A DIGITAL IMAGE PREPROCESSOR FOR OPTICAL CHARACTER RECOGNITION

by

WOLFRAM H.H.J. LUNSCHER
B.A.Sc., The University of British Columbia, 1980
B.Sc., The University of Toronto, 1974

A THESIS SUBMITTED IN PARTIAL FULFILMENT OF
THE REQUIREMENTS FOR THE DEGREE OF
MASTER OF APPLIED SCIENCE

in

THE FACULTY OF GRADUATE STUDIES
(Department of Electrical Engineering)

We accept this thesis as conforming
to the required standard

THE UNIVERSITY OF BRITISH COLUMBIA
October 1983

In presenting this thesis in partial fulfilment of the requirements for an advanced degree at the University of British Columbia, I agree that the Library shall make it freely available for reference and study. I further agree that permission for extensive copying of this thesis for scholarly purposes may be granted by the head of my department or by his or her representatives. It is understood that copying or publication of this thesis for financial gain shall not be allowed without my written permission.

Wolfram Lunscher

Department of Electrical Engineering

The University of British Columbia
2075 Wesbrook Place
Vancouver, Canada
V6T 1W5

October 17, 1983

ABSTRACT

Optical character recognition requires that data acquired from a camera device first be reconditioned into a suitable form. This thesis presents a design study of a preprocessor intended to accomplish this task under the constraints of real-time, autonomous operation as part of a reading machine for the blind.

The preprocessor contains three components: a filter to reduce the influence of noise and enhance the text seen; a binarization stage to identify the character and background pixels by a single bit; and a segmention system which isolates individual characters for delivery to the recognizer in proper causal order.

Filtering of the acquired video information is performed with the Laplacian of a Gaussian, $\nabla^2 g$, edge detection operator developed by D. Marr. This filter is shown to locate the character edges and to control the amount of detail seen optimally. Developed from models of the human visual system, this filter promises that the preprocessor could attain a human text resolution capability. Binarization is also reduced to a simple thresholding of the filter's output. To achieve optimal enhancement of print structures of known dimensions a filter design strategy is presented incorporating two periodic edge models. Since the filter must be digitized, the design method is further extended to include an analysis of the effects of

sampling the continuous filter and quantizing its coefficients for a direct-form finite impulse response implementation.

To validate the claim that this filter's performance is superior to other edge detection methods, a chapter is devoted to quantitative evaluation of $\nabla^2 g$ filtered test images. The results are compared to others published and found superior, as well as remarkably noise immune.

Segmentation of the binarized text is accomplished with a technique adapted from the binary-image description method of C.T. Zahn. The text images are reduced to a hybrid chain-code-and-coordinate description of all internal and external boundaries concurrent with the incoming raster-acquired data. No more than two image scan lines need be stored. All closed borders within the text are detected immediately as they occur by monitoring the local changes in the image's Euler number during a scan, and verifying global closure by following pointers through a linked-list data structure. A simulation of an implementation of the segmentation system, operating on fifty $\nabla^2 g$ filtered test images, indicated that the real-time performance objective can be achieved through a combined serial and parallel architecture.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ACKNOWLEDGEMENTS

I wish to extend my thanks to Dr. Michael P. Beddoes for offering the opportunity to explore this project, and for much valued guidance.

The author acknowledges financial support from a research assistantship 1980-82 awarded by the National Research Council of Canada, grant number 67-3290. A University of British Columbia fellowship 1982-83 is gratefully appreciated.

Special thanks go out to my lovely fiancée Cindy K.Y. Chan for her unfaultering emotional support throughout this degree program, and without whose frequent assistance and skill at the keyboard, preparation of this thesis would have been truly tedious.

I. INTRODUCTION

This thesis presents a design study preliminary to the development of a digital image preprocessor for optical character recognition. As a design study, the objective of this thesis is to examine a number of approaches to the development of a preprocessor, and then endorse a particular configuration. The operational parameters incorporated in this configuration are then studied to facilitate the final assembly and adjustment of the preprocessor. No attempt is made to address the problem of optical character recognition (OCR). The dominant reason for this is that this system was envisioned to complement an existing OCR system developed by Dr. M.P. Beddoes of the Department of Electrical Engineering at the University of British Columbia as part of a reading machine for the blind. However, the preprocessor design is kept as general as possible so that it may serve as an input to most other OCR systems.

The purpose of an OCR device is to accept a description of a machine-printed or handwritten character as input and then make a decision as to the identity of the character. The purpose of an OCR preprocessor is to accept the raw video information from an imaging device as input and then transform it into a form the OCR device can accept. Among the tasks performed by the preprocessor are noise removal (filtering), identification of character and background pixels (binarization), and "packaging" of the characters into individual units (segmentation). Not all OCR preprocessing systems perform all of these tasks, but the system

proposed in this thesis does. Figure 1.1 illustrates these operations in the order they will be performed.

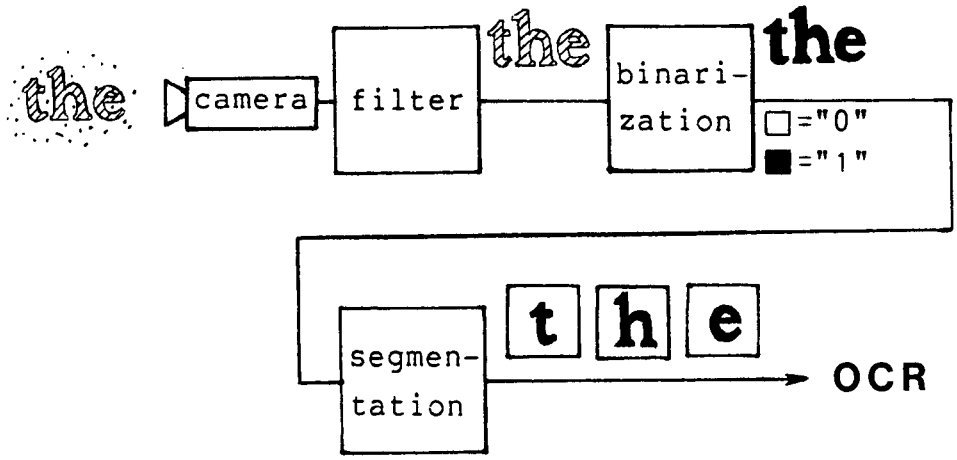


Figure 1.1 Preprocessor block structure

Since the first OCR system was commercially marketed in 1954 by the Intelligent Machines Research Corporation, the literature has come to abound with material related to OCR. Comprehensive surveys of the early machines and the problems they encountered were published by the British Computer Society [1], Auerbach [2], and Harmon [3]. More recently Ullmann [4], [5], [6], has presented several surveys of the field spanning the developments of the past decade. The current state of the art is reviewed by Schürmann [7]. These surveys illustrate the diversity that exists among OCR systems. Not only do many systems choose to organize the preprocessor differently from Figure 1.1, but some systems choose to omit one or more of these stages. Indeed, purely optical systems, utilizing the principles of laser holography, may do away with the entire system altogether.

This diversity in designs is largely a consequence of the diversity of OCR applications. These range from magnetic ink

character readers, employed to identify bank cheques, for which binarization and segmentation is trivial, to hand-printed-document readers for which preprocessing and particularly recognition is very difficult. Most of these systems require the input material to be presented in a suitable and sometimes rigidly fixed format. This includes characters hand-printed at fixed locations on the document, or machine-printed characters organized into well defined lines without the interspersion of non-text material. However, modern research is making progress towards relaxing the restrictions on the input to permit variable text formats, and random placement of illustrations (Wong et al. [8]). This added sophistication requires the addition of more components to the preprocessing system of Figure 1.1 to locate the text regions and to direct their acquisition.

The system developed here is targeted to serve as a component of a reading machine for the blind. This imposes certain constraints on its operation. Blindness is frequently accompanied by various motor disabilities. This demands the system's operation to require little or no manual dexterity. In order to produce a natural reading pace, real-time operation is required. Lastly, since the complete system is envisioned to be operated autonomously by a single user, it must be robust. Deviations in print quality, or minor operator errors must not generate a situation requiring intervention by a second party. Of course, the preprocessing system alone cannot be responsible for fullfilling all these constraints; these are issues that must be addressed by the OCR system as a whole. However, to be acceptable, the preprocessor must be compatible with these

constraints.

The image acquisiton device, or camera, can take on many forms. These include flying sport scanners, vidicons, laser scanners, and charge-coupled device (CCD) arrays. The system currently in use by Dr. Beddoes utilizes a 64 element linear CCD array. This device is readily amenable to handheld or simple machine-directed operation. Therefore, the preprocessor was designed to accomodate a linear-array-type acquisition device. Vidicon and other two-dimensional acquisition systems are also suitable. These permit the camera to remain stationary while the document is scanned electronically, requiring less dexterity and training on the part of the blind user. The primary constraints on the camera are pixel resolution, and field of view. Ullmann [5] notes that the optimal pixel size for most print styles covers 130μm, though Schürmann [7] claims 50μm is necessary for book print. The field of view must, of course, at least include the largest character expected.

The noise filtering and binarization stages are frequently arranged in a different order from Figure 1.1, and sometimes combined into a single operation. Binarization reduces the original pixel data representation to a single bit with character pixels receiving value "1" and background the value "0". The reasons behind this operation are largely economic. Binarized images require less storage space, and allow both segmentation and recognition to be performed with simple boolean operations.

Binarization always involves some form of thresholding operation. The success of the preprocessor correctly identifying

the character pixels depends on the proper setting of this threshold. In the majority of systems reviewed the threshold is applied directly to the input data without an intervening filter. The simplest such operation is to pick a fixed threshold intensity level and apply it throughout the image. Ullmann [4] notes that this is not a very useful operation since background and character intensity levels vary continually making it impossible to select a satisfactory level for all circumstances. An alternative approach is to adopt a floating threshold level adjusted according to local grey-level measurements. One of the most sophisticated such methods was developed by Bartz [9] for the IBM 1975 optical page reader. The threshold was not only influenced by local grey-level averaging, but also by the character line widths seen. Furthermore, a noise analysis revealed that typewritten and machine printed documents differed substantially in their noise content. Therefore, two different, user selectable, thresholding algorithms were applied. This, however, is inconsistent with our autonomous operation constraint. Another, less common, threshold selection procedure requires analysis of the document's grey-level histogram. This requires selection of an optimal partition to separate the distribution modes identified with character and background pixels. Such a method was outlined by Otsu [10], and applied successfully by Tou et al. [11]. However, the method was not considered suitable here because the grey-level statistics are accumulated by prescanning the entire document causing the method to be insufficiently sensitive to local contrast variation.

Since all of the above binarization techniques operate

directly on the input image without an intervening noise filter, a certain amount of extraneous detail invariably receives an inappropriate label. This may take the form of stray dark pixels in the background, light holes within the character regions, or rough character outlines. Removal of this noise is considered sufficiently important that almost every developer of an OCR system employing this form of binarization has included a noise cleaning stage. A general study of this procedure was published by Rosenfeld and Park [12].

In view of the unsatisfactory performance of most binarization procedures, it was felt that an alternative approach, founded on the principles of edge detection, was justified. Edge detection procedures have the advantage that their primary objective is the detection of local intensity variations while maximizing noise rejection. The intensity variations of interest are, of course, those caused by the presence of printed characters. Use of edge detection in OCR preprocessing is uncommon, but not new. The flying spot scanner used by Greanias et al. [13] sensed the local discontinuity at character edges to direct a contour trace, and the BCS [1] discuss another flying spot scanner control method where the signal difference between a focused and defocused beam (an approximation to a Laplacian) located character edges. Recent edge detector research has developed a class of edge filters which simulates the operation of the lower levels of the human visual system. Such a filter holds a particular attraction because the range of detail resolved and noise rejection achieved would be similar to that of a human reader. Therefore, any

document that a human could read, a machine incorporating such an edge filter should be able to read also. Even though edge detection was not developed for image binarization, it was observed that this new filter class could readily be modified to do this. Furthermore, this modification simply involves the application of a fixed threshold. This new edge filter class is called the optimal edge detection filters, and forms the subject of the next two chapters.

Chapter 2 introduces these optimal edge filters and defines the nature of their optimality. The rest of the chapter will address the design of the filter to render it suitable for any given image processing task. It will be seen that optimal filter resolution depends on at least one parameter. Since the filter must be capable of resolving all character detail while at the same time rejecting unwanted noise, considerable attention is given to the resolution response and noise sensitivity of the filter under variations of this parameter. Also, since the filter is initially specified as a continuous two-dimensional point spread function, it must be digitized before installation into the preprocessor. Digitization takes the form of spatial sampling and coefficient quantization. These operations are therefore also examined at length to determine the necessary sample spacing and coefficient word size.

Chapter 3 will attempt to substantiate the claims made in Chapter 2 that the performance of an optimal filter will exceed that of all other edge filters in common use. To achieve this objective, a purely quantitative edge detector evaluation method will be applied to an optimal edge filter. When compared

to published evaluation scores, the optimal filter's superior performance, and hence suitability for the preprocessor, is clearly seen.

The binarized image serves as the input to the segmentation stage of the preprocessor. Segmentation is viewed as necessary because it separates the individual characters from the remainder of the image for analysis and recognition. In this way, the recognition system can be reasonably certain that the data it has received represents a single character. However, not all researchers agree on the need for segmentation. Clayden, Clowes and Parks [14], maintain that segmentation is neither necessary nor desirable since it requires a certain degree of position and orientation control over the input. Instead, they applied a mask matching recognition technique directly to the running text without prior segmentation. While it is true that some positional control is necessary, it is also true that this necessity is largely a response to the need for the camera to consistently follow a given line of text. They also claim that correct segmentation may not always be possible. However, direct recognition did not seem to offer a useful alternative since their own results show recognition accuracy to deteriorate as character spacing decreases, a situation in which most segmentation schemes also break down. In an attempt to avoid the difficulties posed by variable spacing and touching characters, the IBM 1275 reader [15] places the binarized data into a series of shift registers which in turn are connected to a template correlation-type recognizer. The shift registers then move the characters to every possible vertical and horizontal position

within the correlator as though mounted on a rotating drum. Ullmann [5], [6] notes two shortcomings with this approach. It has difficulty with certain types of close characters such as "rn" which can be confused with "m". Also, this method is slow and costly to implement.

Chapter 4 will present a review of a number of segmentation methods and then describe the method endorsed for this system. Segmentation can be both a complex and time-consuming process. For this reason, the real-time constraint will have its greatest impact on this system component. In an attempt to estimate both the hardware requirements and expected processing delay times, a software simulation is performed on the segmentation system with fifty binarized images serving as input. The problem of separating touching characters is, however, addressed only lightly. This is seen as a complex problem, but suggestions for further work in this area are presented.

II. Optimal Edge Detector Design

2.1 Introduction

Recently a new approach to the problem of edge detection has appeared. This approach involves an image filter whose response was derived to optimize the enhancement of edges in a clearly defined way. Two such filters have been published. The first, developed by D. Marr and E. Hildreth [16], shows, through an heuristic argument, that the local response and range of spatial variation can optimally be minimized through application of a Gaussian filter, followed by a Laplacian operation to facilitate edge detection. The second, first published by Dickey and Shanmugam [17], is more rigorous in its definition of an edge and criteria for optimality. By defining an edge as a step discontinuity between adjacent regions of differing, but individually uniform, intensities, they show that the ideal bandlimited filter to optimally localize its response about the edge is given by a prolate spheroidal wave function. It will later be shown that the Marr filter is essentially an approximation to the Dickey filter; its simpler form also makes it more amenable to application and study.

The primary difference between these operators and the many others published is that they were developed from a narrow set of assumptions describing the response desired, which then constrained the filter to a form necessary to provide this response. Most edge operators, however, begin by making certain observations about the properties of edges in the input image and

then seek to isolate pixels with these properties from the background. A large class of edge operators start with the assumption that edges are signified by maxima in the gradient of the input image. This idea stimulated the development of the simple gradient operator which largely owes its ancestry to Roberts [18]. However, owing to the well-known noise enhancement properties of the pure gradient, it soon became apparent that some sort of smoothing operation must be included to reduce spurious edge detail. Rosenfeld and Thurston [19] used the difference between the average intensities in adjacent square regions, whose spatial dimensions were powers of two, to estimate the gradient at a pixel between them. The final size and orientation of the regions was determined dynamically according to the greatest absolute difference. Macleod [20] avoids the wideband response and the resultant "ringing" characteristic of a square region average by differencing image areas smoothed by superimposed displaced exponentials windowed by a Gaussian. For both edge operators smoothing large areas of the input image reduces the amount of image detail seen in the output image. This reduces noise detail, but little control is provided over the exact amount of true image detail that can be resolved.

A different approach from the gradient methods is to model the ideal edge as an intensity step, then formulate a series of templates that will produce a maximum response when centered on an edge. Perhaps the most sophisticated of these was developed by Hueckel [21], [22] where up to nine templates, representing a set of orthogonal basis functions over a circular region, are applied. After combining the response of the templates into a measure of the Hilbert distance from the image sample to the

ideal edge, a decision as to the presence of an edge is made. Like the gradient methods, the template approach also suffers in the presence of noise or non-ideal blurred edges. The problem of choosing the template size to prevent undesirable smoothing of edge detail also remains. However the greatest problem of both the gradient and template methods is seen in the results; the edges are generally disconnected and irregular regardless of whether generated by objects or noise. To combat this shortcoming, most authors provide some form of relaxation or line fitting technique to connect the edge segments into closed regions. The optimal edge detector, it will be shown, does not have this shortcoming. It naturally encloses image regions in unbroken edge boundaries.

Another objection is that the spatial frequency response is seldom considered in edge detector development. One exception is when the detector is to act as a matched filter for a particular edge or line feature, e.g. Halé [23]. Fourier transforms of some of these operators, particularly those involving local averaging over discontinuous fields like the Rosenfeld - Thurston, reveal high frequency passbands extending well beyond the principal passband. As a consequence, high frequency noise is not entirely suppressed. Also, additional features not present in the input image, such as the "ringing" of certain edges, appear. To remove these unwanted features careful attention is given to the selection of a threshold level to be applied against the resultant operator output.

In contrast, the optimal edge detectors were designed primarily in the frequency domain. The Marr and Hildreth filter

evolved from a model of human vision. The lowest levels of mammal vision were observed to include bandpass spatial frequency channels to filter image detail at various degrees of resolution [24]. After Wilson and Giese [25] showed that the frequency response of these bandpass channels can be approximated by a difference of two Gaussian distributions, Marr and Poggio [26] showed that each channel could detect edges through the zero crossings it produced. Further considerations as to what constitutes the best form of an edge filter lead Marr and Hildreth to the Laplacian of a Gaussian which they show to be the limiting condition of the difference of two Gaussians as their variances converge. The Dickey and Shanmugam filter, on the other hand, was developed in a more direct, analytical manner, but again involved serious consideration of the form of the frequency response. The fact that a prolate spheroidal wave function optimizes the edge filter results in part from strictly bandlimiting the filter to control noise at the outset.

Another distinguishing feature of optimal edge filters is that the form of their response is not fixed. Instead it is controlled by the standard deviation of the Gaussian in the case of the Marr and Hildreth filter, and by the bandwidth and the resolution-bandwidth product in the case of the Dickey and Shanmugam filter. This differs from many other edge detection filters which are specified as a mask of fixed form. The two optimal filters will be combined into one with the form of the Marr-Hildreth filter through choice of a common bandwidth and adopting an asymptotic approximation to the Dickey-Shanmugam filter. This will leave only one parameter, a standard deviation, to determine the form of the filter. The choice of

the appropriate standard deviation and the resultant consequences for the filter's accuracy constitute the principal design problem addressed here. Neither Marr and Hildreth nor Dickey and Shanmugam provide much guidance in this matter. Marr and Hildreth claim that parallel edges could not be resolved accurately when the filter's central wavelength exceeds twice the edge spacing. The Dickey and Shanmugam filters, on the other hand, involved specification of a resolution interval within which it is unlikely another edge feature will be found.

To improve on this state of affairs four stages of filter design will be considered. The first two will involve selection of an appropriate filter standard deviation to meet specified edge resolution requirements according to the structure of the input image and accuracy in the presence of noise. This will be done by examining the response to two periodic edge models blurred with a Gaussian distribution. Later, the positional accuracy of an isolated zero crossing in the presence of additive Gaussian noise will be considered. The result will be a set of design rules outlining the maximum filter standard deviation necessary to resolve an image which, in part, resembles one of the edge models. The expected accuracy to be achieved with that standard deviation is then presented, as well as the minimum tolerable signal to noise ratio in the input image. The remaining two design stages will address the problems of digital implementation: sampling the continuous filter; and quantizing the resultant coefficients. It will be seen, however, that the filter is very robust under both these operations allowing a comparatively coarse sample spacing and a quantization word size to match most image acquisition systems.

This design methodology should provide a comprehensive guide for the selection and implementation of a filter designed at the outset to be optimal at the task of resolving edge features. First we will review how the form of the optimal filter can be found.

2.2 The Case for Optimality

Marr and Hildreth, and Dickey and Shanmugam have independent arguments to substantiate their respective claim of having found an optimal filter. Each is rooted in a different cost function. That used by Marr and Hildreth measures the localization of the filter's influence in both the spatial and frequency domains. Dickey and Shanmugam, on the other hand, adopted the proportion of the filter's energy within a specified resolution interval about an edge feature. We will now examine each of these approaches in turn and then combine the results into a single near-optimal filter.

2.2.1 Marr Optimality

Marr and Hildreth avoided committing themselves to a rigorous mathematical model of an edge preferring instead to draw attention to the variety of natural phenomena giving rise to intensity changes in the visual world. These include illumination changes such as shadows, the orientation of visible surfaces, and changes in surface reflectance. One observation is that intensity changes are spatially localized rather than extended and wavelike. Furthermore, these changes occur over a wide range of scales, i.e., spatial frequencies. In order to identify the processes responsible for the image details within

this range the optimal filter must have a restricted bandwidth corresponding to a certain edge resolution in the output image.

The constraints placed on the edge filter are therefore conflicting. To restrict the range of scale in the filtered image the filter must possess a minimal bandwidth of standard deviation $\Delta\omega$. To localize edge positions accurately the filter's influence must be concentrated about those positions with minimal standard deviation Δx. The cost function to be minimized by this optimal filter, therefore, is the product of these two standard deviations, $\Delta x \Delta\omega$. Only a filter in the form of a Gaussian distribution, $g(x,y)$, can minimize this product [27].

