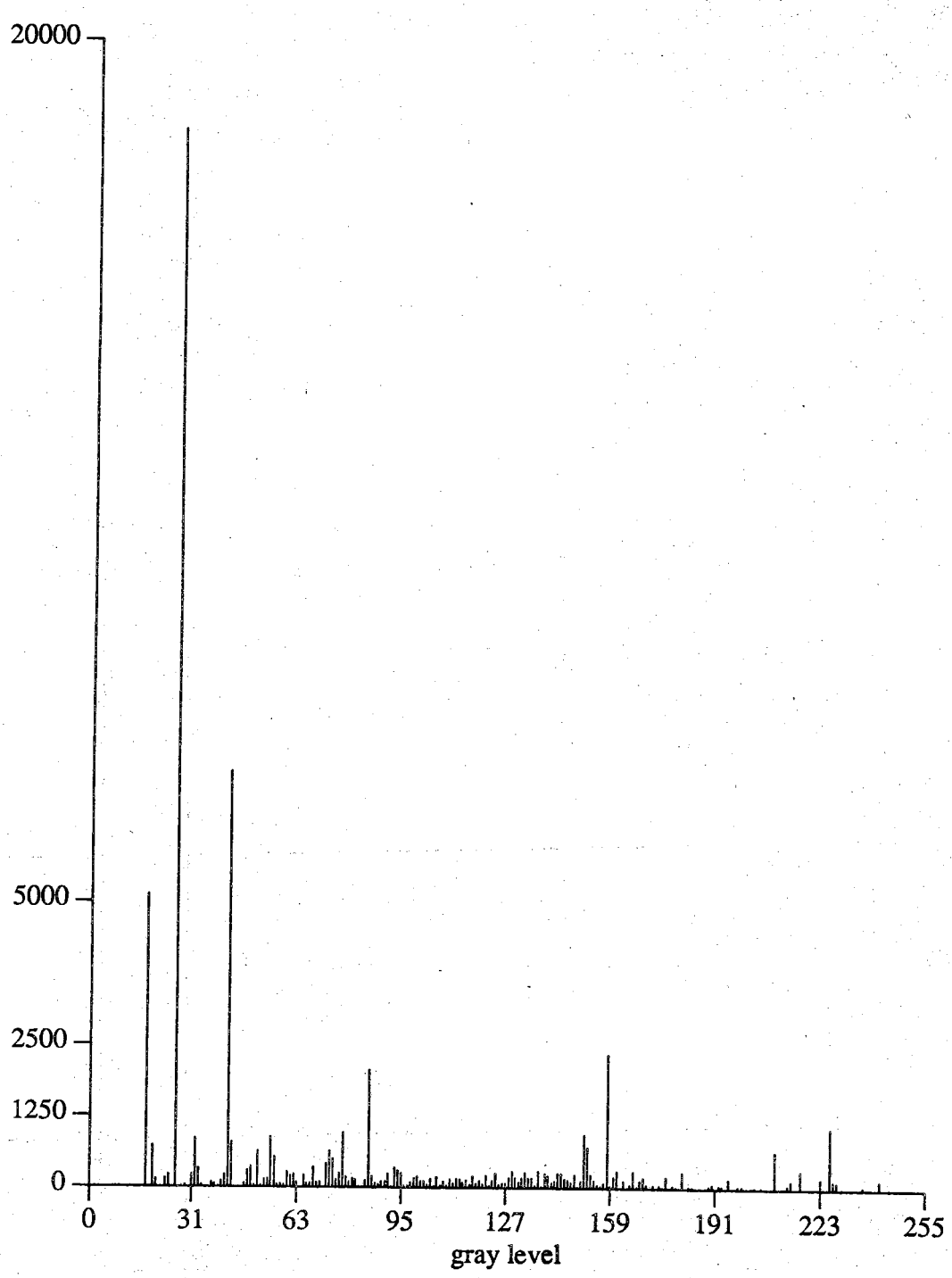


Figure D7. Histogram for the Girl image in Figure 3.2.