However, filtering an image with a Gaussian function alone is only a smoothing operation reducing the range of detail by blurring the original. The position of the edge is defined to coincide with lines of steepest gradient in the filtered image. This indicates that an edge is characterized by a zero crossing in the Laplacian of the filtered image if intensity variation near and parallel to the line of zero crossings is locally linear (Marr and Hildreth's condition of linear variation). If linear variation does not hold, the position of the zero crossing will be displaced to one side of the true edge (i.e. gradient maximum).

There are a number of advantages to using the Laplacian for edge detection. It is an orientation independent operator, therefore producing only a scalar value for each pixel in the final image. The zero crossings produced form closed curves for edge phenomena totally enclosed by the image. This is an

important consideration should subsequent region or object segmentation schemes be applied. Finally, the Laplacian can be combined with the Gaussian to condense the image filtering process into one step:

$$\nabla^2[g(x,y) * I(x,y)] = \nabla^2 g(x,y) * I(x,y) \ . \qquad (2.1)$$

Therefore, Marr and Hildreth argue, the optimal edge detection filter to apply is the Laplacian of a Gaussian point spread function with the resulting zero crossings defining the image edges:

$$\nabla^2 g(x,y) = -[1 - (x^2 + y^2)/2\sigma_f^2] \exp[-(x^2 + y^2)/2\sigma_f^2]/(\pi\sigma_f^4) . \qquad (2.2)$$

The spatial frequency response of this filter is bandpass as described by

$$G''(u,v) = -4\pi^2(u^2 + v^2) \exp[-2\pi^2(u^2 + v^2)\sigma_f^2] \ . \qquad (2.3)$$

The filter's half power bandwidth is 1.2 octaves. This almost meets the one octave requirement of Logan's theorem [28] which claims that a one octave bandpass signal is completely determined by its zero crossings. Marr, Ullman and Poggio [29], however, have found experimentally that this bandwidth requirement can be relaxed considerably with little introduction of error. Therefore image detail in the resolution band of interest is fully described by the edges found on application of the $\nabla^2 g$ filter.

This bandpass character of the $\nabla^2 g$ filter would seem to weaken its claim to optimality by the previous argument since it was found that a lowpass, Gaussian filter was of the correct

form. Furthermore, since nothing has been said of the exact form of the edge features being sought, little can be said of the suitability of this filter for highlighting specific edge structures. These criticisms arise largely because of the heuristic nature of the arguments used in this filter's development. In fact the following argument will show that the $\nabla^2 g$ filter is asymptotic optimal up to a predetermined cutoff frequency.

2.2.2 Dickey and Shanmugam Optimality

By defining an edge as a unit step feature in image space, Dickey and Shanmugam derived an edge filter that optimized a very different measure of local influence. The optimal filter was defined as maximizing the proportion of output image energy in the vicinity of the edge location for given bandwidth and resolution requirements. In one dimension this translates into maximization of the following cost function:

$$\gamma = \frac{\int_{-I/2}^{I/2} |g(x)|^2 dx}{\int_{-\infty}^{\infty} |g(x)|^2 dx} \qquad (2.4)$$

where g(x) is the step spread function of the optimal filter, and I is the resolution interval centered on the step edge. The filter is also constrained to be bandlimited to radian frequency Ω.

With this information Dickey and Shanmugam were able to show that the optimal filter transfer function is given by

$$H(\omega) = \begin{cases} K_1 \omega \psi_1(c, \omega I/2\Omega), & |\omega| < \Omega \\ 0 & \text{elsewhere,} \end{cases} \tag{2.5}$$

where K_1 is a real constant, ψ_1 is the first degree zero order prolate spheroidal wave function and c is the resolution - bandwidth product, defined by

$$c = \Omega I/2 \ . \tag{2.6}$$

Note that c is similar in concept to the resolution - bandwidth product minimized in Marr's derivation. However, $H(\omega)$ does not seek to minimize c. Rather, c is specified by the filter designer and relates to the maximum γ attainable as given in Table I. Observe that Dickey and Shanmugam's approach provides a sharper measure of concentration of g(x) and $H(\omega)$ than that of Marr and Hildreth because integrations are now restricted to a finite interval in space and frequency.

c	0.5	1	2	4	8
$\gamma_{max}(c)$	0.00858	0.06279	0.35564	0.91211	0.99988

Table I. Values of maximum γ

Representing the filter as given in (2.5) makes for a number of problems when its practical implementation is considered. $\psi_1(c,x)$ must either be numerically computed or referenced from tables [30]. Also spatial convolution systems would require that either the two dimensional form of $H(\omega)$ be transformed to the

spatial domain with the discrete Fourier transform or that the line spread function of $H(\omega)$, the derivative of a rescaled $\psi_1(c,x)$, be converted to a point spread function through the Abel transform [31, p. 210]. These are cumbersome processes which do not admit well to an analysis of the filter's performance given a more realistic edge detection problem.

Addressing these difficulties, Shanmugam, Dickey and Green [32] attempted to replace $\psi_1(c,x)$ with a closed form asymptotic approximation. The resultant form of the filter was much more amenable to analysis than (2.5). Lunscher [33], however, pointed out that the applied approximation was improperly scaled and that the correct form of the asymptotic optimal filter is given by

$$H(\omega) \simeq \begin{cases} K\,\omega^2 \exp(-c\omega^2/2\Omega^2), & |\omega|<\Omega \\ 0 & \text{elsewhere,} \end{cases} \tag{2.7}$$

valid on $|\omega| < \Omega c^{-1/4}$.

Note that in the region $|\omega| < \Omega$ the form of $H(\omega)$ is that of the Laplacian of a Gaussian distribution. Furthermore, since $H(\omega)$ is an even function and the step edge model is an odd function, the resultant filter response, $g(x)$, is also an odd function. Therefore, the edges in an image filtered by the two dimensional extension of (2.7) are indicated by zero crossings in the output image.

From these results, it can be concluded that up to a suitable cutoff frequency, Ω, the Marr and Shanmugam filters are identical. Therefore the $\nabla^2 g$ filter is actually suboptimal in

the sense of maximizing the filtered image energy about step-like edges and bandlimiting the range of scales passed.

2.2.3 Unifying the Filters

By the Paley-Wiener Criterion [34, pp.16-20] neither $H(\omega)$ nor $G''(\omega)$ are physically realizable because the high frequency cutoff falls to zero faster than exponential order. Particularly in the case of $H(\omega)$, the sharp cutoff would be smoothed somewhat by the act of windowing the spatial filter or input image. The result of implementing the filters, then, is to force their form into closer agreement. The filters can therefore be related by agreeing upon a high cutoff frequency on $G''(\omega)$ which will correspond to Ω. By standard convention the 3 db half power point is chosen. It will later be shown that this occurs at

$$\Omega = 2\pi 0.325/\sigma_f \quad . \tag{2.8}$$

$G''(\omega)$ and $H(\omega)$ are therefore equivalent in form when

$$\sigma_f^2 = c/\Omega^2 = I/2\Omega, \tag{2.9}$$

resulting in

$$c = \Omega^2\sigma_f^2, \tag{2.10a}$$

$$I/2 = \Omega\sigma_f^2 \quad . \tag{2.10b}$$

Substituting (2.8) into (2.10):

$$c = 4.17,$$

$$I/2 = 2.04\sigma_f \quad .$$

Table I shows that this value of c corresponds to 91% of the filtered image energy being concentrated within the resolution

interval I. This implies a very low probability of encountering another zero crossing within $2\sigma_f$ of the main edge feature. This observation is supported by the probability analysis of $\nabla^2 g$ filtered white noise of Grimson [35], later corrected by Clark [36]. This would also imply that if image edges are part of a periodic structure such as a square wave, then if the edge spacing is less than $2\sigma_f$, the structure would be poorly seen, if at all. These performance questions are addressed in a more rigorous manner in the next section.

2.3 Predicting Filter Performance

It has been established that the $\nabla^2 g$ filter maximizes filtered image response about step-like edge structures, and controls the range of detail of these structures through its bandpass response. Furthermore, this passband being about one octave wide, the zero crossings in this image provide a complete description of the image detail passed. However, beyond the prediction that zero crossing spacing is unlikely to be narrower than $2\sigma_f$ no quantitative measure of the expected range of detail is evident. It can be argued that the frequency response, G"(u,v), provides this measure; but, as Marr and Hildreth correctly point out, real images do not consist of smooth sinusoidal variations in intensity. Rather, they consist of abrupt changes occurring over a variety of periods. Furthermore these intensity changes rarely exhibit the ideal abruptness of a step function. Processes such as diffraction of light, diffusion of ink and additive noise tend to blur such detail. The extreme case of the spacing of detail resolved in randomly structured images with respect to σ_f was examined by Grimson, and Clark.

However nothing was said of the magnitude of the resultant response against edge spacing in the original image.

Since real images do exhibit some form of measurable structure, what is needed is a method of relating the filter standard deviation, σ_f , to the magnitude of response seen at the edges of these structures. With these results the degree of invisible detail for σ_f can be identified. Also, since the $\nabla^2 g$ filter was originally modeled on the human visual system, predictions could be made concerning human visual performance.

The method chosen to predict the filter's performance was to observe its response when presented with two sets of periodic non-ideal edge structures. The result permits a process of filter design through the choice of a suitable σ_f for resolving image structures of a known regularity.

The two periodic edge models used were an ascending staircase, and a square wave pulse train. To model non-ideal edges blurred by various processes, these structures are convolved with another Gaussian of variable standard deviation σ_b . Gaussian blur of this kind is the kernel of the diffusion equation [37, pp.113-117] and so most accurately describes deterioration of edges through diffusion of dye through paper or exposed grains through photographic emulsion. Some success has also been reported on applying this model to blur found in industrial X-ray radiographs [38]. Shanmugam et al. examined the influence of blurring an ideal edge with essentially a first order approximation to a Gaussian. Their conclusions, translated in terms of a Gaussian, show that the filter retains optimal performance if the resolution interval is greater than $\pi\sigma_b/\sqrt{2}$,

i.e., $I/2 = 2\sigma_f > 1.11\sigma_b$. The following analysis will show this result to be too optimistic.

The edge models presented will be considered dependent on only one spatial variable. This will simplify the analysis through use of the filter line spread function:

$$\nabla^2 g_f(x) = \int_{y=-\infty}^{\infty} \nabla^2 g_f(x,y)\,dy$$

$$= -(1 - x^2/\sigma_f^2)\exp(-x^2/2\sigma_f^2)/(\sqrt{2\pi}\sigma_f^3) \qquad (2.11a)$$

with Fourier transform

$$G''(f) = -4\pi^2 f^2 \exp(-2\pi^2 f^2 \sigma_f^2)\ . \qquad (2.11b)$$

Edge magnitudes will be defined by the slope of the output signal at the zero crossing.

2.3.1 Staircase Edge Response

The edge model used to represent a blurred infinite staircase of ascending magnitude is,

$$e_{ST}(x) = g_b(x) * \sum_{n=-\infty}^{\infty} u(x - nT) \qquad (2.12)$$

where

$$g_b(x) = \text{Gaussian blur function} = \exp(-x^2/2\sigma_b^2)/\sqrt{2\pi\sigma_b^2}. \qquad (2.13)$$

The result of filtering $e_{ST}(x)$ with $\nabla^2 g_f(x)$ is,

$$e_{STO}(x) = \nabla^2[g_f(x) * e_{ST}(x)]$$

$$= \nabla g_f(x) * g_b(x) * \sum_{n=-\infty}^{\infty} \delta(x - nT), \qquad (2.14)$$

with Fourier transform

$$E_{STO}(f) = j2\pi f_s^2 \sum_{n=-\infty}^{\infty} n \exp[-2\pi^2 n^2 f_s^2(\sigma_b^2 + \sigma_f^2)]\ \delta(f - nf_s), \qquad (2.15)$$

where $f_s = 1/T$.

Normalize the edge spacing T and the blur standard deviation with respect to the filter standard deviation:

$$\sigma_b = a\sigma_f \,, \tag{2.16a}$$

$$T = \beta\sigma_f \,. \tag{2.16b}$$

Substituting into (2.15):

$$E_{STO}(f) = j2\pi/(\beta^2\sigma_f^2)\sum_{n=-\infty}^{\infty} n \exp[-2\pi^2 n^2(1 + a^2)/\beta^2]\,\delta(f - nf_s). \tag{2.17}$$

The magnitude of the slope of the zero crossing at the origin will define the edge magnitude:

$$\begin{aligned} |\nabla e_{STO}(x=0)| &= |j2\pi\int_{-\infty}^{\infty} fE_{STO}(f)df| \\ &= 8\pi^2/(\beta^3\sigma_f^3)\sum_{n=1}^{\infty} n^2 \exp[-2\pi^2 n^2(1 + a^2)/\beta^2]. \end{aligned} \tag{2.18}$$

The result is plotted in Figure 2.1 in db normalized to peak magnitude against a and β.

Observe that for small a ($a < 0.2$) the 3 db turn on spacing is $\beta = 2.75$. At large a the 3 db of peak region is at $a = 0.51$ for $\beta \geq 5.5$. Outside the 3 db level bordering the plateau the response decays linearly in db per decade a, and at a higher rate per decade β. Therefore it is expected that the edges are fully resolved in this image for

$$\sigma_b \leq 0.51\ \sigma_f \,,$$

$$T \geq 5.5\ \sigma_f \,, \quad \text{and in the case of negligible blur}$$

$$T \geq 2.75\ \sigma_f \,.$$

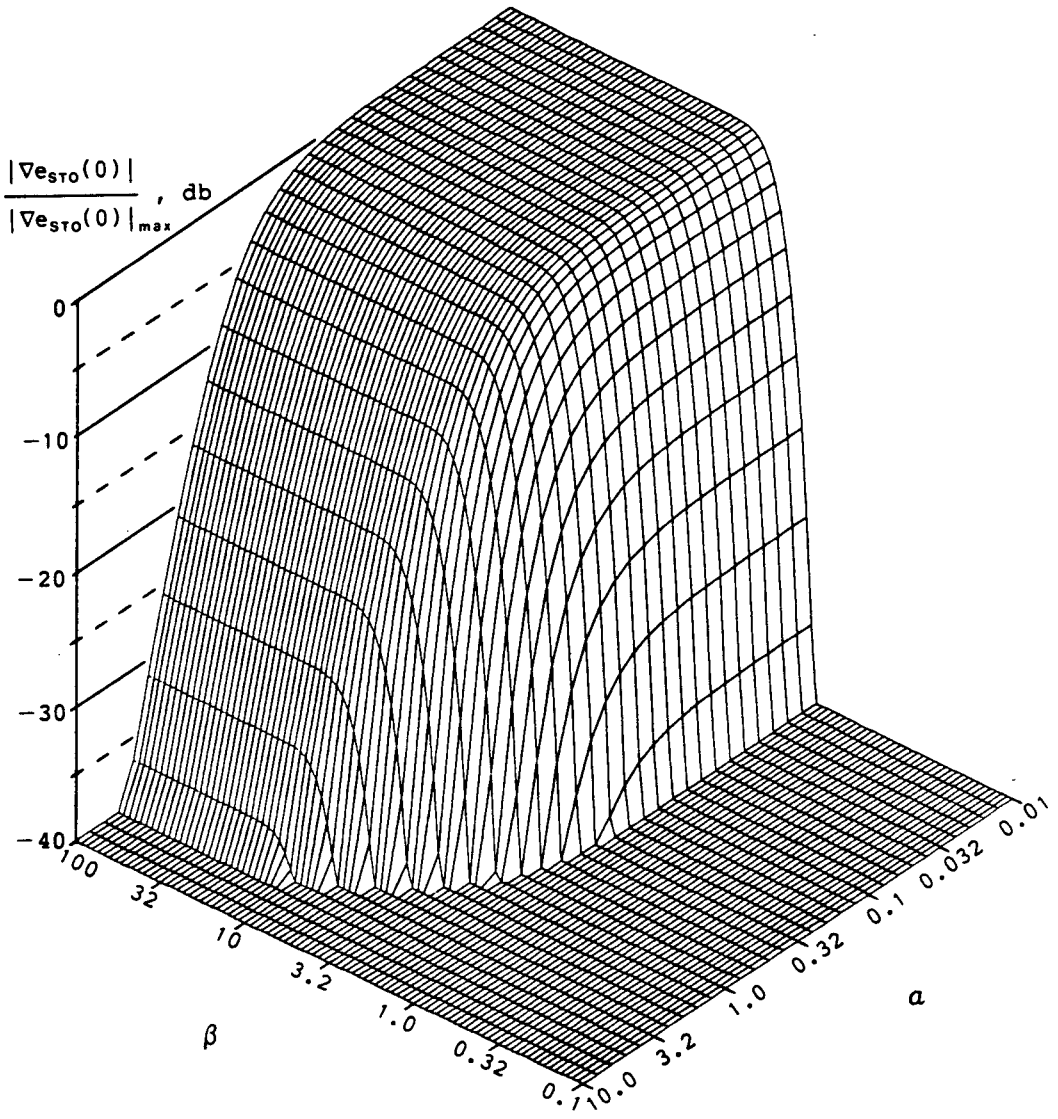


Figure 2.1 Staircase edge model magnitude response

It can be readily shown that there is another zero crossing between the steps at (n + 1/2)T of magnitude decreasing with increasing β for $\beta > 5$. The existence of this zero crossing may confuse the process of edge detection for these types of image structures. A method will be discussed later to resolve this problem using multiple filters of differing bandwidths.

2.3.2 Square-Wave Edge Response

A more common image structure is one of periodically rising and falling intensity. In its simplest form it can be modeled by a blurred square wave,

$$e_{sq}(x) = g_b(x) * u(x)u(T-x) * \sum_{n=-\infty}^{\infty} \delta(x - 2nT) \ . \qquad (2.19)$$

Here T represents the spacing of the edges of the square wave rather than the period of the square wave itself. After filtering with $\nabla^2 g_f(x)$ the result is

$$\begin{aligned} e_{sqo}(x) &= \nabla^2 [g_f(x) * e_{sq}(x)] \\ &= \nabla g_f(x) * g_b(x) * \{ \sum_{n=-\infty}^{\infty} [\delta(x - 2nT) - \\ &\quad \delta(x - (2n+1)T)] \} \end{aligned} \qquad (2.20)$$

with Fourier transform,

$$\begin{aligned} E_{sqo}(f) &= j2\pi f_s^2 \sum_{n=-\infty}^{\infty} n[1 - \exp(-jn\pi)] \\ &\quad \exp[-2\pi^2 n^2 f_s^2 (\sigma_b^2 + \sigma_f^2)] \ \delta(f - nf_s) \end{aligned} \qquad (2.21)$$

where $f_s = (2T)^{-1}$. Substituting the normalizing factors, (2.16):

$$E_{sqo}(f) = j\pi/(2\beta^2\sigma_f^2)\sum_{n=-\infty}^{\infty} n[1 - \exp(-jn\pi)]$$

$$\exp[-\pi^2 n^2(1 + a^2)/2\beta^2]\ \delta(f - nf_s)\ . \qquad (2.22)$$

The magnitude of the slope of the zero crossing at the origin is

$$|\nabla e_{sqo}(0)| = \left|j2\pi\int_{-\infty}^{\infty} f\ E_{sqo}(f)df\right|$$

$$= \pi^2/(\beta^3\sigma_f^3)\sum_{n=1}^{\infty} n^2(1 - \cos(n\pi))$$

$$\exp[-\pi^2 n^2(1 + a^2)/2\beta^2\ ]\ . \qquad (2.23)$$

This result is plotted in Figure 2.2.

Note that the response is more complex and sensitive to narrow spacing than in the case of the staircase model. The strong 5.3 db peak above steady state at β=1.81 for small a indicates that the edges are most strongly enhanced at the brink of visibility. The response reaches 3 db below the small a - large β plateau at β=1.15 for small a ($a < 0.2$). For large a this level of response is first reached for a=0.51 and $\beta \geq 1.36$. Note that this 3 db maximum blur level is identical to that of the staircase model. Finally, unchanging steady state response is achieved for a<0.51 and $\beta \geq 5.00$. Again it is seen that the response falls off linearly in db per decade a, and at a much higher rate per decade β. Therefore it can be concluded that a blurred square wave image is fully resolved for

$$\sigma \leq 0.51\ \sigma_f\ ,$$

$$T \geq 1.36\ \sigma_f\ ,$$ and in the case of negligible blur

$$T \geq 1.15\ \sigma_f\ .$$

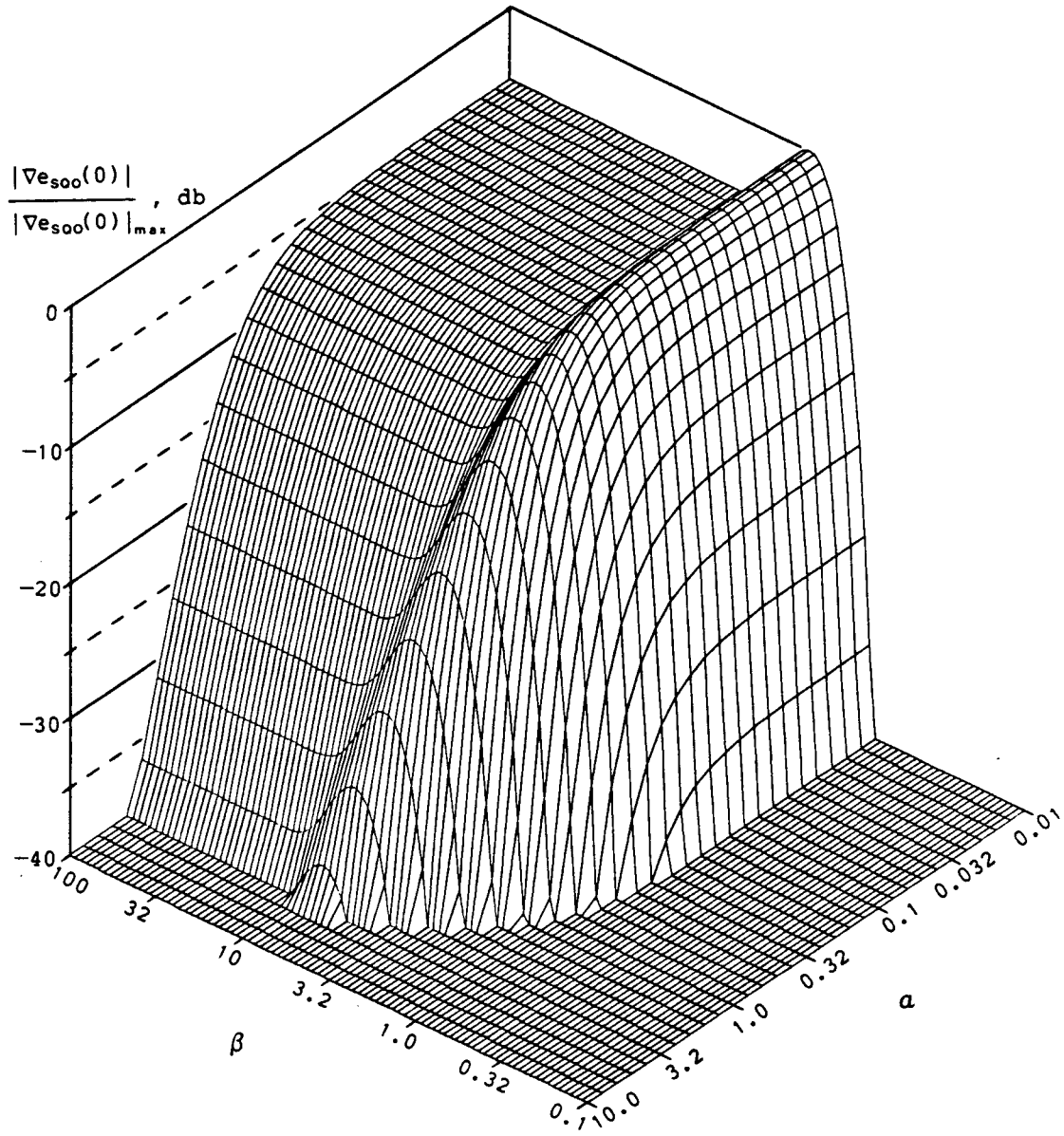


Figure 2.2 Square wave edge model magnitude response

2.4 Performance in Additive Noise

The forgoing analysis has succeeded in providing a guide as to when certain edge structures can be expected to become visible given σ_f. However, since the signal models used were perfectly deterministic, the filter performance in the presence of additive noise remains to be resolved. To address this problem, retain the one-dimensional signal models used thus far, and corrupt them with additive white Gaussian noise of power spectral density $\eta_0/2$.