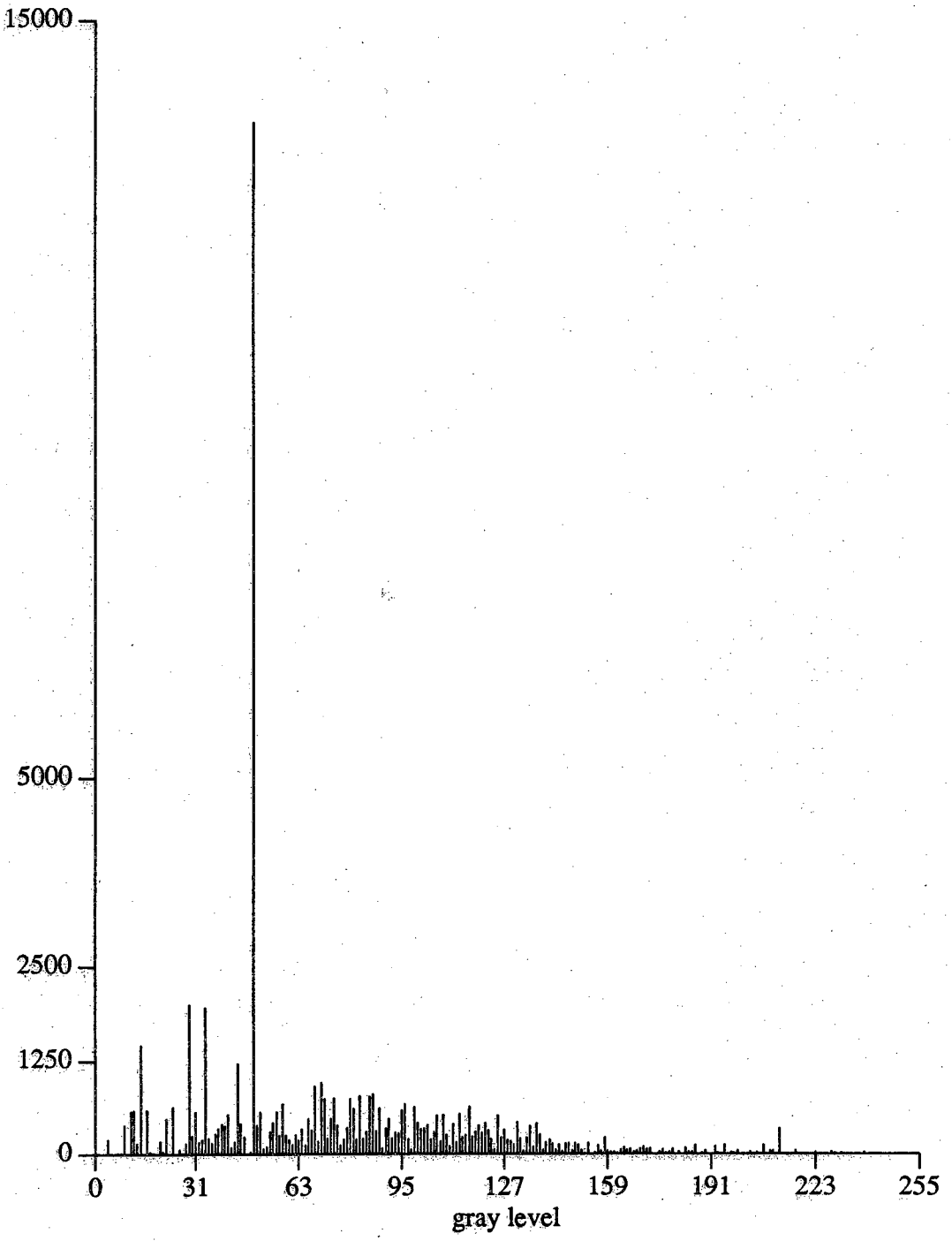


Figure D8. Histogram for the Eric image in Figure 3.2.

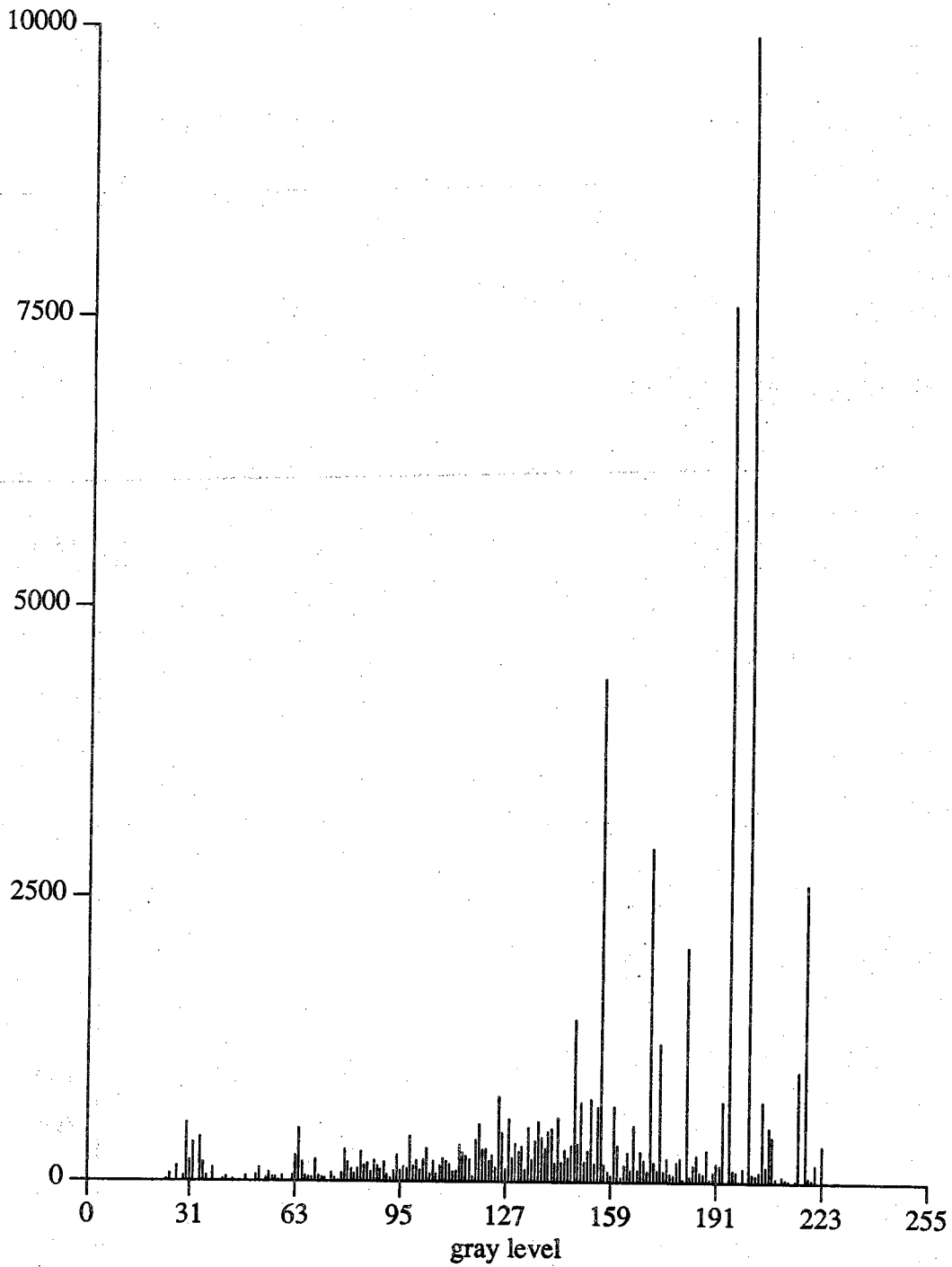


Figure D9. Histogram for the Natalie image in Figure 3.2.