The first question to arise concerns the accuracy of the zero crossings as a measure of the true edge positions. Consider an isolated filtered edge signal, $y_0(x)$, centered on the zero crossing with filtered Gaussian noise, $n(x)$, added. Being a random process, nothing can be said of where exactly $n(x) + y_0(x)$ will cross the x-axis nearest $x=0$. Instead, a statistical description in the form of the standard deviation of the zero crossing about the origin, σ_z, must be sought. To make the problem manageable, assume a linear model for both the signal and the noise at the origin. The signal then becomes

$$y_1(x) = Ax \tag{2.24a}$$

and the noise, with y-intercept n_0 and slope $\dot{n}_0$, becomes

$$y_2(x) = \dot{n}_0 x + n_0 \tag{2.24b}$$

Clearly this assumption of linearity has a limited range. For the signal, this range is bounded by the signal peak at $|x| = \Delta$. For an ideal step edge, corresponding to ideal staircase and square wave edges with $\beta > 5$, this occurs at $|x| = \sigma_f$. Plausible

values of σ_z should therefore be bounded by $\sigma_t/2$ for at least 95.4% of the zero crossings to be contained in the linear region. This in turn will lead to a lower bound on the signal to noise ratio in both the filtered and unfiltered image for which the filter can be expected to yield accurate results.

The most straightforward approach to determining σ_z involves solving the variance of the x-intercept of the sum of y_1 and y_2, i.e.,

$$y_1(x) + y_2(x) = (A + \dot{n}_0)x + n_0 = 0,$$

yields an x-intercept of

$$x_0 = -n_0/(A + \dot{n}_0).$$

The variance of x_0 forms the variance of the zero crossing estimate, σ_z^2:

$$\sigma_z^2 = E[n_0^2/(A + \dot{n}_0)^2] - E[n_0/(A + \dot{n}_0)]^2. \tag{2.25}$$

Since a random variable and its derivative are uncorrelated [39, p. 245], and uncorrelated Gaussian processes are independent

$$\begin{aligned}\sigma_z^2 &= E[n_0^2]\, E[(A + \dot{n}_0)^{-2}] - E[n_0]^2 E[(A + \dot{n}_0)^{-1}]^2 \\ &= E[n_0^2]\, E[(A + \dot{n}_0)^{-2}].\end{aligned} \tag{2.26}$$

where $E\{n_0\} = 0$ since the filter does not pass dc. Therefore, writing the variance of n_0 and $\dot{n}_0$ as $\overline{n_0^2}$ and $\overline{\dot{n}_0^2}$,

$$\sigma_z^2 = \int_{-\infty}^{\infty} \frac{n_0^2 \exp(-n_0^2/2\overline{n_0^2})\,dn_0}{\sqrt{2\pi\overline{n_0^2}}} \int_{-\infty}^{\infty} \frac{\exp(-\dot{n}_0^2/2\overline{\dot{n}_0^2})\,dn_0}{(A + \dot{n}_0)^2\sqrt{2\pi\overline{\dot{n}_0^2}}}\,. \tag{2.27}$$

The first integral simply yields $\overline{n_0^2}$. However, the second

integral cannot be directly evaluated because of the singularity at $\dot{n}_0 = -A$. This singularity implies an intercept at infinity which underscores the limitations of the linear model applied here. In fact any $\dot{n}_0$, which, together with n_0, produces an intercept outside $|x_0|=\sigma_t$ violates the linearity assumption. By integrating $\dot{n}_0$ only over the region which maintains x_0 within the bounds of $|x_0| < \sigma_t$, slopes outside of this region can be effectively ignored as highly improbable. However, this solution is not satisfactory for the simple reason that slopes about $\dot{n}_0 = -A$ are perfectly reasonable since the filter obviously passed the signal with that slope.

A more workable approach is made possible by estimating σ_z from the x-intercept of the noise standard deviation bounds about the signal near the zero crossing. This time, assume a linear model only for the signal within $|x| < \Delta$, and add to this the noise, n(x):

$$y(x) = Ax + n(x).$$

Since the noise is assumed stationary, y(x) has a constant variance of $\overline{n_0^2}$ and standard deviation

$$\sigma_0 = \sqrt{\overline{n_0^2}} \tag{2.28}$$

centered on the signal. The x-intercepts of the lines passing through the upper and lower bounds on y(x) as defined by σ_0 will be used to estimate σ_z. Figure 2.3 shows these lines superimposed on the signal model. The line defined by the lower bound is

$$y_1(x) = Ax - \sigma_0 \tag{2.29}$$

with x-intercept

$$x_0 = \sigma_0/A. \tag{2.30}$$

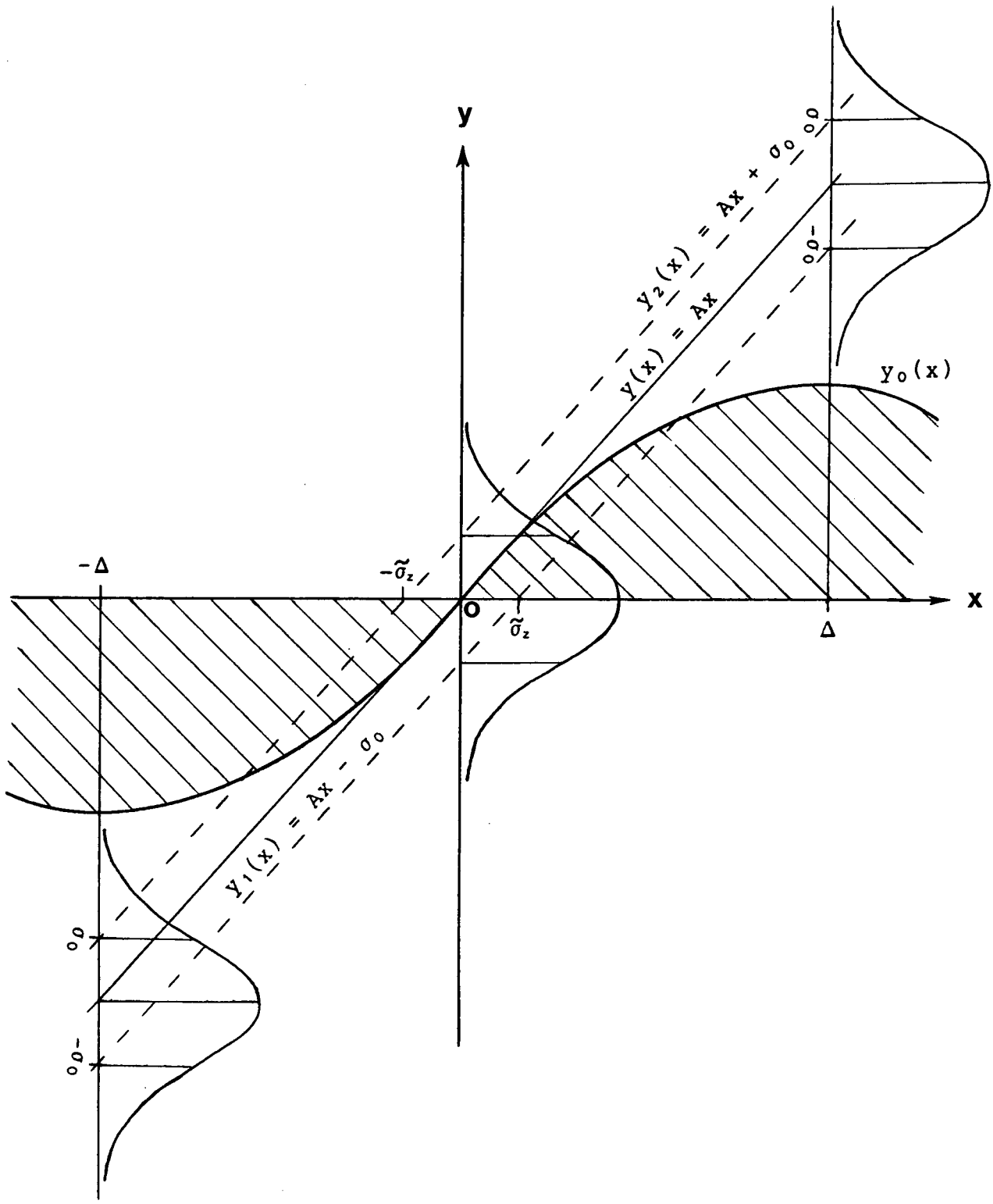


Figure 2.3 Calculation of the zero crossing standard deviation in the presence of noise

The x-intercept of the upper line is obviously of equal magnitude and opposite sign. Being the intercept, or zero crossing, of the standard deviation bounds of y(x), x_o will serve as the estimate of the standard deviation of the zero crossing, $\tilde{\sigma}_z$.

Before proceeding, it is necessary to evaluate σ_o. The filtered noise power spectral density is given by

$$S_{no}(f) = |G''(f)|^2 S_n(f) \tag{2.31}$$

where, from equation (2.11b),

$$|G''(f)|^2 = 16\pi^4 f^4 \exp(-4\pi^2 f^2 \sigma_f^2). \tag{2.32}$$

and

$$S_n(f) = \eta_o/2.$$

Since n(x) is an ergodic process the filtered noise variance is given by the output noise power,

$$\begin{aligned} \overline{n_o^2} &= \int_{-\infty}^{\infty} S_{no}(f)df \\ &= 16\pi^4 \eta_o \int_0^{\infty} f^4 \exp(-4\pi^2 f^2 \sigma_f^2). \\ &= 3\eta_o/(16\sqrt{\pi}\sigma_f^5). \end{aligned} \tag{2.33}$$

Substituting $\overline{n_o^2}$ into (2.28) and (2.30), the estimate for the zero crossing standard deviation becomes

$$\tilde{\sigma}_z = \frac{1}{A}\left(\frac{3\eta_o}{16\sqrt{\pi}\sigma_f^5}\right)^{1/2} \tag{2.34}$$

The magnitude of the slope at the zero crossing, A, was determined earlier for the periodic edge models. Substituting (2.18) and (2.23) yields $\tilde{\sigma}_z$ for

The staircase model:

$$\tilde{\sigma}_{zST} = \left(\frac{3\eta_0\sigma_f}{16\pi^{9/2}}\right)^{1/2} \left\{ \frac{8}{\beta^3} \sum_{n=1}^{\infty} n^2 \exp[-2\pi^2 n^2(1+a^2)/\beta^2] \right\}^{-1} \quad (2.35)$$

The square wave model:

$$\tilde{\sigma}_{zSQ} = \left(\frac{3\eta_0\sigma_f}{16\pi^{9/2}}\right)^{1/2} \left\{ \frac{1}{\beta^3} \sum_{n=1}^{\infty} n^2(1-\cos n\pi)\exp[-\pi^2 n^2(1+a^2)/2\beta^2] \right\}^{-1} \quad (2.36)$$

As was observed earlier, these models converge on ideal step edge performance for $a << 1$, $\beta > 5$. To simplify further analysis assume these conditions to be true. In this case the terms in the parenthesis of (2.35) and (2.36) are all equal to

$$\{ \ldots \} = 4.0421 \times 10^{-2}.$$

Therefore for the case of fully resolved edges,

$$\tilde{\sigma}_z = 24.74 \left(\frac{3\eta_0\sigma_f}{16\pi^{9/2}}\right)^{1/2}. \quad (2.37)$$

Clearly this estimate predicts that the standard deviation of the zero crossing in the presence of noise varies as the root of the filter standard deviation. Therefore in order to achieve a more accurate measure of the edge position the narrowest filter available should be used. This seems intuitively correct since the smaller σ_f, the more spatially local the filter influence. However a wider filter would be more effective in smoothing the noise and would therefore be tolerable of lower signal to noise

ratios.

To test this hypothesis and establish the lowest bound on the signal to noise ratio given σ_f, a limit must be placed on $\tilde{\sigma}_z$. As illustrated earlier the linear model used in the derivation of $\tilde{\sigma}_z$ breaks down for $\tilde{\sigma}_z > \Delta$. For the fully resolved edges used to find (2.37), $\Delta = \sigma_f$. Therefore the distribution of zero crossings must largely be contained within a filter standard deviation which, to 95.4% probability, implies

$$\frac{\tilde{\sigma}_z}{\sigma_f} = 24.74 \left(\frac{3\eta_0}{16\pi^{9/2}\sigma_f} \right)^{1/2} < 0.5 \qquad (2.38)$$

Since ideal square waves can be resolved with an edge spacing as little as $1.15\sigma_f$ the above criteria could be made even more restrictive to guarantee meaningful edge positions. However, such adjustments can be readily made to (2.38) as needed. σ_f and η_0 are therefore bounded by

$$\sigma_f / \eta_0 > 2448 \, [3/(16\pi^{9/2})] = 2.66 \, . \qquad (2.39)$$

This result can be used to establish the minimum signal to noise ratio in the filtered resolution interval, and initial image for the $\tilde{\sigma}_z$ estimate to be valid. As was shown by Lunscher [33], refined from Shanmugam et al. [32], the signal to noise ratio within the resolution interval of an ideal step edge after filtering with the asymptotic optimal filter is

$$SNR_0 = \frac{\gamma\Omega}{C\ \eta_0} \frac{\int_0^{\Omega} \omega^2 \exp(-c\omega^2/\Omega^2)\,d\omega}{\int_0^{\Omega} \omega^4 \exp(-c\omega^2/\Omega^2)\,d\omega} \quad . \tag{2.40}$$

Since the $\nabla^2 g$ filter is not sharply bandlimited to Ω, as the filter used to derive (2.40), the integration limits may extend to infinity. By fixing Ω to the 3 db high cutoff frequency, it was shown earlier that $\gamma=0.91$. Therefore, substituting for γ, and (2.8) and (2.10a) for Ω and c, (2.40) becomes

$$SNR_0 = \frac{1.4}{\pi\sigma\ \eta_0} \frac{\int_0^{\infty} \omega^2 \exp(-\sigma_f^2\omega^2)\,d\omega}{\int_0^{\infty} \omega^4 \exp(-\sigma_f^2\omega^2)\,d\omega}$$

$$= 2.8\sigma_f/(3\pi\eta_0). \tag{2.41}$$

representing the signal to noise ratio within $2\sigma_f$ of the $\nabla^2 g$ filtered zero crossing. Substituting inequality (2.39) into this result,

$$SNR_0 > 2.8(2.66)/(3\pi) = 0.79 \tag{2.42}$$

provides the minimum signal to noise ratio about the zero crossing for $\tilde{\sigma}_z$ to be meaningful. Clearly this is a very low SNR figure by most image processing standards. Furthermore, since (2.41) shows SNR_0 increasing directly with σ_f, limit (2.42) simply bounds the narrowest usable filter.

Of greater interest than the signal to noise ratio of (2.41) is that of the unfiltered image, SNR_i. The ideal step and square wave image models yield identical signal power of

$$S_i = \lim_{T\to\infty}[(1/2T)\int_{-T}^{T} u(t)^2 dt] = 0.5 \ . \qquad (2.43)$$

Since the noise spectrum was assumed to be white, the input noise power is ideally infinite. However, since image processing is generally done on a sampled image the initial image must be lowpass filtered with a maximum bandwidth of half the sampling rate to avoid aliasing effects. The noise power passed is clearly finite with a maximum value of

$$N_0 = 2(\eta_0/2)(2T)^{-1} \qquad (2.44)$$

where T is the sampling period. Normalize T to the filter standard deviation with

$$T = \delta\sigma_f \ . \qquad (2.45)$$

Substituting (2.45) into (2.44) produces the normalized noise power,

$$N_0 = \eta_0/(2\delta\sigma_f) \ . \qquad (2.46)$$

Together with S this shows the input signal to noise ratio to be

$$SNR_i = S_i/N_0 = \delta\sigma_f/\eta_0 \ . \qquad (2.47)$$

Substituting (2.39) it is found that the range for which $\tilde{\sigma}_z < \sigma_f/2$ is

$$SNR_i > 2.66\delta \ . \qquad (2.48)$$

As the filter width increases, δ decreases lowering the required signal to noise ratio. To establish the maximum lower bound on SNR_i note that the highest frequency edge feature possible in the sampled image would be a square wave or pulse train of period 2T

with effective edge spacing of T. Since such a feature is highly aliased, and therefore poorly represented by the act of sampling, the filter sensitivity expected would correspond to that of a somewhat blurred square wave. The initial detection spacing for such a feature is about $1.25\sigma_f$. Therefore, with this σ_f, the maximum δ is 1.25. Substituting into (2.48):

$$SNR_i(\delta=1.25) > 3.325 \quad . \tag{2.49}$$

This establishes the most pessimistic lower bound on the input signal to noise ratio. This figure is higher than that required within the filtered image, (2.41), but is still remarkably low and seen to decrease without limit as σ_f is increased. Of course increasing σ_f decreases the level of detail that can be resolved, and, by increasing $\tilde{\sigma}_z$, decreases the accuracy of the edge position.

When SNR_i satisfies (2.48) it can be used directly to estimate $\tilde{\sigma}_z$. Substituting (2.47) into (2.37) gives

$$\tilde{\sigma}_z = 24.74 \left(\frac{3\delta\sigma_f^2}{16\pi^{9/2}SNR_i} \right)^{1/2} . \tag{2.50}$$

On adopting the pixel spacing, T, as the unit of measure, $\delta\sigma_f = 1$. Therefore,

$$\tilde{\sigma}_z = 24.74 \left(\frac{3\sigma_f}{16\pi^{9/2}SNR_i} \right)^{1/2} . \tag{2.51}$$

The edge position accuracy is therefore improved by increasing signal to noise ratio, as expected.

The foregoing analysis has addressed the questions of the zero crossing accuracy and signal to noise ratio within the resolution interval. The question remains as to the image quality outside the resolution interval. Since this region can be quite large in the case of large features and a narrow filter, the quality can best be addressed through the signal to noise ratio there, SNR_0'. For the case of the ideal step edge model extending to infinity the signal energy outside the resolution interval is $(1 - \gamma)$. The signal power is therefore zero. To render the problem more realistic place this edge centrally within a large image of finite diameter D much larger than $I = 2\sigma_f$. In this case SNR_0' is simply found by replacing γ with $(1 - \gamma)$ in (2.40), and multiplying by I/(D-I). Allowing the integration limits to go to infinity as before and substituting for c and Ω gives:

$$SNR_0' = \frac{4(1-\gamma)\sigma_f^2}{3(D-I)\eta_0} \simeq \frac{4(1-\gamma)\sigma_f^2}{3D\ \eta_0} \quad . \tag{2.52}$$

Clearly SNR_0' is less than SNR_0. Also, as this region becomes more removed from the zero crossing by increasing I, γ rapidly approaches unity because c increases. Therefore the signal to noise ratio approaches zero for regions far removed from the zero crossing.

Intuitively this conclusion can be generalized to the previous edge models when fully resolved. Without repeating the details here, it was found that SNR_0' decreases inversely with β for staircase and square wave edges. The presence of blur serves

to decrease SNR_o', and SNR_o, further by $1/a$ for $a > 0.51$. The expected result, then, on filtering a noisy, periodic image with a large β filter is an essentially noise free area about $2\sigma_f$ to either side of the true edge positions surrounded by wide areas filled with random noise edges. Since any edge structures closer than $1.15\sigma_f$ cannot be resolved, these noise edges will have dimensions commensurate with those of the minimum resolvable image detail.

This noise analysis has served to point out again the conflicting nature of the demands placed on the performance of an edge detector. Ideally the edge detector should reject all noise detail, enhancing only the edges of known image objects. From the foregoing, this requires forcing the resolution intervals about these edges to overlap through selection of the widest filter possible. The edge models analyzed will serve as the guide for this selection. However, the edge detector should also pinpoint the edge positions with maximum accuracy. As was seen, this is best achieved with the narrowest filter possible, hence the conflict.

One way to resolve this conflict is to filter the image with a narrow filter and then threshold the resultant edges at some suitable level. The premise behind this procedure is that the edges generated by noise would generally have a lower gradient magnitude than those generated by image objects. However, because of the statistical nature of the noise there must be incidences where the noise edges will exceed the threshold, and likewise incidences where weak or corrupted object edges fall below. Therefore, though there are a variety of threshold

selection techniques that can be applied [40], their performance can only be statistical at best, resulting in an image of broken edge segments. This is unfortunate since the $\nabla^2 g$ filter is guaranteed to produce closed loop edges for fully contained objects. To restore the image to this condition a post-processing algorithm must be applied to screen out the smallest edge segments and repair the larger ones.

Another approach, founded on the same premise as thresholding, is to apply relaxation techniques to the filtered image. Again there are a variety of relaxation techniques available [41], and they can be adapted to insure closed edges. The basic concept here is to examine the magnitudes of the edge segments on a pixel by pixel basis and assign a confidence value to each segment. In successive iterative passes over the image, edges of low confidence are progressively eliminated while those of high confidence are formed into closed structures. Though relaxation would filter out noise while retaining closed object edges, it does so at the expense of requiring another image-size array to store the confidence measures. Processing time may also increase considerably since a small algorithm must be executed at each pixel in the image over the course of a number of iterations. The performance is also influenced by the qualitative selection of certain parameters such as the magnitude of the confidence level increment [42].

Both the above techniques are subject to another failing. If the observation of gross image detail is desired and a narrow filter chosen for the purpose of precisely locating that detail, then fine object detail will also appear. Furthermore, as seen

in the square wave response plot, this fine detail can produce a substantially larger response than the gross detail when its dimensions coincide with the peak after turn-on. The net result is that these edges would certainly be given high confidence during post-processing, thereby obscuring the gross detail.

The approach to solving this conflict, while avoiding the previous difficulties, is multiband filtering.

2.5 Multiband Filtering

The idea of multiband filtering is simple. Initially filter the image with the widest filter appropriate to the level of detail desired and note the edge positions. Then filter the image with progressively narrower filters retaining only those edges in close proximity to the initial edges. This process is terminated when the desired level of edge precision is attained.

This method of refining edge positions hierarchically through stages of differing resolution characterizes a class of techniques known as processing cones [43]. Often this approach involves three stages of processing at each level of resolution. A typical example is given by the pioneering work of Kelly [44] where the images of low resolution are produced by local averaging of the original image. A simple edge detector was then applied followed by a relaxation procedure to remove extraneous detail.

In contrast, the $\nabla^2 g$ filter performs this operation in essentially one step. Use of a wide filter matched to the gross image detail smooths out fine image features.

Furthermore, since it contains no peaks in the frequency response above Ω, it performs this operation better than local averaging which does possess such peaks. The act of edge detection is limited to a search for zero crossings, and since these crossings are guaranteed to be closed no relaxation stage is necessary.

The remaining structure of the processing cone depends on the objective of the imaging system.

If the objective is precise measurement of the dimensions of gross image detail then the following method would be applied. Begin with an estimate of the size of the image object of interest and the degree of blur present. Depending on which of the edge models suits this image best, figures 2.1 and 2.2 will supply the minimum β for σ_{f1} to just resolve the image. If possible, also estimate the signal to noise ratio which, with equation (2.51), provides $\tilde{\sigma}_z$. After filtering, the zero crossing positions are noted.

The next level of detail is determined by a filter of width $\sigma_{f2}=\tilde{\sigma}_{z1}$. The search for the new edge position is performed within one σ_{f2} resolution interval about the old σ_{f1} edges. This interval is $4\sigma_{f2}$ wide. Since the zero crossing locations were assumed Gaussian distributed, with 95.4% confidence, subsequent edges will be located within $2\tilde{\sigma}_{z1}$ of the crossing. Therefore the width of the search interval should be $4\tilde{\sigma}_{z1}= 4\sigma_{f2}$. After locating the edges in this interval this process is repeated until the desired level of precision is attained or until $\tilde{\sigma}_{z2} > \sigma_{f2}$ in which case the extra precision is lost in noise.

On the other hand, if the objective of the imaging system is to examine and classify all levels of image detail, or if $\tilde{\sigma}_z$ is too difficult to estimate, then a set of filters must be applied which together span all practical spatial frequencies. Since the $\nabla^2 g$ filter is bandpass with a bandwidth of 1.2 octaves, minimum overlap is achieved if successive bands are separated by 1.2 octaves. However to make some allowance for error and ease of implementation a one octave spacing is preferred. Therefore, after choosing a suitably large σ_{f1}, successive filter widths are simply halved until the maximum precision is attained. A method similar to this was used by Grimson [35] to control vergence for the determination of depth from the binocular disparity information in stereograms.

One additional, important application of multiband filtering is the elimination of the pseudo-edges in staircase type images. You will recall that when considering the staircase edge model it was pointed out that another zero crossing was located exactly half way between the true edge positions. Unlike the true-edge zero crossing, however, the magnitude of this pseudo-edge decreases with increased edge spacing. The presence of this pseudo-edge can confuse an imaging system using the $\nabla^2 g$ filter. To eliminate it, it is noted that in any practical system the $\nabla^2 g$ filter attains a finite size before falling below the minimum quantization level, and that this size decreases with σ_f. Consequently a pseudo-edge that is well established for one filter size will eventually bifurcate into two parallel edges for progressively narrower

filters. Further progression will cause these two edges to diverge from the original low resolution pseudo-edge position, a behavior exactly opposite to that of true edges. Inclusion of provisions for the detection of this divergence, would allow a multiband system to ignore these false edges.

2.6 Sampling the $\nabla^2 g$ Filter

To this point in our discussion the $\nabla^2 g$ image filter has been represented in its continuous form. This has proved useful because it permitted analytic derivation of its performance under various circumstances. However, with the exception of optical signal processing, image processing involves sampled data signals. Such discrete signal processing requires that the filter be sampled and quantized. In this section the act of sampling will be considered, and in the next, quantization. The objective will be to determine a minimum sampling rate before aliasing effects become excessive in a discrete convolution implementation.

Recall that the $\nabla^2 g$ two dimensional point spread function is:

$$\nabla^2 g(x,y) = -[1-(x^2+y^2)/2\sigma_f^2]\exp[-(x^2+y^2)/2\sigma_f^2]/(\pi\sigma_f^4) \qquad (2.53)$$

with frequency response:

$$G''(u,v) = -4\pi^2(u^2+v^2)\exp[-2\pi^2(u^2+v^2)\sigma_f^2] \ .$$

To ideally sample $\nabla^2 g$ multiply (2.53) by

$$\delta_T(x,y) = \sum_{m=-\infty}^{\infty}\sum_{n=-\infty}^{\infty}\delta(x-mT,y-nT), \qquad (2.54)$$

which, on substituting $T = \delta\sigma_f$, yields

$$\nabla^2 g(m,n) = -[1-(m^2+n^2)\delta^2/2]\exp[-(m^2+n^2)\delta^2/2]/(\pi\sigma_f^4) \qquad (2.55)$$

with frequency response,

$$G''(u,v)_T = G''(u,v) * [\delta_T(u)\delta_T(v)]/T^2 \qquad (2.56)$$

where

$$\delta_T(x) = \sum_{k=-\infty}^{\infty}\delta(x-k/T), \quad k = \begin{cases} m, & \text{for } x=u \\ n, & \text{for } x=v. \end{cases}$$

Because $G''(u,v)$ is not bandlimited, the frequency convolution causes a certain degree of aliasing between the harmonics of $G''(u,v)_T$. The aliasing with the baseband ($|u|, |v| \leq (2T)^{-1}$) is most severe along the u, v axis where the harmonics are most closely spaced. For the time being let's assume that the contribution from off-axis harmonics is negligible, and that the worst case sampled response can be adequately represented along a single axis. Label this the f axis, for consistency with previous sections, and set the other spatial frequency component to zero. The simplified sampled response then becomes

$$G''(f)_T = \frac{-4\pi^2 f^2}{T^2} \exp(-2\pi^2 f^2 \sigma_f^2) * \sum_{n=-\infty}^{\infty}\delta(f-n/T) \; . \qquad (2.57)$$

To investigate the degree of aliasing, let us examine the spectral energy density of the baseband harmonic:

$$\mathcal{E}''(f) = -16\pi^4 f^4 \exp(-4\pi^2 f^2 \sigma_f^2)/T^4 \; . \qquad (2.58)$$

Most of the aliasing effects come as a result of overlap with the

first harmonic centered at $f = 1/T$. To estimate the degree of this overlap, the amount of aliased filter energy will be found by integrating $\mathcal{E}'(f)$ from $f = (2T)^{-1}$ to infinity, ie.,

$$\mathcal{E}_a = \int_{(2T)^{-1}}^{\infty} \mathcal{E}'(f)df = \int_0^{\infty} \mathcal{E}'(f)df - \int_0^{(2T)^{-1}} \mathcal{E}'(f)df \ . \qquad (2.59)$$

A perhaps more useful measure is the fraction of aliased energy with respect to the baseband energy,

$$\mathcal{E}_{af} = 1 - \int_0^{(2T)^{-1}} \mathcal{E}'(f)df \Big/ \int_0^{\infty} \mathcal{E}'(f)df$$

$$= 1 - \frac{\int_0^{(2T)^{-1}} f^4 \exp(-4\pi^2 f^2 \sigma_f^2)df}{\int_0^{\infty} f^4 \exp(-4\pi^2 f^2 \sigma_f^2)df} \ . \qquad (2.60)$$

Normalize (2.60) with $f = (\delta\sigma_f)^{-1}$ so that $T = \delta_s \sigma_f$:

$$\mathcal{E}_{af} = 1 - \frac{\int_0^{(2\delta_s)^{-1}} \delta^{-4} \exp(-4\pi^2/\delta^2)d\delta^{-1}}{\int_0^{\infty} \delta^{-4} \exp(-4\pi^2/\delta^2)d\delta^{-1}}$$

$$= 1 - \frac{256}{3}\pi^{9/2} \int_0^{(2\delta_s)^{-1}} \delta^{-4} \exp(-4\pi^2/\sigma^2)d\delta^{-1} \ . \qquad (2.61)$$

The remaining integral was evaluated numerically and the resultant $\mathcal{E}_{af}$ was plotted in Figure 2.4 together with $\mathcal{E}'(f)$ also normalized with $f = (\delta\sigma_f)^{-1}$ and unit peak, Figure 2.5. Note in Figure 2.5 the key points of the frequency response: peak at $f = \sqrt{2}/(2\pi\sigma_f)$, and half power points at $f = 0.139/\sigma_f$ and $0.325/\sigma_f$. The latter leads to the previous choice of Ω.

Figure 2.4, then, illustrates how the energy of figure

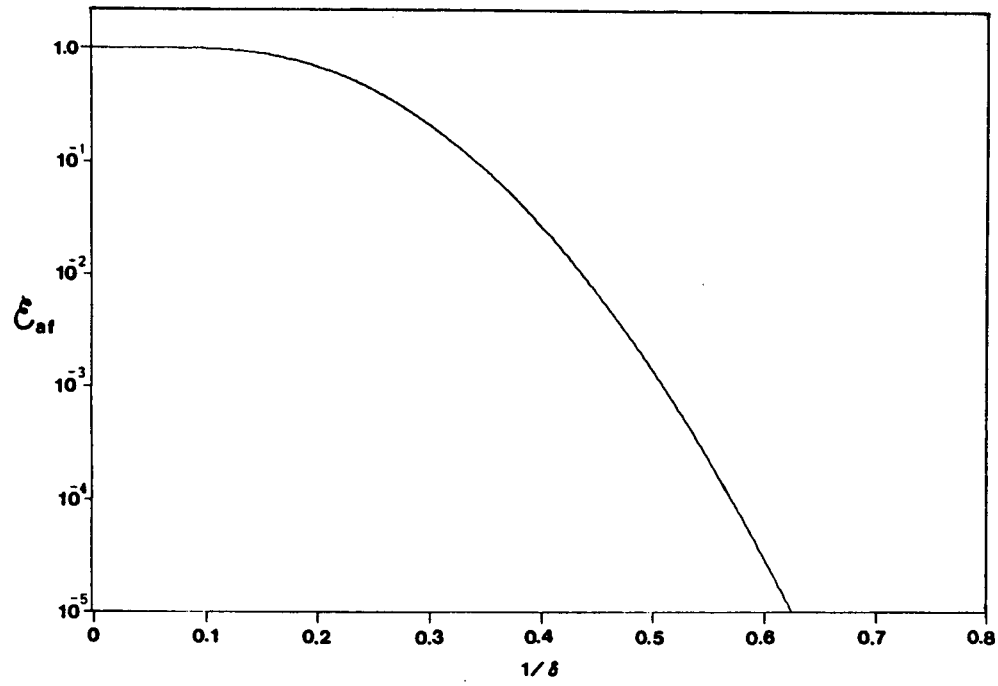


Figure 2.4 Proportion of aliased filter energy plotted against the normalized half-sampling frequency

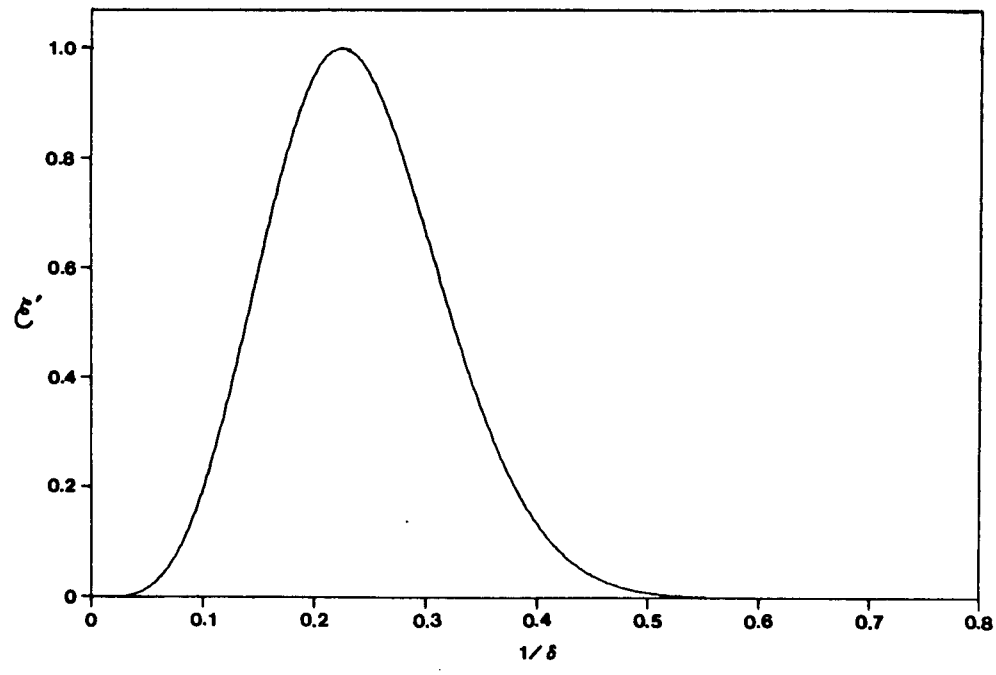


Figure 2.5 $\nabla^2 g$ normalized spectral energy density

2.5 is distributed. Note that for $\delta_s = 1$ leading to $T = \sigma_f$ only 0.14% of the filter energy is contained in frequencies above $f_s/2 = 0.5/\sigma_f$. Aliasing can be considered to become extreme when 10% of the filter energy is above $f_s/2$. This is seen to occur at $\delta=2.92$, leading to a sample spacing of $1.46\sigma_f$. As shown earlier, the maximum filter resolution desirable would be one pixel wide for square wave or pulse edges. At such a sample spacing, these edge spectra would be so aliased as to consist of only impulses at $f = (2T)^{-1}$. If the filter is sampled to just resolve such a feature, then $T \simeq 1.25\sigma_f$. δ_s therefore is 1.25. On examining Figure 2.4, it is seen that only 2.7% of the filter energy lies above $f_s/2 = 0.4/\sigma_f$. Therefore, even for the narrowest filter desirable, the degree of aliasing is slight.

What of the previous assumption that off-axis harmonics and second or higher on-axis harmonics have negligible effect? The next greatest influence beyond the on-axis first harmonic would come from the first off-axis harmonic. From the foregoing, the nearest of these is centered at least $0.8/\sigma_f$ frequency units away from the axis, ie., $\delta = 1.25$. From Figure 2.4, it is clear that the amount of filter energy above $\delta=1.25$ is negligible, therefore confirming the previous assumptions.

In conclusion then, filter response can be considered ideal for filters with $\delta < 1$, and even for the narrowest desirable filter, $\delta=1.25$, the degree of aliasing in the filter response is slight, compared to that of the finest edge models considered resolvable.

2.7 Coefficient Quantization

Sampling is the first step for digital representation of a filter. The second step is quantization of the resulting coefficients. The limit of quantization is determined by the processing system to be used and can range from 64 bits for large mainframes, to 16 or 8 bits for minis and micros, or even 6 bits for special systems intended to match typical video resolution [45, p. 114]. Again a direct form discrete convolution implementation will be considered. The objective of this analysis will be to examine the influence of the quantization error on the filter spectrum and thereby develop a strategy for the selection of the minimum word size necessary to keep the error within acceptable bounds.

The analysis of the quantization errors in one-dimensional direct form finite impulse response (FIR) filters was first performed by Chan and Rabiner [46]. Their techniques have been adapted here for two-dimensional filters in general, and the $\nabla^2 g$ filter in particular.

The spatial frequency response for an arbitrary, non-causal (2N-1) X (2N-1) two-dimensional FIR filter with point spread function h(m,n), symmetrical about the origin, is given by,

$$H(e^{j\omega_1}, e^{j\omega_2}) = \sum_{n=(N-1)}^{N-1} \sum_{m=(N-1)}^{N-1} h(m,n) e^{j\omega_1} e^{j\omega_2}$$

$$= h(0,0) + 2\sum_{n=1}^{N-1} h(m,0)\cos\omega_1 m + 2\sum_{n=1}^{N-1} h(0,n)\cos\omega_2 n + 4\sum_{m=1}^{N-1}\sum_{n=1}^{N-1} h(m,n)\cos\omega_1 m \cos\omega_2 n \quad . \qquad (2.62)$$

On quantization the coefficients differ from the ideal by an error term e(m,n):

$$h^*(m,n) = h(m,n) + e(m,n) \tag{2.63}$$

where

$$-Q/2 \le e(m,n) \le Q/2 \ . \tag{2.64}$$

The error in the frequency response at (ω_1,ω_2) resulting from quantization is represented as

$$\begin{aligned} E_L(e^{j\omega_1},e^{j\omega_2}) &= H^*(e^{j\omega_1},e^{j\omega_2}) - H(e^{j\omega_1},e^{j\omega_2}) \\ &= e(0,0) + \sum_{m=1}^{N-1} e(m,0)\cos\omega_1 m + \sum_{n=1}^{N-1} e(0,n)\cos\omega_2 n + \\ &\quad \sum_{m=1}^{N-1}\sum_{n=1}^{N-1} 4\cos\omega_1 m \cos\omega_2 n \ . \end{aligned} \tag{2.65}$$

The upper bound on this error can be found by maximizing all of the components in (2.65) including $|e(n,m)|_{max} = Q/2$.

$$\begin{aligned} |E_L(e^{j\omega_1},e^{j\omega_2})| &\le (Q/2)[1 + \sum_{m=1}^{N-1} 2|\cos\omega_1 m| + \\ &\quad \sum_{n=1}^{N-1} 2|\cos\omega_2 n| + \sum_{m=1}^{N-1}\sum_{n=1}^{N-1} 4|\cos\omega_1 m \cos\omega_2 n| \ ] \\ &= Q(2N-1)^2/2 \ . \end{aligned} \tag{2.66}$$

This error bound is excessively pessimistic, reflecting only the extreme limit of the error distribution at the frequencies $\omega_1 = \omega_2 = n\pi$.

A more useful model for error response prediction can be formed by assuming statistical independence of each error coefficient. This assumption, coupled with the fact that the e(m,n) are uniformly distributed over a finite interval, satisfies the Lindeburg condition [47, pp. 262-264] of the central limit theorem causing E_L, a sum of many e(m,n), to be essentially Gaussian. Therefore a complete statistical description of E_L is possible through its mean (=0), and standard deviation. To find this standard deviation, first

evaluate the ensemble mean square of E_L :

$$\overline{E_L^2(e^{j\omega_1},e^{j\omega_2})} = (Q^2/12)[1 + 4\sum_{m=1}^{N-1}\cos^2\omega_1 m + 4\sum_{n=1}^{N-1}\cos^2\omega_2 n + 16\sum_{m=1}^{N-1}\sum_{n=1}^{N-1}\cos^2\omega_1 m\cos^2\omega_2 n] . \quad (2.67)$$

Defining:

$$\begin{aligned} W_N(\omega_1,\omega_2) &= (4N-3)^{-1}[1 + 4\sum_{m=1}^{N-1}\cos^2\omega_1 m + 4\sum_{n=1}^{N-1}\cos^2\omega_2 n + 16\sum_{m=1}^{N-1}\sum_{n=1}^{N-1}\cos^2\omega_1 m\cos^2\omega_2^2 n]^{1/2} \\ &= (4N-3)^{-1}[1 + 4\sum_{m=1}^{N-1}\cos^2\omega_1 m]^{1/2}[1 + 4\sum_{n=1}^{N-1}\cos^2\omega_2 n]^{1/2} \\ &= W_M(\omega_1)W_M(\omega_2), \qquad M = 2N - 1 \end{aligned} \quad (2.68)$$

where Chan and Rabiner have found

$$W_M(\omega) = [1/2 + (2M-1)^{-1}(\sin M\omega/\sin\omega - 1/2)]^{1/2} . \quad (2.69)$$

Substituting (2.68) into the square root of (2.67) provides the expression for the standard deviation of E_L :

$$\begin{aligned} \sigma_{EL}(\omega_1,\omega_2) &= [\overline{E_L^2(\omega_1,\omega_2)}]^{1/2} \\ &= (4N-3)QW_N(\omega_1,\omega_2)/(2\sqrt{3}) . \end{aligned} \quad (2.70)$$

$W_N(\omega_1,\omega_2)$ attains a maximum value of unity at $\omega_1 = \omega_2 = k\pi$, ie., multiples of half the sampling frequency. It can also be shown that

$$\lim_{N\to\infty} W_N(\omega_1,\omega_2) = 1/2, \qquad 0 < \omega_1, \omega_2 < \pi . \quad (2.71)$$

Therefore σ_{EL} is bounded by

$$\sigma_{EL} \le (4N-3)Q/(2\sqrt{3}), \quad (2.72)$$

but for large N,

$$\lim_{N\to\infty} \sigma_{EL}(\omega_1,\omega_2) = (4N-3)Q/(4\sqrt{3}), \qquad 0 < \omega_1, \omega_2 < \pi . \quad (2.73)$$

Since equation (2.72) represents a definite bound on σ_{EL} over all frequency bands, it will be used to estimate σ_{EL} in the remainder of this section. However, equation (2.73) shows that the results may be pessimistic by up to a factor of two in some bands of interest.

σ_{EL} represents an absolute measure of the error statistics that can be expected for an arbitrary filter quantized with level spacing Q. To be useful, this result must be formulated in terms of the error expected for a specific filter in a certain band of interest, b_k. Since Q will be seen to specify N for the $\nabla^2 g$ filter, the coefficient word size can be designed from this error bound.