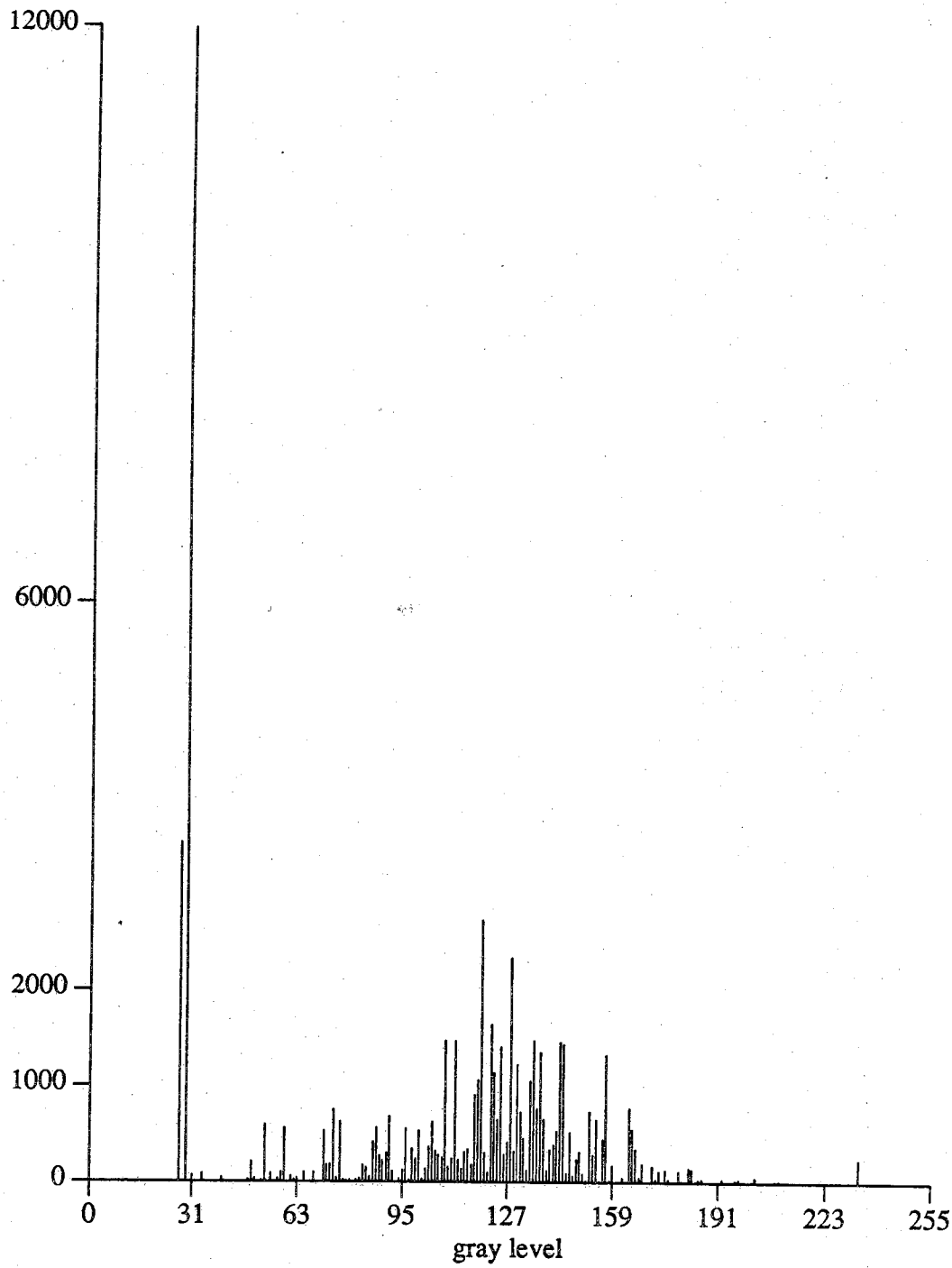


Figure D10. Histogram for the Krista image in Figure 3.2.