Prior to quantization, the baseband response of the $\nabla^2 g$ filter peak scaled to unity magnitude (ie., $\nabla^2 g(0,0) = -1$) is given by

$$G''(\omega_1,\omega_2) = -\pi\sigma_f^4(\omega_1^2 + \omega_2^2)\exp[-(\omega_1^2 + \omega_2^2)\sigma_f^2/2]/T^2 \qquad (2.74)$$

This will represent the model for the ideal $\nabla^2 g$ frequency response. The error resulting from aliasing into the baseband represents the first deviation from the ideal response, $\overline{G''}$. The aliasing error in band, b_k, at frequency ω is defined by

$$\max_{\omega \in b_k} |\overline{G''}(e^{j\omega_1},e^{j\omega_2}) - G''(\omega_1,\omega_2)| = \delta_k \ . \qquad (2.75)$$

After quantization, the error in the resultant response, $\overline{G^{*''}}$, increases:

$$|\overline{G^{*"}}(e^{j\omega_1},e^{j\omega_2}) - G"(\omega_1,\omega_2)| \le |\overline{G^{*"}}(e^{j\omega_1},e^{j\omega_2}) - \overline{G"}(e^{j\omega_1},e^{j\omega_2})| + |\overline{G"}(e^{j\omega_1},e^{j\omega_2}) - G"(\omega_1,\omega_2)|$$

$$\le \max_{\omega\epsilon b_k}|E_L(e^{j\omega_1},e^{j\omega_2})| + \delta_k \quad . \tag{2.76}$$

Since E_L is Gaussian distributed, to 95.4% probability,

$$|E_L(e^{j\omega_1},e^{j\omega_2})| \le 2\sigma_{E_L} \le (4N-3)Q/\sqrt{3} \quad . \tag{2.77}$$

Given that $\nabla^2 g(m,n)$ is peak scaled to unity, with a t-bit word size, excluding sign,

$$Q = 2^{-t} \quad . \tag{2.78}$$

Therefore, on substituting (2.77) and (2.78) into (2.76), the total error including quantization is bounded to high probability by

$$|\overline{G}^{*"}(e^{j\omega_1},e^{j\omega_2}) - G"(\omega_1,\omega_2)| \le 2^{-t}(4N-3)/\sqrt{3} + \delta_k, \quad \omega\epsilon b_k \quad . \tag{2.79}$$

Define the maximum in-band error after quantization as

$$\epsilon_k = 2^{-t}(4N-3)/\sqrt{3} + \delta_k \quad , \tag{2.80}$$

the ideal sampled filter, $\overline{G"}$, must now remain within a tolerance of

$$|\overline{G"}(e^{j\omega_1},e^{j\omega_2}) - G"(\omega_1,\omega_2)| \le \epsilon_k - 2^{-t}(4N-3)/\sqrt{3} \tag{2.81}$$

if the quantized filter is to maintain an error bound of ϵ_k to $G"(\omega_1,\omega_2)$ in band b_k. This essentially provides a new δ_k about which to design the precisely sampled filter.

A more common representation of the response error is in terms of in-band rejections in decibels, defined for the

quantized and unquantized filters in b_k as,

$$DL^*_k = -20\log_{10}[\max|\overline{G^{*"}}(e^{j\omega_1},e^{j\omega_2}) - G"(\omega_1,\omega_2)|] \qquad (2.82a)$$

$$DL_k = -20\log_{10}[\max|\overline{G"}(e^{j\omega_1},e^{j\omega_2}) - G"(\omega_1,\omega_2)|] \ . \qquad (2.82b)$$

The quantized in-band rejection can now be predicted from the original in-band rejection, DL_k, and the quantization error by substituting (2.75) and (2.79) into (2.82a):

$$\begin{aligned} DL^*_k &\geq -20\log_{10}[\delta_k + 2^{-t}(4N-3)/\sqrt{3}] \\ &= -20\log_{10}[10^{-DL_k/20} + 2^{-t}(4N-3)/\sqrt{3}] \ . \end{aligned} \qquad (2.83)$$

The minimum bound on (2.83), DL^*m_k, for a given DL_k is established by the minimum word size

$$\begin{aligned} t_{min} &= -\log_2[\sqrt{3}(10^{-DL^*m_k/20} - 10^{-DL_k/20})/(4N-3)] \\ &= t_\infty - \log_2(1 - 10^{-\Delta_k/20}), \end{aligned} \qquad (2.84)$$

where

$$t_\infty = \{DL^*m_k + 20\log_{10}[(4N-3)/\sqrt{3}]\} \ / \ (20\log_{10}2) \qquad (2.85)$$

and

$$\Delta_k = DL_k - DL^*m_k \ .$$

t_∞ represents the lowest bound possible on the word size, attained when δ_k approaches zero (ie., $DL_k \rightarrow \infty$). Since the $\nabla^2 g$ filter is unbounded in the frequency domain, t_{min} serves as a more realistic estimate of the necessary word size, at least for small σ_f. For fixed N and Δ_k, t_{min} is seen from (2.84) and (2.85) to increase linearly with DL^*m_K at a rate of one bit per 6 db, representing a halving of the quantization-induced error. Likewise, it is also seen from Figure 2.6 that an increase in t_{min} of one bit over t_∞ produces a change in in-band rejection, Δ_K, of 6 db, ie., the total error is only double δ_K.

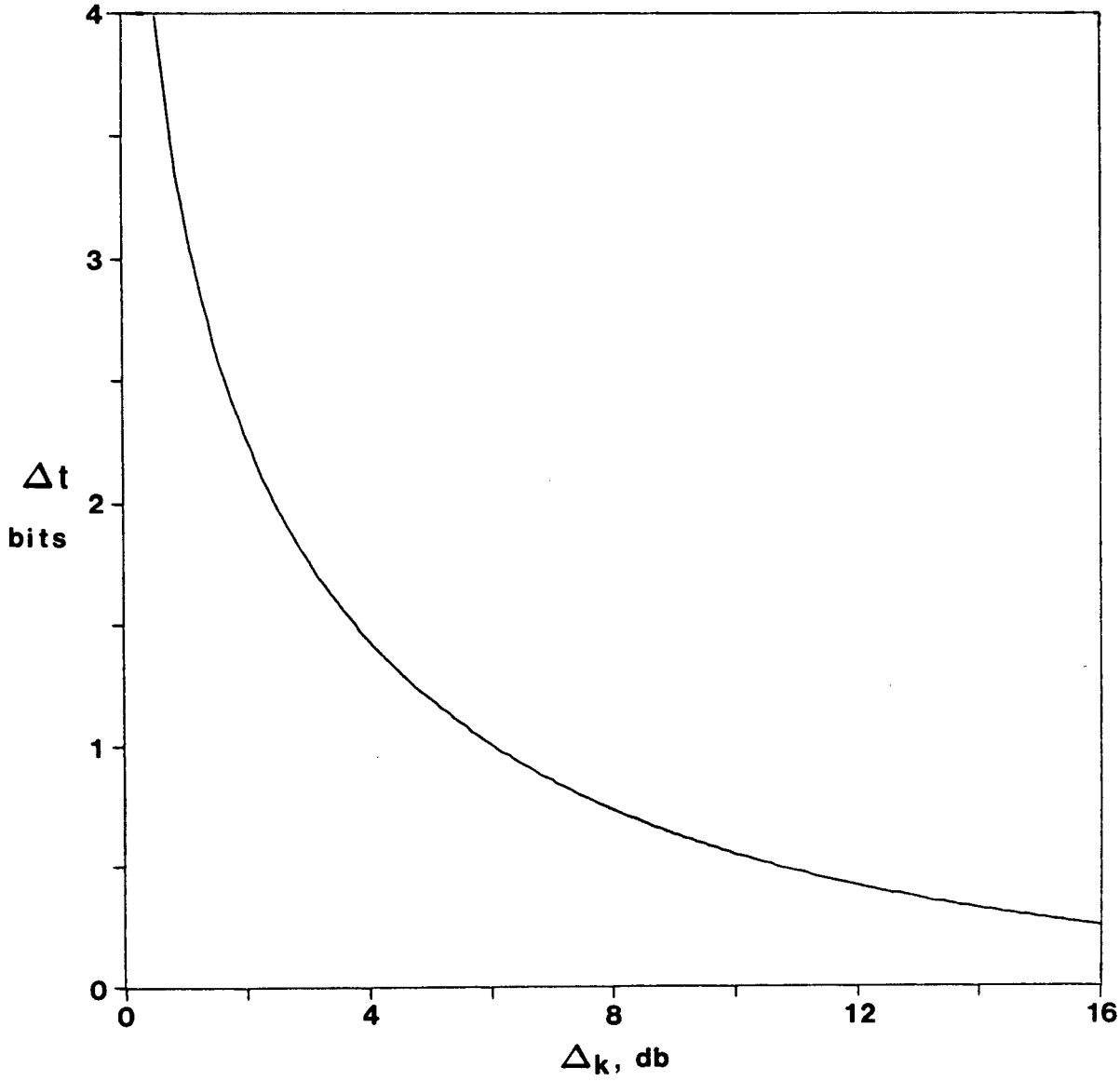


Figure 2.6 Additional bits, Δt, required to produce a given in-band rejection change, Δ_k, due to quantization

The filter implementation design path is now clear. From the selection of σ_f the maximum degree of aliasing in band b_K from the first harmonic determines DL_K. The acceptable decrease in in-band rejection for b_K also determines Δ_K. Finally, substitution into (2.84) yields the required word size to guarantee Δ_K to 95.4% probability. On the other hand, if the system available is of a fixed word size t, then it is easier simply to use equation (2.83) to determine if the resultant $DL*m_K$ is acceptable.

The design approach just outlined ignored one crucial parameter, the filter halfwidth N. The existence of N in the right-hand side of many of the equations in this section complicates matters somewhat. This is because N is dependent on both t and the filter quantization period $\delta\sigma_f$. The quantization period, in turn, also determines DL_K. The solution of equations (2.83) and (2.84) suddenly requires further knowledge of the filter. Chan and Rabiner recognized this difficulty and provided a method of approximating N for low-pass optimal filters. However, since the explicit point spread response of the $\nabla^2 g$ filter is known, N can readily be determined numerically.

In peak-normalized form:

$$\nabla^2 g(m,n) = -[1 - (m^2 + n^2)\delta^2/2]\exp[-(m^2 + n^2)\delta^2/2] \ . \quad (2.86)$$

Since the filter is symmetrical about (0,0), and we are interested in the extent along the axis, set m = 0:

$$\nabla^2 g(n) = -(1 - n^2\delta^2/2)\exp(-n^2\delta^2/2) \ . \quad (2.87)$$

Rounding of (2.86) will be used to determine the filter

coefficients. Therefore N is determined from the value of n where (2.87) first drops below 2^{-t-1} as $N = n_{max} + 1$. This point is most readily found by substituting $r = n^2\delta^2/2$ into (2.87) and solving

$$(1-r)\exp(-r) + 2^{-t-1} = 0 \; . \qquad (2.88)$$

Given t, r can be solved numerically. N being an integer, the truncated value of n_{max} found through (2.88) is used to give

$$N = \text{TRUNC}\,(\sqrt{2r}/\delta) + 1 \; . \qquad (2.89)$$

Table II lists the values of r and $\sqrt{2r}$ for t = 2 through 15 found by solving equation (2.88) by the Newton-Raphson method.

The curves of equation (2.89) plotted against δ for the four most popular word sizes of 4, 8, 12, and 16 bits, including sign, are shown in Figure 2.7. The results of Table II and equation (2.89) can now be used in the solution of equations (2.83) and (2.84).

Solving DL^*m_K in equation (2.83) is straightforward. Knowing t and δ, either refer to Table II and equation (2.89), or figure 2.7 for N. Substituting N, t and DL_K produces DL^*m_K. Some representative curves for DL_K = 20, 40, 60, and 80 are shown in Figure 2.8. The inverse process of solving for t_{min} given DL_K, DL^*m_K and δ is more complex but still straightforward.

This process involves two steps. First it must be recognized that t_{min} is a descrete integer variable and can therefore only change in discrete steps. The extreme minimum representation of t_{min} is t_∞ which therefore should also be represented as an integer. The consequence of these two observations is that $t_{min} - t_\infty = \Delta t$ can realisticly only change in

t	r	$\sqrt{2r}$
2	2.4532	2.2151
3	3.8034	2.7580
4	4.8010	3.0987
5	5.7082	3.3788
6	6.5693	3.6248
7	7.4017	3.8475
8	8.2144	4.0533
9	9.0125	4.2456
10	9.7993	4.4271
11	10.577	4.5994
12	11.348	4.7639
13	12.112	4.9217
14	12.871	5.0738
15	13.626	5.2204

Table II. r and $\sqrt{2r}$ against unsigned word size t

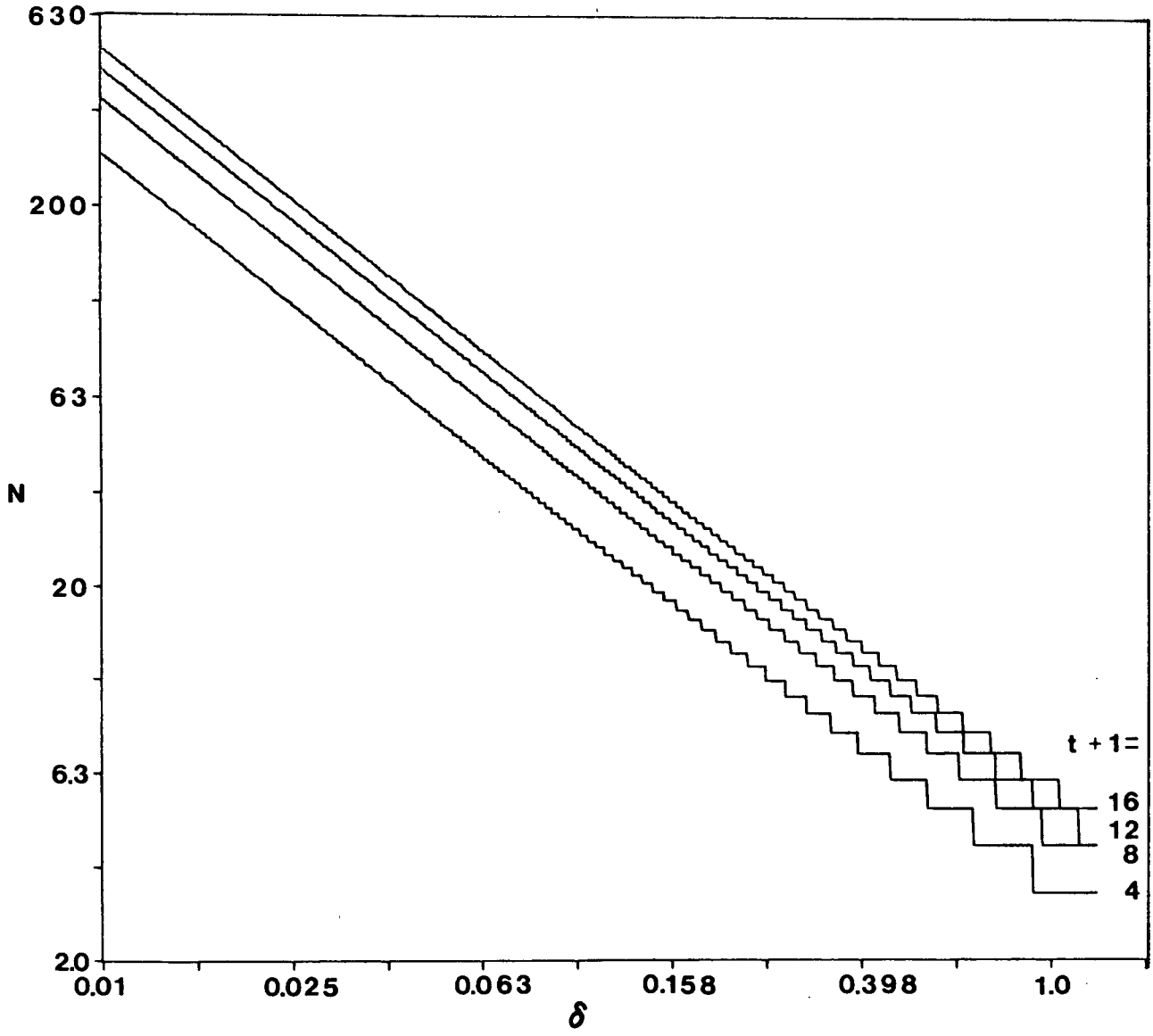


Figure 2.7 Filter half-width, N, resulting from normalized sample spacing δ, for 4, 8, 12, and 16 bit coefficient word sizes

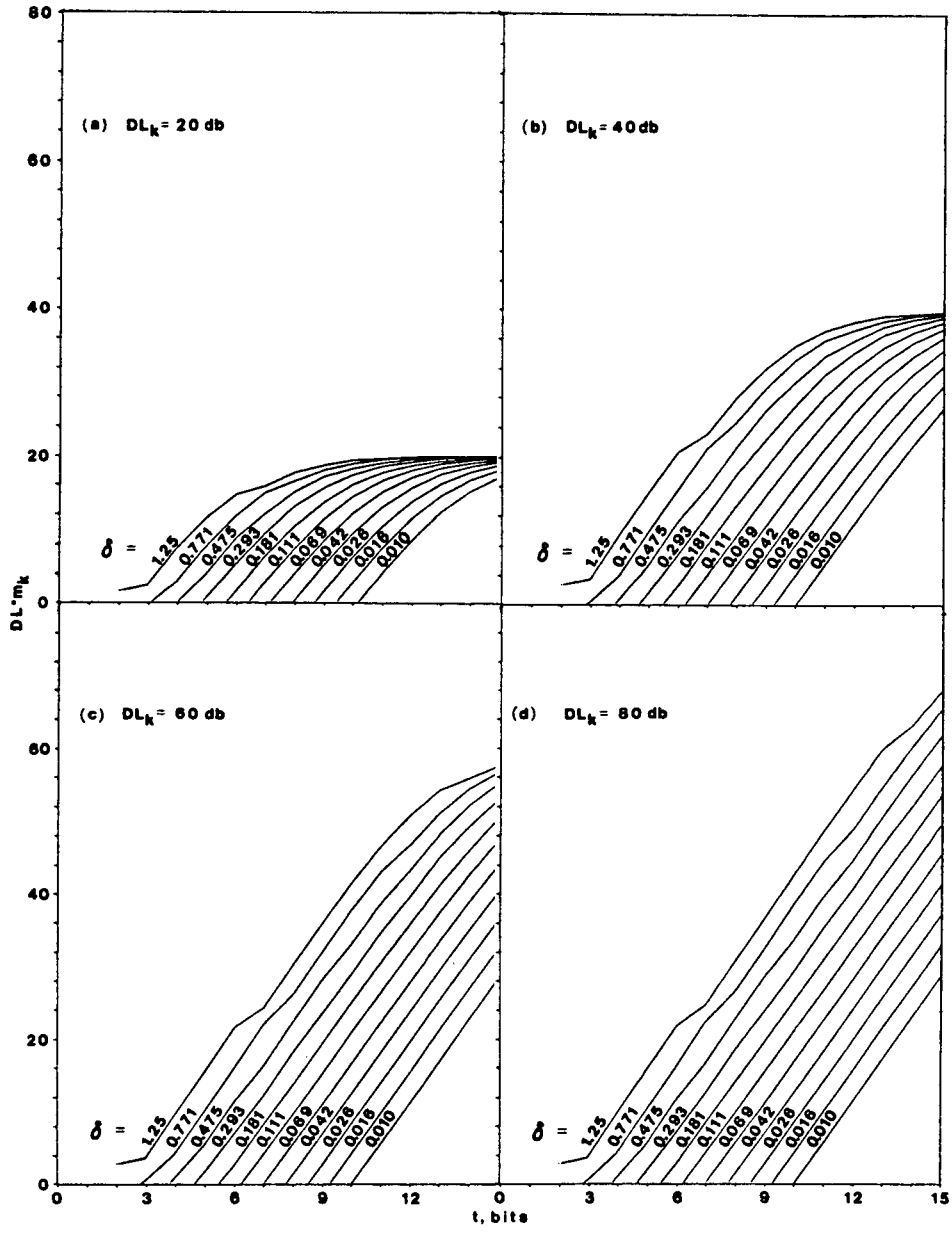


Figure 2.8 Quantized in-band rejection resulting from unsigned word size t for unquantized in-band rejections of (a) 20, (b) 40, (c) 60, and (d) 80 db

integer amounts. Therefore, the first step is, given DL_K and $DL*m_K$, calculate Δ_K then look up Δt on Figure 2.6, rounding up to the nearest integer unless Δ_K is so large as to be effectively infinite in which case $\Delta t \simeq 0$.

Equipped with Δt, the next step involves reference to an appropriate plot of equation (2.84). These plots appear in Figure 2.9 calculated by substituting the appropriate Δ_K into equation (2.84) for $\Delta t = 0, 1, 2, 3$ and 4. On choice of the appropriate plot, reference $DL*m_K$ to the abscissa, δ will select the appropriate curve from which t can then be read. For a fractional t round up to the next integer. The total word size, including sign, is then t + 1.

This indirect, graphical approach to solving t_{min} is necessary largely because N appears on the left-hand side of equation (2.84). Since N is a function of t, direct solution of (2.84) is not possible. Instead the approach chosen was solution of the inverse problem,

$$DL*m_K = 20\{(t_{min} - \Delta t)\log_{10}2 - \log_{10}[(4N-3)/\sqrt{3}]\} \qquad (2.90)$$

for the previous Δt and $t_{min} = 2,\ldots,15$. In this case t_{min} and N appear on the same side of the equation so solution of $DL*m_K$ is straightforward. These results were then plotted as shown in Figure 2.9, ignoring $DL*m_K < 0$.

Since a great deal of rounding of intermediate values is involved in this procedure, it may prove valuable to substitute the t_{min} found, and the other parameters into equation (2.83) as a check on $DL*m_K$. It may even be discovered that the t_{min} found was too conservative and may be reduced by a bit and still

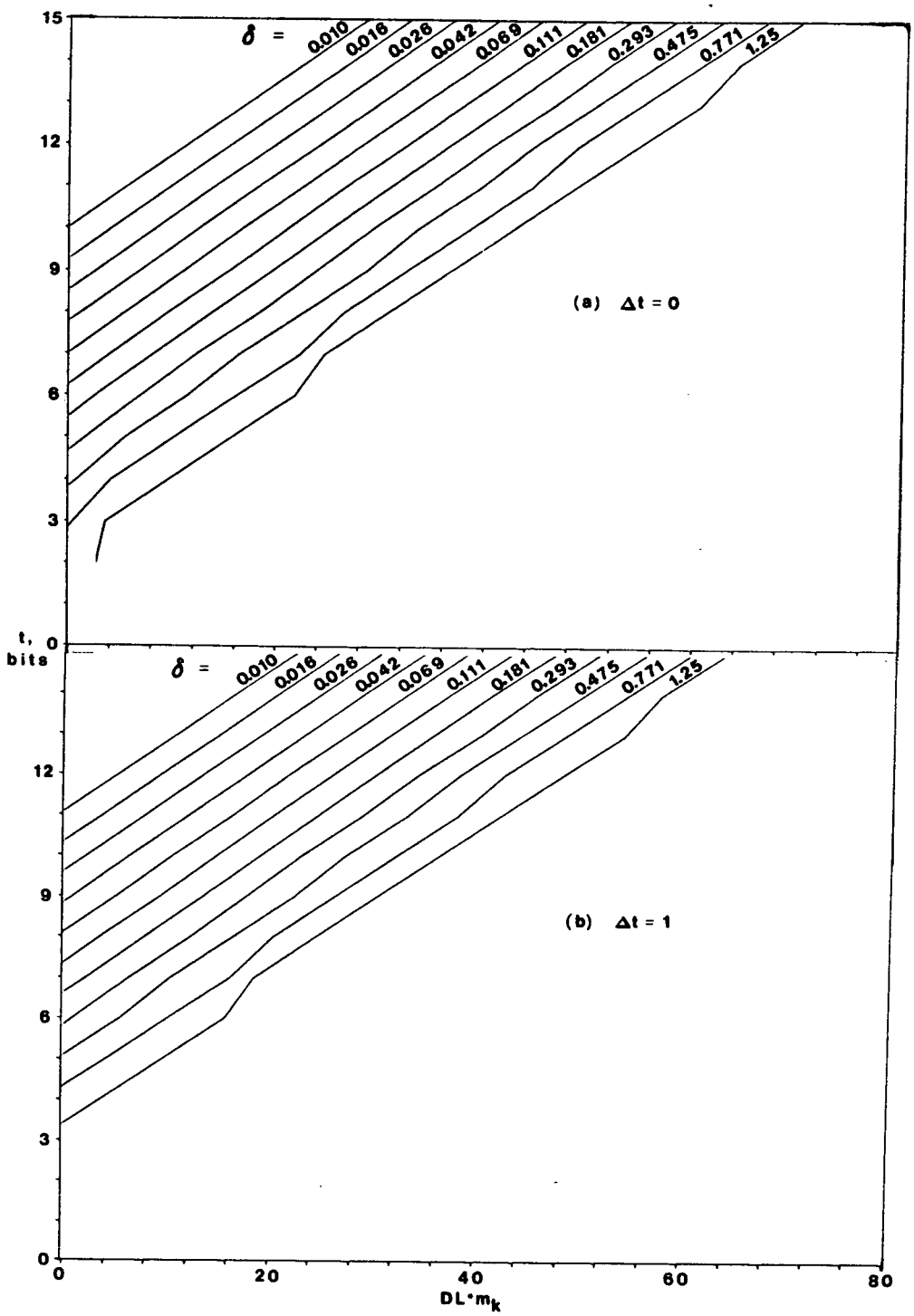


Figure 2.9 Minimum unsigned word size required given the quantized in-band rejection for Δt of (a) 0, (b) 1, (c) 2, (d) 3, and (e) 4

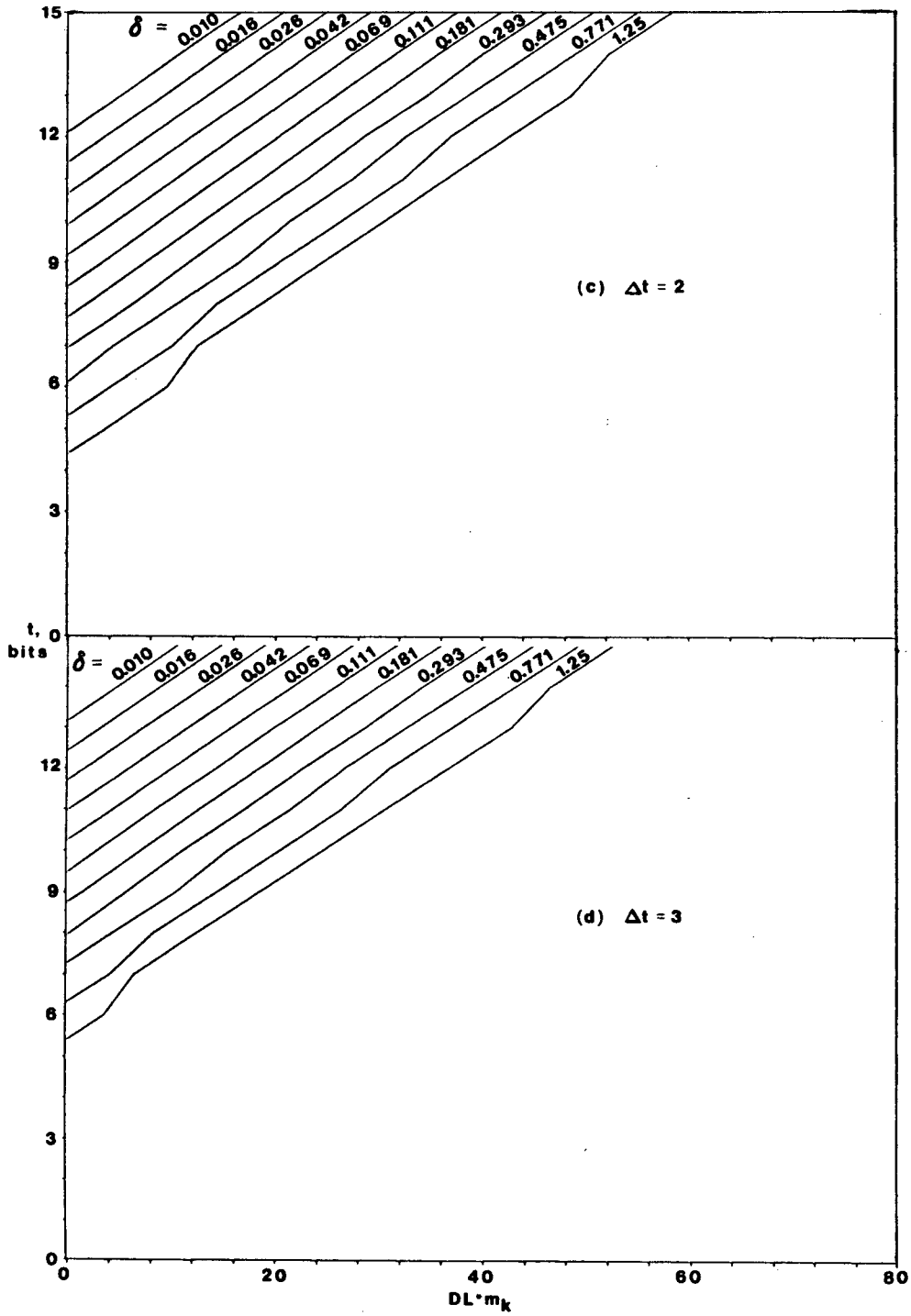


Figure 2.9 (continued)

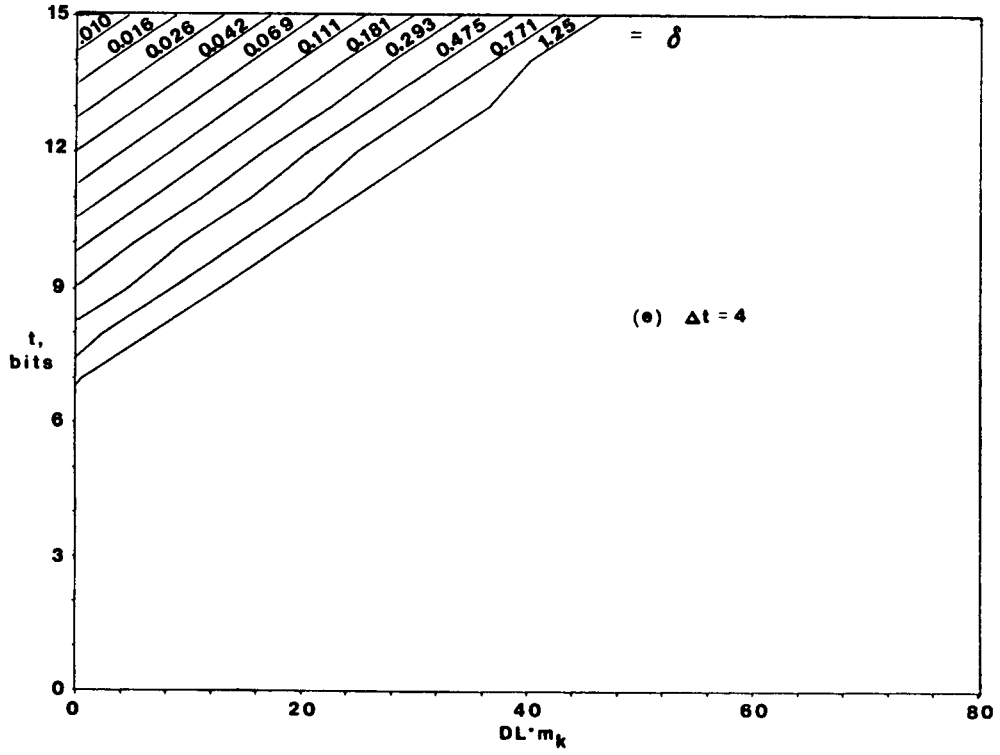


Figure 2.9 (continued)

maintain an acceptable in-band rejection.

2.8 Filter Design Example

We will now consider an example in the application of the previous techniques to the design of an appropriate filter word size. To choose an extreme case, it is desired that the unquantized in-band rejection be as small as possible. This implies a maximum of aliasing which was pointed out earlier to occur for the smallest σ_f chosen for the system. If this σ_f is chosen to produce a minimum of one pixel resolution then, with unit pixel spacing, it was found $\sigma_f = 0.8$. Therefore the pixel spacing is $1.25\sigma_f$ giving $\delta=1.25$.

From the aliased energy plot 2.7% of the baseband energy is aliased at this δ. To determine the resulting in-band rejection examine an arbitrarily narrow high pass-band, b_K, cut off at Ω. Being the highest frequency band of concern, it experiences the greatest degree of aliasing. Recall

$$\Omega = 2\pi(0.325)/\sigma_f \ .$$

To estimate the magnitude of the aliased response at Ω, consider only the contribution of the first harmonic along either axis. This contribution at Ω corresponds to the baseband response at

$$\omega = 2\pi(1 - 0.325/\sigma_f),$$

which when substituted into (2.74) $\omega_1=\omega$ and $\omega_2=0$ gives

$$\begin{aligned}\delta_K &= 4\pi^3[\sigma_f^2(1 - 0.325/\sigma_f)]^2\exp[-2\pi^2(\sigma_f - 0.325)^2] \\ &= 0.208\end{aligned}$$

for $\sigma_f=0.8$. Therefore, the unquantized in-band rejection is:

$$DL_K = -20\log_{10}(0.208) = 13.62 \text{ db.}$$

The ideal baseband response at Ω is

$$|G''(\Omega)| = 4\pi^3\sigma_f^2(0.325)^2\exp(-2\pi^2 0.325^2),$$
$$= 1.042.$$

DL_K, therefore, represents a 20% increase in the response magnitude at Ω over the ideal.

Since DL_K is already quite large, permit only another 5% increase in response magnitude, ie. 25% of the ideal, after quantization. Therefore

$$\overline{G^{*''}}(\Omega) \leq 1.303,$$

and $\epsilon_K \leq 0.261,$

giving $DL^*m_K = -20\log_{10}(0.261) = 11.67$ db.

Therefore, $\Delta_K = 13.62 - 11.67 = 1.95$

which must be rounded down to 1.16 to correspond to $\Delta t=3$. On referencing Figure 2.9 for $\Delta t=3$ and curve $\delta=1.25$, t for $DL^*m_K = 11.67$ is about 7.5 which must be rounded up to 8 bits. Therefore a total word size of 9 bits is necessary for at least 11.67 db in-band rejection at Ω for $\sigma_f = 0.8$.

From Table II and equation (2.89), N is found to be 4 causing the filter to occupy (2N-1) X (2N-1) = 7 X 7 pixels. To verify DL^*m_K, substitute DL_K, t=8, and N=4 into equation (2.83). The result is $DL^*m_K = 12.49$ db. Therefore, through these successive rounding operations, the predicted error has improved by 0.82 db.

A 9-bit word size is non-standard. If an 8-bit system were to be adopted instead, N would still be 4 but DL^*m_K would decrease to 11.48 db, for which $\epsilon_K = 0.267$. Therefore, the additional penalty of the lost bit is only 0.19 db, producing a

25.6% increase in response magnitude over G"(Ω).

n				
3	1	0	0	0
2	12	7	1	0
1	-13	15	7	0
0	-128	-13	12	1
	0	1	2	3 m

$$\times \frac{1}{128}$$

Figure 2.10 $\nabla^2 g(n,m)$ filter coefficients for sample spacing δ = 1.25 and 8-bit total word size

To test this prediction of ϵ_K for the 8-bit system, evaluate the response of the filter at Ω for t=7. One quadrant of the point spread response of $\nabla^2 g(m,n)$ from equation (2.86) for δ = 1.25 and truncated to seven bits plus sign is shown in figure 2.10. Substituting these values into equation (2.62) for $\omega_1=\Omega$ and $\omega_2=0$ gives

$$|\overline{G^{*"}}(e^{j\Omega}, e^{j0})| = 1.232.$$

The actual error, ϵ_K, is 0.190, or 14.42 db. Therefore, the 7-bit prediction is actually pessimistic by 2.94 db.

As a final exercise, let's compare the in-band rejections just found to those of the filter with the next octave lower passband, ie., σ_f =1.6, δ=0.625. The total 8-bit word size of the σ_f =0.8 filter will be retained, but now δ=1/1.6=0.625, which gives N=7. The magnitude of the aliased response at Ω is again found by substituting σ_f into (2.74), this time resulting in δ_K=5.983 X 10^{-12} for which DL_K = 224.5 db which for all intents and purposes here can be considered infinite. This reaffirms the earlier observation that aliasing effects for σ_f greater than unity can be ignored. The expected in-band rejection after quantization therefore simplifies to

$$DL^*m_K \simeq -20\log_{10}[\, 2^{-t}\ (4N-3)/\sqrt{3}\ ]$$

$$= 18.96 \text{ db}$$

corresponding to $\epsilon_K = 0.113$. Since the ideal magnitude response at Ω now is 4.169, ϵ_K represents a 2.7% response change at Ω. This represents a nearly one magnitude improvement in accuracy over the next highest octave case. Therefore, if the word size of the filter is designed to meet the tightest in-band rejection constraints for the narrowest filter of interest, it will meet at least those constraints for the other, wider filters also.

2.9 Test Image Examples

To provide examples of the design principles discussed, one of the tightly-controlled test images of Kitchen and Rosenfeld [48] were chosen. Kitchen and Rosenfeld had used two kinds of test images on which to evaluate various edge detectors: a series of concentric rings; and a single step edge. The rings image will be explored here since it is more complex and permits systematic study of the filters' behavior. The image was constructed originally as a 512 by 512 image consisting of only two levels of brightness: 115 (dark) and 140 (light). The image contains a dark circle of radius 64 at its center surrounded by six concentric rings of width 32 and alternating light and dark intensity. The impression is that of a "bull's eye" of dark background and center. The final image is 128 X 128, produced by replacing each 4 X 4 pixel block by a single pixel having the average grey level of the block. To include the effects of noise, independent zero mean Gaussian noise was added with a variance to produce a SNR of fifty. SNR is defined in the manner chosen by Kitchen and Rosenfeld:

$$SNR = h^2/\sigma_n^2$$

where h is the edge contrast, given here as 25.

The ring structure of this image has the advantage of being exactly periodic along the radius while at the same time being bounded in extent, and primarily two-dimensional in structure. This permits conclusions to be drawn concerning applicability of the previous unbounded one-dimensional models to more realistic two-dimensional structures. The primary periodicity of the test image is the edge spacing of the rings. In the 128 X 128 image, this period is eight pixels. The result of $\nabla^2 g$ filtering this image should be most closely predicted by the square wave edge model of edge spacing T = 8. If the edges were ideal steps, the filter just able to resolve them with maximum noise rejection would have a β of 1.15 resulting in a standard deviation of

$$\sigma_f = T/\beta = 6.96.$$

However, since the final image was reduced from a larger one by local averaging, there is some blur present. Though not Gaussian, this blur has a standard deviation of about 0.29. Clearly this is a great deal less than $0.51\sigma_f = 3.55$ for which seriously degraded performance is expected. However, in recognition of the overall non-ideal nature of the image, a somewhat more conservative β of 1.25 is chosen, producing $\sigma_f = 6.4$.

Reference to figure 2.2 shows that no response to a periodic structure is expected for $\beta=0.5$. Therefore, only the gross structure of the image is expected to be revealed, and little noise, on filtering with $\sigma_f = 16$. It is also seen that for minimal blur, the steady state response is established at $\beta=5$. A filter of $\sigma_f = 1.6$ is therefore expected to fully resolve the ring edges.

Furthermore, it is expected to do so to the highest precision but the poorest noise rejection of these three filter sizes.

The test image was therefore filtered by the peak scaled $\nabla^2 g$ filter with standard deviations of 16, 6.4, and 1.6. The results are shown in Figure 2.11. The filtering was done through fast convolution using a double precision floating-point 2-D FFT. The filter coefficients were also represented to double floating -point precision. The edge pixels are marked by the Freeman direction numbers [49] pointing in the direction of maximum gradient. The actual edge positions are chosen to coincide with the positive side of the zero crossing between horizontally or vertically adjacent pixels. This also corresponds to the borders of the test image dark regions.

From Figure 2.11a, it is readily apparent that only the gross detail of the test image is resolved. The fine structure of the rings is not detected, but the extreme inner and outer borders are. Since the central dark circle was 32 pixels wide, giving a β of 2, and the ring structure was 40 pixels wide, giving a β of 2.5, this behavior is expected. Note the complete absence of any extra noise detail. With a resolution interval of $2\sigma_f = 32$, the entire image lies within the overlapping resolution intervals of the edges seen resulting in perfect noise removal.

The filter σ_f of 6.4 predicts full resolution of the ring structure with maximum noise rejection. Figure 2.11b validates this claim. Note that there is no noise within the ring structure. Noise, however, does appear in the form of extraneous edge structures beyond the outer edge. The closest of these structures lies 11 pixels from the outer ring which is close to

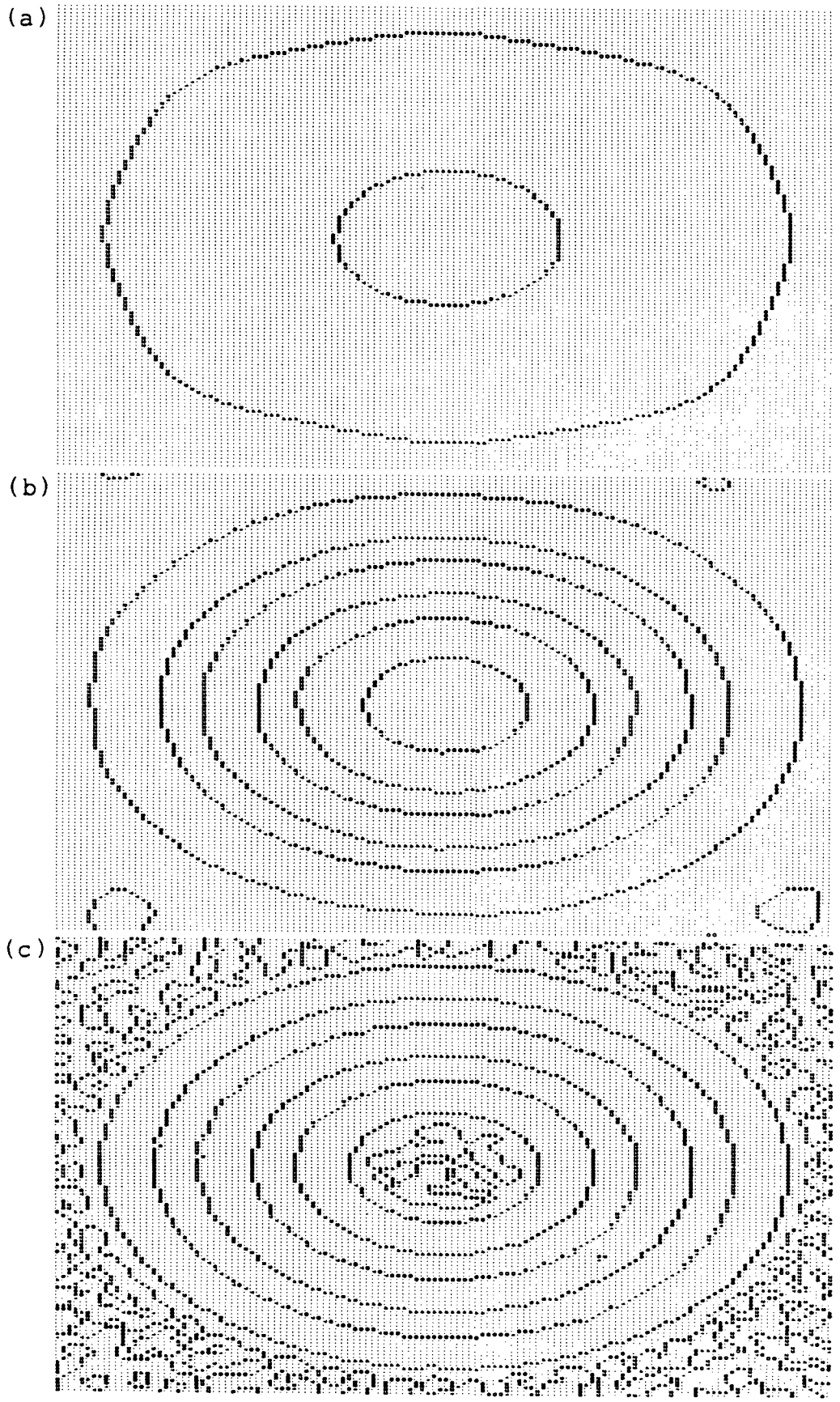


Figure 2.11 Ideal $\nabla^2 g$ filtered rings image with σ_f of (a) 16, (b) 6.4, and (c) 1.6

the $2\sigma_f$ noise free resolution interval predicted.