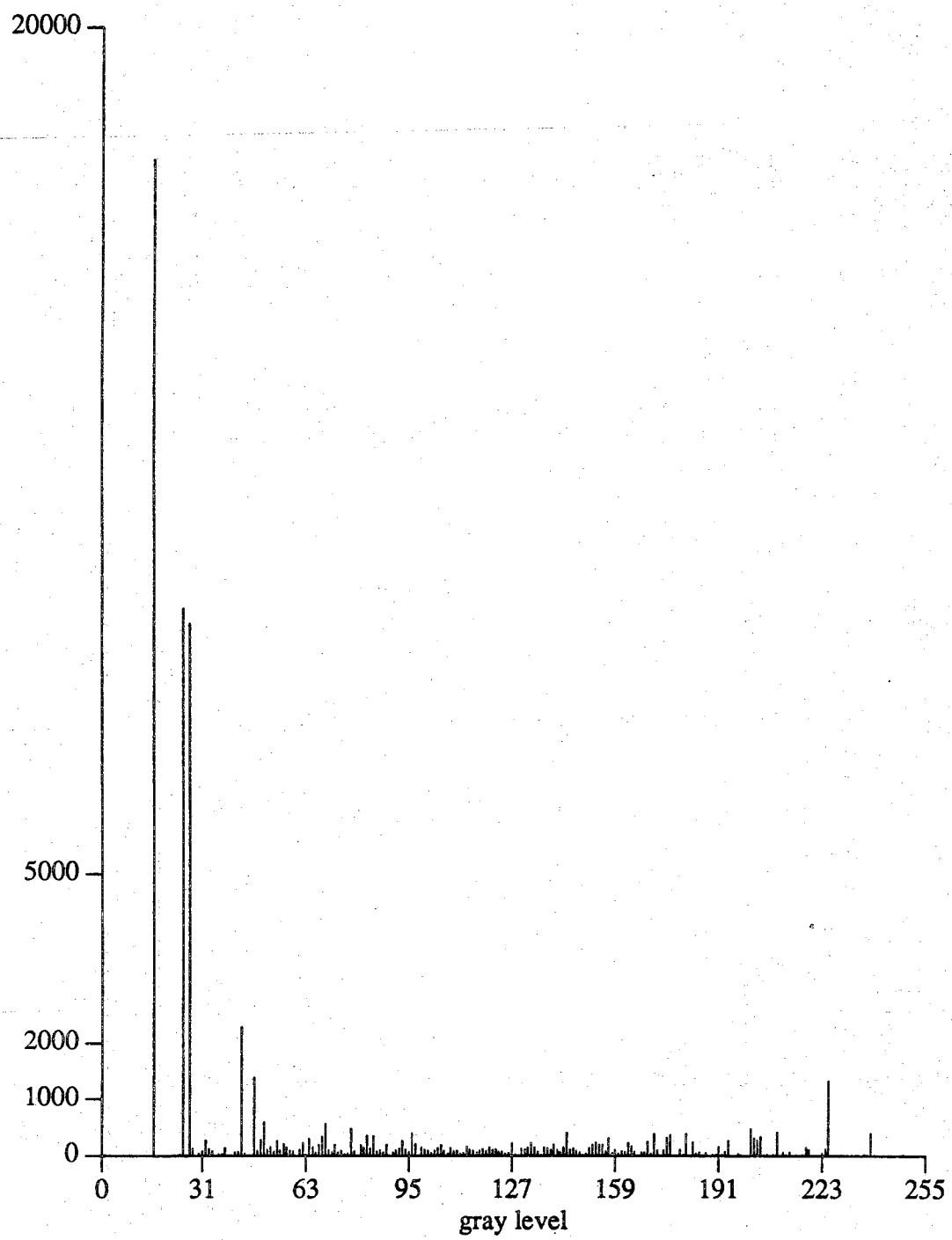


Figure D11. Histogram for the House image in Figure 3.2.

APPENDIX E.