The σ_f =1.6 filtered image should resolve the test image but have poor noise rejection. Figure 2.11c indeed clearly shows the ring detail resolved as well as a great deal of noise. Most of the noise resides in the space beyond the outer and within the inner rings. Only one noise feature appears within the ring structure in the third ring. The closest approach of the noise edges to the ring edges is two pixels. This lies one third of the way within the expected 3.2 pixel resolution interval but exactly on the two pixel search interval to be used in multiband systems. However, it is seen that most noise structures remain beyond three pixels from the rings so the incidence of errors caused by unacceptably close approaches in multiband systems is expected to be low.

In general, the expected increase in edge position precision with decreasing σ_f is also observed. In fact, for σ_f =1.6, the dark image boundaries are located to the best precision that can be expected for a quantized image. The σ_f =6.4 image, though fully resolved, is however quite inaccurate, tending to expand the bright regions while contracting the dark ones. For instance, the central dark region has contracted by four pixels while the outer light ring has widened by three pixels. Interestingly, the two rings resolved in Figure 2.11a differ in accuracy. The outer border remains within 2.5 pixels of the correct position but the inner border is too wide by up to six pixels. Perhaps this behavior is attributable to the more gradual curvature of the outer ring which thereby more closely resembles a one-dimensional step. Also note that all edges form

closed curves. The only seeming violation of this occurs among the noise regions of Figure 2.11c. However, in actuality, this indicates the formation of one pixel wide dark regions after filtering. Since the edge pixels are associated with dark region boundaries, the result is apparently broken edges.

The previous results were generated with essentially infinite precision coefficients. Figures 2.12 and 2.13 show the results after the test image was filtered by the same set of filters, but with the coefficients quantized to eight and six bits respectively. The expected in-band rejection for these images at Ω are listed in Table III. When expressed in proportion to the ideal filter magnitude at Ω, it is seen that the change in response is expected to range from 0.26% for $\sigma_f = 16.0$ at eight bits up to 9.1% for $\sigma_f = 1.6$ at six bits.

In Figures 2.12a and 2.12b, a loss of precision in edge position is evident. However, the correct precision is re-established in Figure 2.12c for $\sigma_f = 1.6$, but there is increased noise present within the ring structure. Figure 2.12b shows the beginning of a phenomenon that has become well-established in the results of Figure 2.13. In the dark corners beyond the outer border, there run an extra set of edges to the image sides. If the image sides are considered circularly-connected, then these edges form a closed region of negative response. The reason for this region is the combined finite extent of the filter and the presence of a strong dc offset in the frequency response. The presence of this offset was predicted earlier in equation (2.70) where $W_N(\omega_1,\omega_2)$ has a peak magnitude at dc of twice the mean midband amplitude. The result of this offset is an expansion of

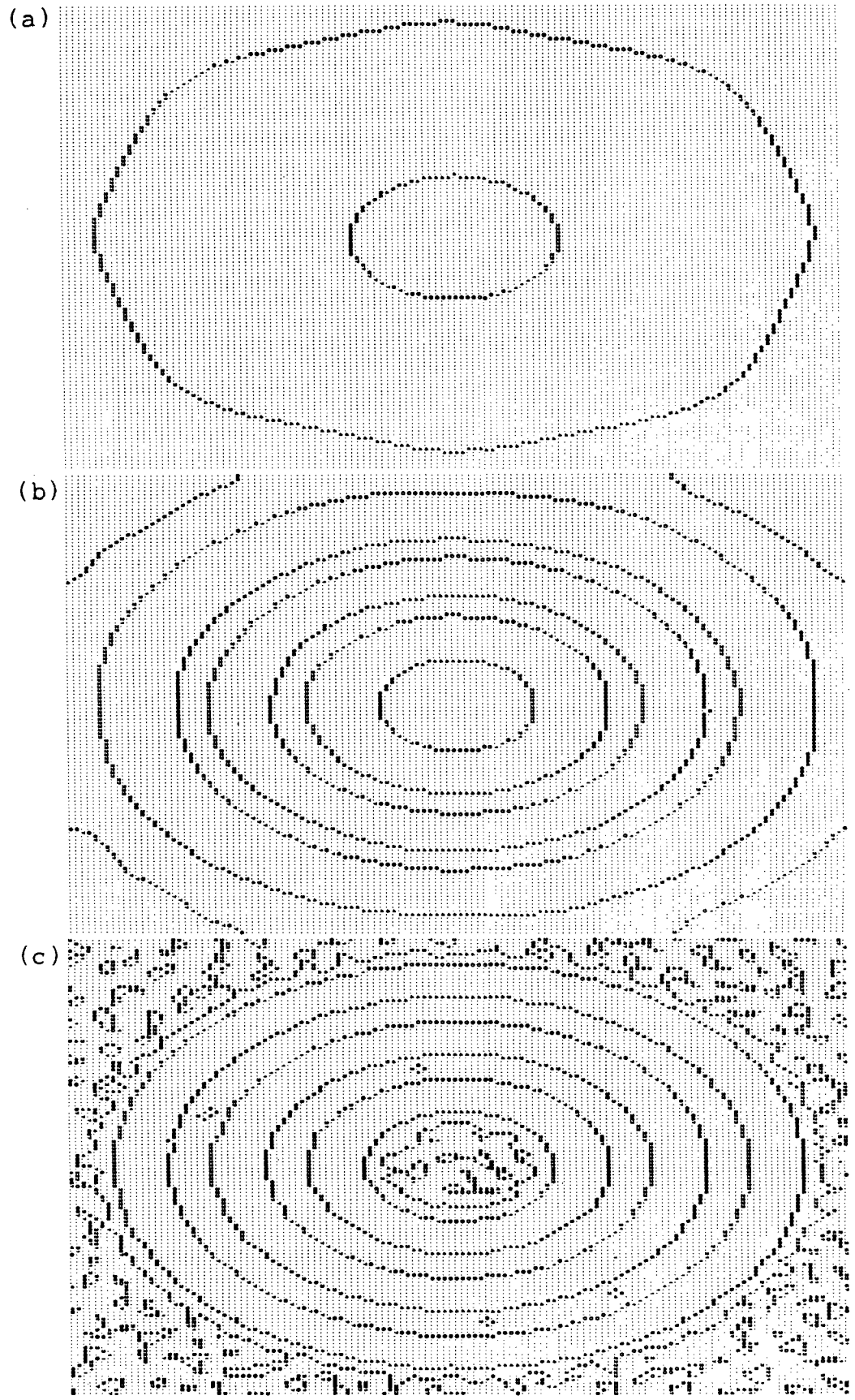


Figure 2.12 8-bit $\nabla^2 g$ filtered rings image with σ_f of (a) 16, (b) 6.4, and (c) 1.6

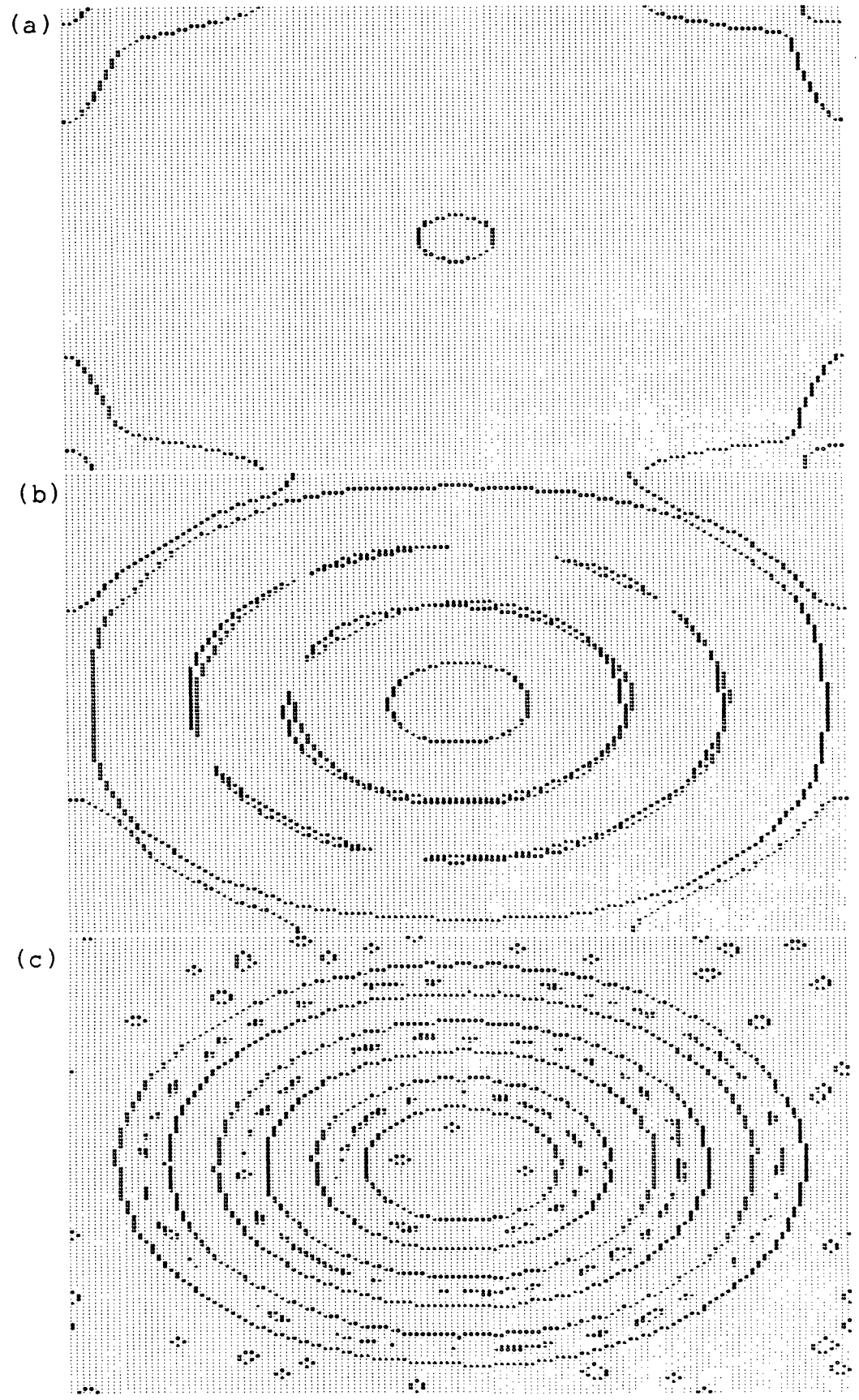


Figure 2.13 6-bit $\nabla^2 g$ filtered rings image with σ_f of (a) 16, (b) 6.4, and (c) 1.6

σ_f	$\lvert G''(\omega)\rvert$ db	DL^*m_K		(DL^*m_K - $\lvert G''(\omega)\rvert$ db)	
		8-bit	6-bit	8-bit	6-bit
16	-52.4	-0.868	-11.85	51.5	40.6
6.4	-36.5	-7.18	-3.71	43.7	32.8
1.6	-12.4	-18.96	-8.43	31.4	20.8

Table III. Expected in-band rejections at Ω for 6- and 8-bit quantized $\nabla^2 g$ filters

the regions with the same sign as the offset, and the creation of islands of this sign in large regions normally of the opposite sign.

In each of the examples of Figure 2.13, the result of the offset and smaller filter size is more pronounced. The outer boundary of 2.13a is so distorted it merged with its circular images and no longer forms a ring about the center. Figure 2.13b no longer fully resolves the ring structure. Instead, many of the edges associated with dark rings have broken into narrow island crescents. Interestingly, the ring structure of Figure 2.13c is intact. However, the precision of the edge positions has evidently diminished somewhat, though in the mean they are correctly placed. It is also interesting to note that in Figure 2.12c and especially 2.13c there is a reduction in the number of noise edges over those in Figure 2.11c. Evidently the magnitude of the noise regions is less than that of the dc offset.

Since dc offset effects have dominated the quantized coefficient examples, little can be said of the effects of the decreased in-band rejection. The offset was therefore removed and the tests repeated. To remove the offset, it was first measured by summing all the quantized coefficients. The offset was then subtracted out of the filter by at most one unit per coefficient. This subtraction was begun at a filter radius where it was felt that deviation from a simple rounding process would have the least effect on the filter response. This radius corresponds to the location where the cross-section of the filter's point spread function has the greatest slope. Here a one unit change in that coefficient would generally still agree

with the filter magnitude within half a pixel of that point. For the two-dimensional $\nabla^2 g$ filter, the radius of greatest slope is $0.79\sigma_f$. The subtractions are performed alternately about the center of the filter along axis that are mutually perpendicular until the four axis at 22.5° to the horizontal and vertical are reached. If the offset is odd, one further subtraction is done on the central peak so that the others remain symmetrical about the origin. The radius at which the subtraction is performed is then alternately increased and decreased by one pixel, as the eight perpendicular axis are exhausted. Fortunately the amount of offset is much smaller than the filter area so this process does not greatly distort the filter.

The results after removal of the filter offset are shown in Figures 2.14 and 2.15. Note that qualitatively there is very little difference in the two sets of images. Even the degree of noise present is about the same. Also, the edge positions are indicated to comparable accuracy in both sets of images. Interestingly the edges are of about the same accuracy of those of the infinite precision coefficients of Figure 2.11. The $\sigma_f = 1.6$ images are almost identical in all cases. It can be concluded therefore that the decreased in-band rejections resulting from coefficient quantization right down to six bit video resolution does not have a significant effect on the resulting edge positions and noise rejection after the dc offset is removed.

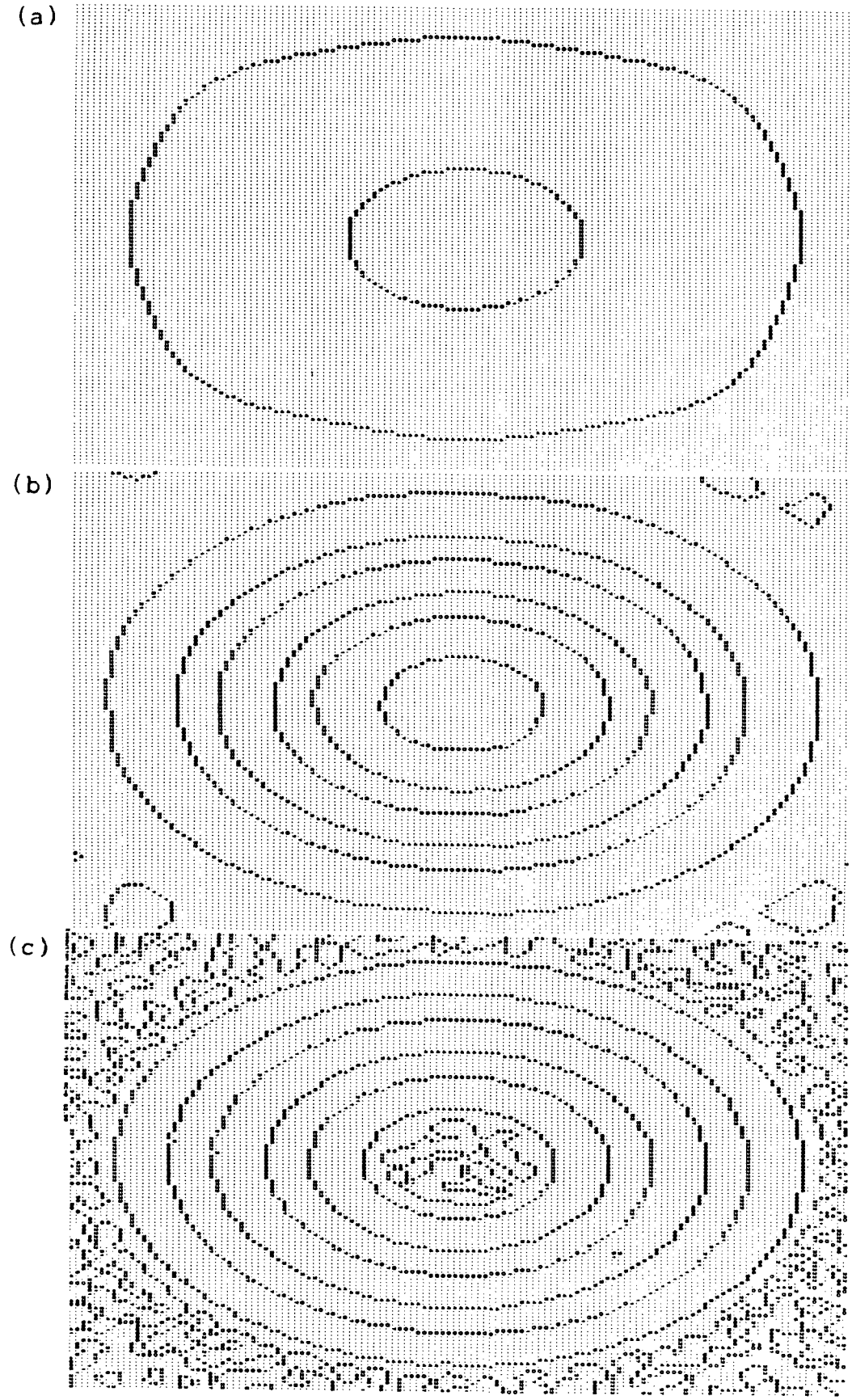


Figure 2.14 Unbiased 8-bit $\nabla^2 g$ filtered rings image with σ_f of (a) 16, (b) 6.4, and (c) 1.6

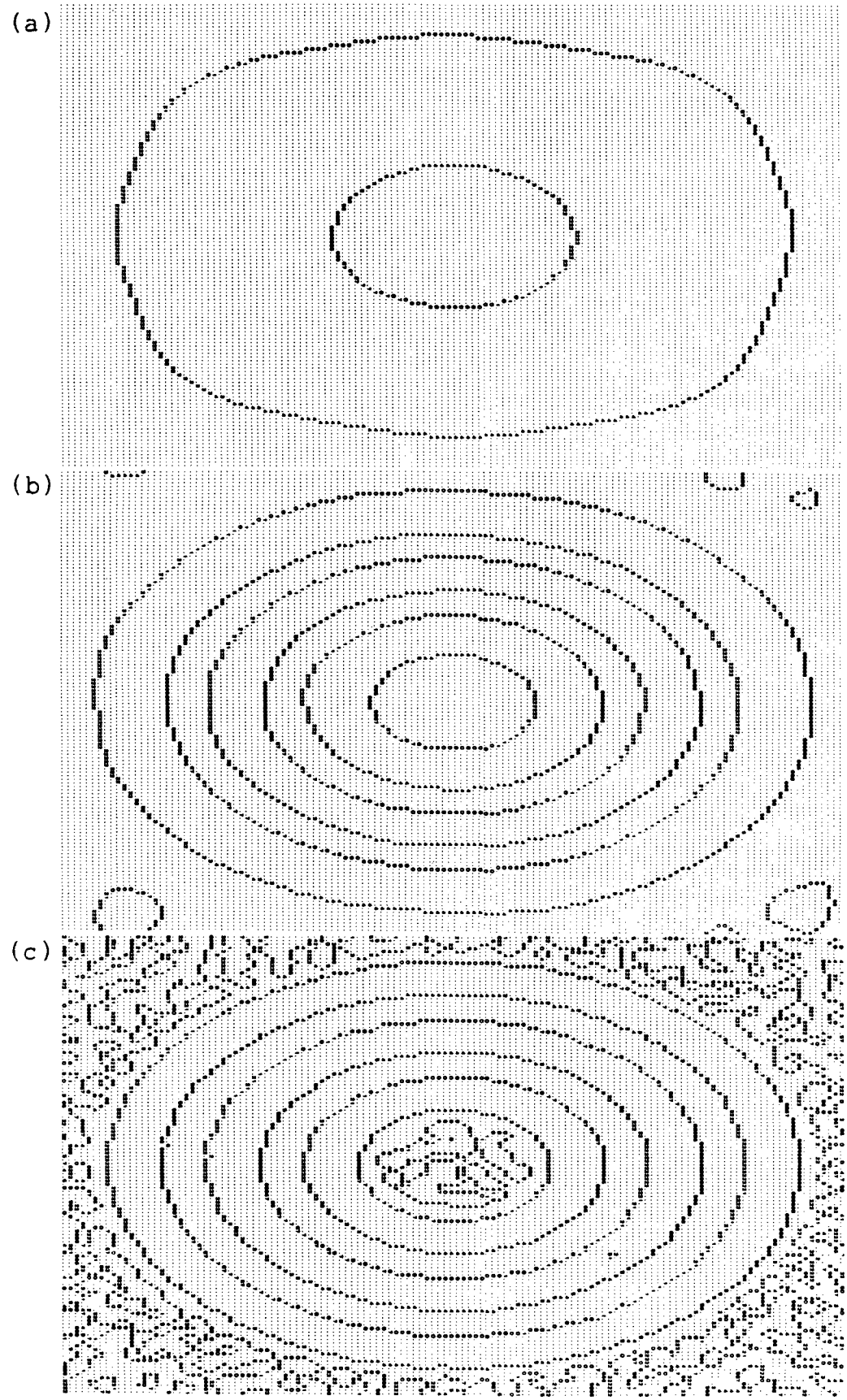


Figure 2.15 Unbiased 6-bit $\nabla^2 g$ filtered rings image with σ_f of (a) 16, (b) 6.4, and (c) 1.6

2.10 Design Summary and Conclusions

The foregoing analysis has established the dual optimality of the D. Marr $\nabla^2 g$ edge filter, and has provided a comprehensive design strategy for the selection of a filter standard deviation and coefficient word size appropriate for the edge detection task contemplated. The performance in additive noise was examined with the result that minimum input signal to noise ratio bounds were established. Also, it was seen that in order to resolve many levels of detail, or locate edges embedded in noise precisely, a multiband system must be employed where the individual passband spacing is at least one octave. It was found that the tightest contraints on the filter standard deviation, input signal to noise ratio, and quantized word size, apply only to the highest frequency passband. The remaining bands satisfy these constraints automatically.

To review, the filter design process follows a number of distinct stages. First it is important to identify the edge features to be resolved and to determine the scale of any periodicity in these features. Selection of the ideal edge model which most closely resembles the feature in question follows. Estimate the degree of blur present through its standard deviation, σ_b. This step may require some guesswork, but is necessary if the edges are to be resolved at the 3 db point of the σ_f chosen. The importance of resolving the edges at the filter 3 db point is a result of the region of maximum noise rejection coinciding with the resolution interval, which in turn increases directly with σ_f. Therefore, the greatest noise rejection follows from the selection of the widest allowable σ_f.