For all the work in this thesis, the images were observed on a DeAnza CRT monitor, manufactured by Mitsubishi Electric, model C-3910. This monitor has 512 × 512 pixel resolution, with 256 possible gray levels. The monitor was calibrated so that luminance was linearly related to the gray level numeric value. In the first step of the calibration process a Minolta Chroma Meter (model CL-100) was used to measure the luminance of the screen for a variety of gray level values. A plot of the luminance versus gray level value before calibration is shown by the dotted line in Figure D1. A mapping was then defined to reassign the gray level values to achieve the desired linear relationship. This mapping is given in Table D1. The plot of luminance versus gray level value after the re-mapping is shown by the solid line in Figure D1. This plot shows that we have achieved the desired linear relationship.

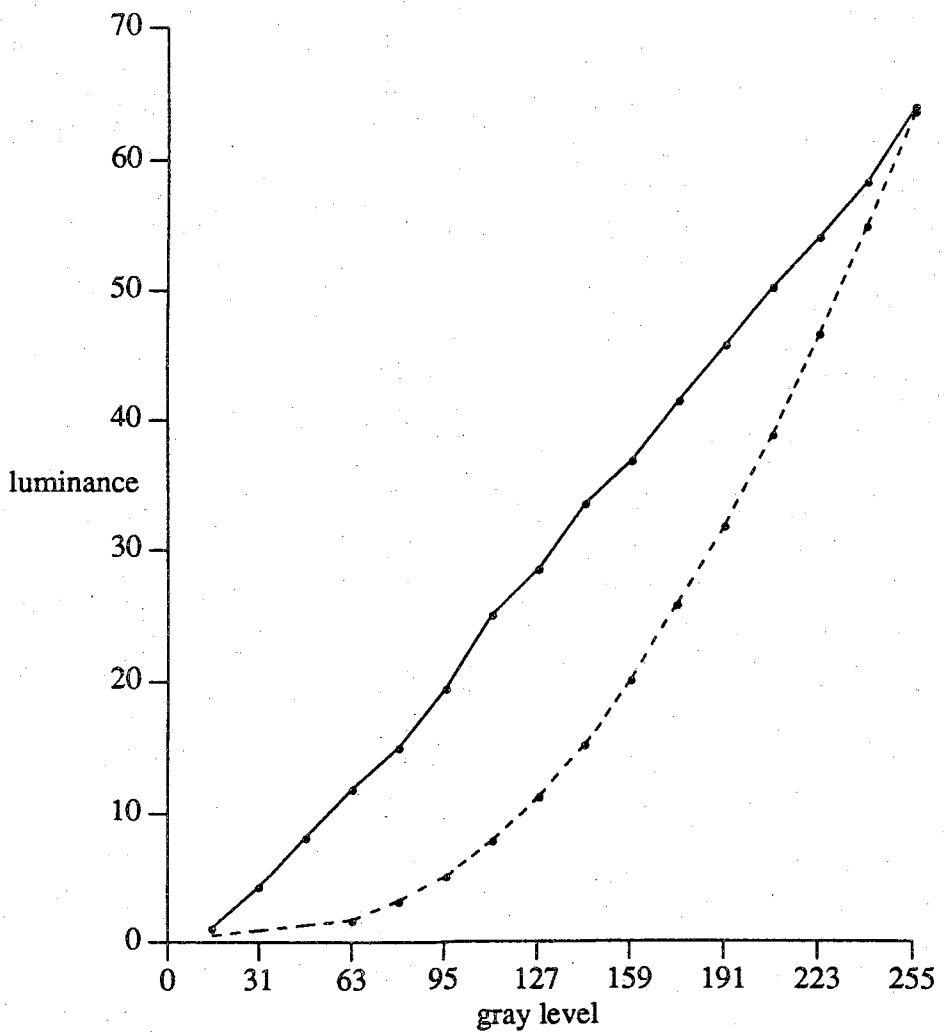


Figure E1. Luminance values before and after calibration (D0, VOC1).