The selection of the filter variance on the basis of the above information is now possible. After reference to Figures 2.1 and 2.2 for the most suitable model with edge spacing T, the 3 db filter standard deviation falls in the range:

Staircase Edges - $T/5.5 \leq \sigma_f \leq T/2.75$;

Square Wave Edges - $T/1.36 \leq \sigma_f \leq T/1.15$;

The lower limits of all these models presume the presence of the maximum degree of blur of $0.51\sigma_f$ beyond which filter response remains below 3 db of steady state. Reference to Figures 2.1 and 2.2 shows that blur ceases to assert a significant influence below $0.2\sigma_f$. Therefore, if σ_f falls within $\sigma_b/0.51 \leq \sigma_f \leq \sigma_b/0.2$ a certain amount of iteration may be necessary before the appropriate σ_f is found.

On selection of σ_f a number of further constraints determine if it is practical. These depend largely on its relation to the sample spacing T , where, normalized, $T = \delta\sigma_f$. It was found that $\delta > 1.25$ is not practical on two grounds: the 3 db point then corresponds to edge structures that are undersampled; and the aliased filter energy exceeds 2.7%, becoming rapidly excessive. For $\delta < 1.0$ the aliased energy drops below 0.14%. Undersampling of the filter is therefore not a problem since it is unlikely that one would choose a σ_f matched to an undersampled image structure. However, δ does play a role in the minimum signal to noise ratio permissible in the original image. Derived on the basis of square wave edges, the permissible range of the signal to noise ratio was found to be $SNR_i > 2.66\delta$. This is a very liberal range, and since δ decreases with increasing σ_f this bound only constrains the narrowest filter in a multiband system.

Once σ_f has been selected the expected standard deviation of the zero crossing about the true edge can be determined if an estimate of the noise power or signal to noise ratio can be made. This accuracy is given by equations (2.37) and (2.51) which show that the standard deviation of the zero crossing increases as the square root of σ_f. To attain the accuracy associated with a narrow filter, but retain the noise rejection of a much wider filter it was suggested that a multiband system be used. Two such systems were proposed, one requiring explicit knowledge of the image signal to noise ratio, and the other, not. Since it is rare that the signal to noise ratio is always known in advance, and uniform for all the images to be processed by a given system, or even uniform across an image, the latter system is recommended. In such a system, the frequency bands are spaced one octave apart ranging from the largest details of interest down to the highest level of precision desired. Such an approach can also be used to classify the full range of detail present in an image according to the edge model 3 db point for each band. Multiband systems can also remove the pseudo-edge zero crossing which occurs between the true edges in the staircase edge model.

When final implementation into a hardware system is considered, the filter coefficients must be quantized. The influence of this was considered for the direct form finite impulse response implementation. The influence of quantization was shown to be a Gaussian error in the frequency reponse. The error standard deviation was not uniform across the spectrum but peaked in amplitude at integer multiples of half the sampling frequency. Twice the standard deviation of the peak error, in

db, was considered the minimum expected in-band rejection after quantization, $DL*m_K$. The quantization word size is determined, in large part, by fixing this quantity at an acceptable value in advance. Furthermore, selection of a frequency band, b_K, within the filter response where $DL*m_K$ is most tightly constrained is required. In the example given, this was chosen to be near Ω since the filter behavior is largely determined by the high frequency cutoff. Within b_K the ideal in-band rejection due solely to aliasing with infinite precision coefficients, DL_K, must also be calculated. It was shown through example that for δ less than 0.625, DL_K is so large as to be essentially infinite. With this information and the normalized sample period, δ, a simple two-step procedure was outlined for determining the minimum unsigned quantized coefficient word size, t_{min}. Since, of necessity, a certain amount of rounding takes place during this process, equation (2.83) can serve as a check on the resultant $DL*m_K$.

A series of test image results substantiated many of the ideas presented in this chapter. It was shown, for example, that the predicted 3 db σ_f for square wave edges matched to the difference in radii of a finite series of concentric rings will resolve such an image even though its only resemblance to a square wave lies in a portion of the cross-section along the radius. It was also seen that the accuracy of the zero crossing increases with decreasing σ_f while the noise rejection decreases. On quantization of the filter coefficients, the most notable effect was the appearance of a strong dc offset predicted by $W_N(\omega_1,\omega_2)$. The offset served to decrease the edge position accuracy, and, in some cases, to break the edges into fictitious

closed regions. After subtracting out the offset, the quantized filters performed almost as well as the precise filters, even at six bits. Only the noise rejection decreased slightly. An important observation of all the test images is the fact that all the edges formed closed regions without any spurious breaks. This was particularly evident when the filter was matched to image detail. This implies that there is no need to apply a further relaxation post-processing stage to repair edge segments.

When D. Marr and E. Hildreth first published the $\nabla^2 g$ edge detector, their arguments and examples made it clear that this should be the detector of choice for unambiguously resolving edge detail. However, no clear procedure was outlined for selecting a σ_f appropriate for the task at hand. It was felt that this was a major disincentive to applying the $\nabla^2 g$ filter in a general manner. This chapter's object was to remedy this situation by providing a systematic design methodology for the selection of σ_f, and final filter implementation. One remaining objection to the final implementation of the $\nabla^2 g$ filter is its size. At eight bits, the previous $\sigma_f = 6.4$ filter requires 49 X 49 pixels. However, for facilities not operating in real-time and performing fast convolution using a two-dimensional FFT, the filter size is no object. Where real-time video rate processing is required, new technology is rapidly providing the means for performing direct convolution using such large filters [50], [51]. Therefore, it is felt that the $\nabla^2 g$ edge detection filter will prove suitable for a wide variety of image analysis tasks.

III. OPTIMAL EDGE DETECTOR EVALUATION

3.1 Introduction

In the previous chapter the claim was made that the Marr and Hildreth $\nabla^2 g$ filter was more rigorously designed to detect edge features than most other edge detectors. It was also claimed that edge magnitude thresholding was an unreliable method of isolating preferred object edges from the background noise. If these claims are true then the $\nabla^2 g$ filter should outperform other detectors in a controlled comparision, but lose its advantage after thresholding. The object of this chapter is to perform this comparison and substantiate these claims.

To facilitate edge detector comparison a series of experiments must be designed incorporating a common set of test images and fixed evaluation measure(s) to be applied to all edge operators tested. Before embarking on these experiments one is faced with a choice: whether to design a new edge evaluation procedure, or whether to adopt a published one that has already proved successful. It was the latter course that was chosen here. Besides there existing an established base of procedures to choose from, this approach has the advantage that the literature would already contain the evaluation scores of most of the popular edge detectors, so this work need not be repeated. The evaluation procedure that will be the focus of this section is that developed by Kitchen and Rosenfeld [48]. Before presenting the reasons behind this choice, let us first review a number of other edge detector evaluation schemes published.

The earliest quantitative evaluation of edge detector performance was that of Fram and Deutsch [52], [53]. The test image consisted of a 36 X 36 pixel array containing feature regions. These were arranged as three vertical panels the right and left of which contained Gaussian intensity statistics of differing mean but a common standard deviation of 24. A center ramp, six columns wide and found by interpolating between the other two regions, constituted the edge feature. The Gaussian intensities were truncated to six bit precision. Ninety-seven such test images were generated by adopting ten different levels of contrast (differing means in the right and left panels), and approximately ten images per contrast level. After filtering, the resultant output was thresholded to produce a binary image which was then evaluated using two parameters, P_1 and P_2. P_1 was something of a signal to noise ratio involving the proportion of pixels generated due to the presence of the edge feature to a weighted sum of all the edge pixels found in the image. P_2 was a continuity measure representing the fraction of rows within the center panel which contained signal pixels. These evaluation scores range from zero for totally random edge features uniformly distributed over the output image to unity if all the pixels' output fall within the center panel with at least one pixel per row. Later, [53], the orientation bias of edge detectors was investigated by rotating the central edge panel 15°, 30°, 45° and 60° with respect to the vertical.

Only three edge operators were examined: the Hueckel, Macleod, and Rosenfeld-Thurston. The threshold for each of these operators was chosen heuristicly by inspecting a small sample of

outputs from the test images for each operator and then selecting a level at which the number of pixels above threshold produce a "well found edge". This is perhaps the greatest difficulty with the Fram and Deutsch approach. Though they claim their method is unbiased, this thresholding procedure does require a qualitative judgement by the user concerning acceptable edge quality which does not facilitate total automation of the procedure and makes comparison of the results between researchers difficult.

Abdou and Pratt [54] later developed a more comprehensive edge detector design and evaluation methodology. The edge evaluation component of their paper consisted of three parts: an edge detector sensitivity analysis, a comparison of the probabilities of correct and false edge detection, and a figure of merit computation. The sensitivity analysis involved a deterministic measurement of the edge filter output amplitude when applied to an ideal edge at differing orientation and locations with respect to the filter center. Ideally an edge operator should show no orientation bias and a rapidly declining response with displacement from center. The above analysis was performed in the absence of noise. The conditional probabilities for correct and false edge detection were evaluated by assuming an ideal step edge corrupted with additive zero mean white Gaussian noise. The results were also used to formulate a Bayes minimum error decision rule for the selection of threshold to maximize the probability of making a correct decision as to whether a given output pixel constitutes an edge or not.

The evaluation procedure of principle interest here however is the figure of merit comparison. To perform this evaluation

the edge operator is applied to an N X N test image containing a vertical step of fixed height h smoothed to a ramp by inclusion of a single column of height h/2 at the step, and corrupted by the addition of white Gaussian noise of varying standard deviation, σ. The output is thresholded to produce a binary edge map. The figure of merit is defined as

$$F = \max\{ I_I , I_A \}^{-1} \sum_{i=1}^{I_A} [1 + a\, d^2(i)]^{-1}$$

where I_I and I_A are the ideal and actual edge points, d(i) is the distance of the ith edge pixel detected to the ideal edge position, and a is a scaling constant emperically set to 1/9. Clearly if all the image edges fall only on the ideal edge position F=1. This technique was also extended to diagonal edges. The intent of this figure of merit is similar to P_1 of Fram and Deutsch which is to estimate the proportion of signal pixels present near the edge. However, d(i) adds a more explicit goodness of fit aspect to F, than the fixed panel width of P_1.

The class of edge operators examined was the enhancement/ thresholding type which included the differential operators of Roberts [18], Prewitt [55] and Sobel [56, p.271], and the template operators of Prewitt [55] (compass gradient), Kirsch [57] and 3- and 5-level template masks. The figure of merit evaluations were plotted against the signal to noise ratio which was defined as

$$SNR = (h / \sigma)^2 . \qquad (3.1)$$

These exact same operators were also evaluated by Kitchen and Rosenfeld who, however, adopt a very different figure of merit.

The principal drawback of the Abdou and Pratt evaluation measures is that edge continuity is never considered. Thus there is no analogy to P_2. In fact, Peli and Malah [58] report that F can produce a higher score for broken, thick edges than for perfect but thick edges. They claim that the problem is that F does not take the distribution of edge points along the edge into account.

Peli and Malah have also undertaken a study of edge detector evaluation. Indeed their approach is the most comprehensive of those published, even though only a small subset of their results were presented. Two basic test images were used: a square, and a circle, both of grey-level 15 superimposed on a background of grey-level 0. The dimensions of the images were not given. Three methods were separately applied to corrupt the test image: the ideal intensity step between object and background was replaced by a five point ramp; binary "salt and pepper" noise was added at 5%, 10% and 20% probability to simulate texture; zero mean Gaussian noise of standard deviation 3, 6, 9, and 12 was added with the resultant intensities clipped at 0 and 15. The performance measures devised were grouped into two broad categories: quantitative, and qualitative. Though seven quantitative measures were considered the results of only two were presented: Abdou and Pratt's figure of merit; and the variance of a detected edge point from the ideal edge. The ideal edge position was defined to a single pixel coinciding with the ideal step discontinuity before smoothing to a ramp. The qualitative measures, on the other hand, involved such observations as the type of edge contour produced (perfect, broken, perfect but broken), single or double edge, and

distortion through shift of the edge. The qualitative measures succeed in filling the gap between rigid evaluation scores and the subjective impression of image quality. Edge detection operators were selected which did not require a prior knowledge of the image structure. On this basis, the Roberts [18], Halé [23] and Rosenfeld-Thurston [19] operators were chosen, though no system for thresholding the results was given.

Generally, the approach of Peli and Malah parallels and extends that of Abdou and Pratt. For instance, by considering a circular object the effects on the evaluation measure of edges at all possible orientations can be observed. Also by including a qualitative aspect to their evaluations, such observations as the poor continuity sensitivity of the Abdou and Pratt results could be noted. Regrettably they chose only one operator in common with that work. However the quantitative aspect of the Peli and Malah approach is also its greatest drawback. A fully automatic approach, amenable to comparison with similar work, is certainly preferable.

Shaw [59] presented a fairly simple edge detector evaluation. This included the change of output signal to noise ratio and change in edge orientation produced by a distorted image when compared to the ideal image. Compared to the other methods examined, this approach was not very useful. It provided little information on the goodness of fit or the continuity of the edges found.

Three basic failings in all of the above edge evaluation schemes were noted by Kitchen and Rosenfeld. With the exception

of Shaw, they all required prior knowledge of the true edge position. While this provides the opportunity to make definite statements concerning spatial precision, similar techniques are not applicable to real world images where the edge positions are unknown. Another failing, also noted by Mero and Vassy, is the general lack of a continuity measure. Edges that are fragmented but consistently displaced from the true edge, receive similar scores from Abdou and Pratt as perfectly continuous but similarly displaced edges. Finally, none of these schemes used consistency in the direction of the detected edges in their evaluation scores. Only Shaw noted edge directions, but only to compare the change in direction of edge segments between noisy and noise-free images. Ideally the edge gradient direction should be everywhere perpendicular to the edge and in a manner consistent with adjacent edge pixels.

To address these criticisms and the earlier ones concerning the undesirability of human intervention, Kitchen and Rosenfeld developed a fully automatic edge evaluation technique based on the idea of local edge coherence.

3.2 Kitchen-Rosenfeld Evaluation

The concept of local edge coherence is founded on the premise that ideally edge features should locally be line-like. Edge coherence has therefore two components: edge pixels should be adjacent and therefore connected; and they should be thin like a line, ideally one pixel wide. Kitchen and Rosenfeld therefore chose to incorporate these two components into one edge evaluation scheme. The first component, continuation, measures the degree to which adjacent pixels agree in their determination of the local edge direction. The thinness component simply measures the local edge density. No knowledge of "true" edge position is ever required.

This approach holds two attractions. In general terms, it permits adjusting edge operator parameters to optimize performance in images about which little is known. The key parameter of interest is often the proper edge detector threshold setting. Threshold selection is a major theme underlying this evaluation technique, and Kitchen and Rosenfeld propose that the optimal threshold setting is that which maximizes the edge evaluation measure. Of more specific interest in the evaluation of the $\nabla^2 g$ edge detector is the claim, frequently made in the previous chapter, that the $\nabla^2 g$ filter returns only perfectly thin and continuous edges. This edge evaluation approach is therefore well tuned to the strengths of the $\nabla^2 g$ filter, and therefore facilitates comparison with similar strengths in other filters. Also, because the $\nabla^2 g$ filter cleans noise to $2\sigma_f$ from the object edge, there is some concern that an evaluation measure, such as that of Abdou and Pratt, which scores any noise clutter found

near the true edge highly would penalize the $\nabla^2 g$ filter for its merits. The Kitchen-Rosenfeld scheme can show no such bias.

The Kitchen-Rosenfeld evaluation examines every 3 X 3 pixel neighbourhood in the thresholded edge filtered image. Each pixel in this image must contain information as to whether or not an edge is present and what direction the gradient of that edge has. This direction is signified by radians in $\pi/4$ increments. If the central pixel in the 3 X 3 neighbourhood is an edge pixel the eight pixels surrounding it are called the edge neighbourhood. Pixels in the edge neighbourhood are identified by their direction numbers relative to the central pixel. Figure 3.1 shows an edge neighbourhood and the pixel labeling system.

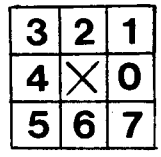


Figure 3.1 Edge pixel neighbourhood

Scores for continuation and thinness are first computed separately over this neighbourhood and then combined into a single evaluation measure whose value ranges from 0 for a poor edge to unity for a perfect edge. The continuation score is given by

$$C = (L(k)_{max} + R(k)_{max})/2 , \quad k = 0,1,...,7 . \qquad (3.2)$$

L(k) and R(k) measure how well the pixel at location k continues the edge to the left or right of the edge at the central pixel. These functions are zero if no edge is located at position k, and are given explicitly by:

$$L(k) = \begin{cases} a(d,d_k)a(\pi k/4, d+\pi/2), & \text{if } k \text{ is an edge pixel} \\ 0, & \text{otherwise;} \end{cases} \quad (3.3a)$$

$$R(k) = \begin{cases} a(d,d_k)a(\pi k/4, d-\pi/2), & \text{if } k \text{ is an edge pixel} \\ 0, & \text{otherwise;} \end{cases} \quad (3.3b)$$

where

$$a(\alpha,\beta) = (\pi - |\alpha - \beta|)/\pi,$$

and $d_0, d_1, \ldots, d_7$ = edge gradient direction at neighbour k;

d = edge gradient direction of the center.

$a(\alpha,\beta)$ measures the agreement between the angles α and β in terms of the minimum arc separating them. When they are equal, a=1; when they differ by π radians, a=0. Therefore, if neighbour pixel k has an edge direction identical to the central edge pixel then $a(d,d_k)$ will drive L(k) and R(k) to a perfect score. However, the premise that an edge should be locally line-like requires weighting neighbours perpendicular to the central edge higher than the rest. This weighting is provided by the $a(\pi k/4, d\pm\pi/2)$ term, where $\pi k/4$ is the direction to neighbour k and $d\pm\pi/2$ is the left- or right-ward direction perpendicular to the center. Only the three pixels to the immediate left and right of the central gradient direction are involved in the evaluation of L(k) and R(k) from which the maximum values, $L(k)_{max}$ and $R(k)_{max}$, are chosen.

The thinness score, T, is simply defined as the fraction of the pixels in the neighbourhood that are non-edge pixels. Since a perfectly thin edge has only two or less pixels adjacent to the center edge, this fraction is given by

$$T = \max\{\ 6,\ \text{no. of non-edge pixels}\ \} \ /\ 6\ . \qquad (3.4)$$

C and T are then combined into the evaluation measure for the central pixel:

$$E = \gamma C + (\ 1 - \gamma\ )\ T, \qquad (3.5)$$

where γ is a weighting coefficient adjusted to give E a suitable balance of thinness and coherence. Various values of γ ranging from zero to unity will be examined. Fortunately selection of γ is the only qualitative aspect to this evaluation process, and is performed after all the other results have been compiled.

Finally, E is averaged over all the edge pixels found in the image to provide an evaluation score for the image as a whole.

Three test images were explored by Kitchen and Rosenfeld. One was the "bull's eye" ring image already introduced in the last chapter, another was the single vertical step ramp image used by Abdou and Pratt, and the last was an image of pure noise. To review, the ring image was originally constructed in a 512 X 512 array. It consisted of a central dark circle of grey level 115 surrounded by a series of five rings alternating between light (grey-level 140) and dark (grey-level 115) of width 32 pixels on an overall dark background. The image was reduced to 128 X 128 by replacing each 4 X 4 pixel block with a single pixel of grey-level equal to that of the block. This image was proposed for study because its rings of differing radius provided an excellent means of subjecting the edge operators to edges in every conceivable orientation but which remain approximately locally line-like with similar contrast to the Abdou and Pratt

image (due to the block averaging).

The vertical step image consisted of a 64 X 64 pixel array where the left half was given grey-level 115 and the right half grey-level 140. To simulate the ramp discontinuity of Abdou and Pratt the single column at the junction of the discontinuity was given an intermediate grey-level of 128.

The final random noise image was also 64 X 64 pixels but contained only independent Gaussian random noise of mean 128 and standard deviation 16. This allowed inspection of edge detector performance in an image containing no well-formed edges, in a sense simulating some real world images.

To study the influence of noise, independent zero mean Gaussian noise was added to the rings (after reduction) and vertical step images. The standard deviation of the noise, σ, was adjusted to provide seven signal to noise ratios logarithmically spaced with values: 1, 2, 5, 10, 20, 50, and 100. Abdou and Pratt's signal to noise ratio definition, equation (3.1), was adopted where the intensity step height, h, is 25. However, note that this signal to noise ratio is exactly double that derived in the previous chapter where the signal power was shown to be $h^2/2$ for a single step image.

The exact same edge detection schemes used by Abdou and Pratt were investigated by Kitchen and Rosenfeld also. This permitted direct comparison of the results, and conclusions about the relative merits of the evaluation schemes to be drawn. Basically, their findings were in accordance with those of Abdou and Pratt in showing largely the same merit ordering of the edge

detectors tested. However, the Abdou and Pratt scores all show a convergence at high and low SNR, whereas the spread of the Kitchen-Rosenfeld scores tends to remain fairly constant. This seems to indicate that local edge coherence is a more individual attribute of an edge detector than the Abdou and Pratt figure of merit. In a sense this is not suprising since E is a structural measure monitoring, locally, how well-formed the edges are, whereas F is more of a statistical measure indicating an overall goodness of fit to the ideal edge.

An important quantity used throughout is the edge pixel fraction (epf) representing the proportion of the image pixels examined that are edges. This can serve as an indication of how well the test edges are resolved, and the degree of noise present. The ideal epf for the rings image is difficult to evaluate analytically. However, by counting the nearly perfectly resolved edge pixels of Figure 2.11c, the ideal epf can be estimated. The total ring circumference is 1218 pixels. With a total image area of 128^2 pixels the ideal epf is found to be 0.074. For the vertical step image the ideal epf is simply 64/(64X58) = 0.017 (excluding the six columns centered on the ideal edge at the image sides to remove its influence).

One drawback of the Kitchen-Rosenfeld approach is that it has an inherent bias against curved edges. This is a result of the premise that ideal edge features should be locally line-like. However, this premise acts against the rings test image reducing high scores when the test edges are perfectly resolved. To illustrate, consider the minimally curved neighbourhood of Figure 3.2 below.

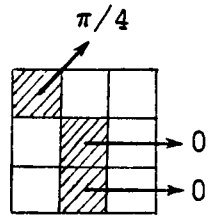


Figure 3.2 Minimally curved neighbourhood

The arrows indicate the edge gradient directions. The leftward pixel is at k=3 with direction $\pi/4$ gives leftward measure

$$L(k=3) = a(d,d_3)\ a(k\pi/4,d+\pi/2),$$
$$= a(0,\pi/4)\ a(3\pi/4,\pi/2),$$
$$= 0.75 \times 0.75 = 0.5625 .$$

The rightward pixel is at k=6 with direction 0 in agreement with the center pixel giving rightward measure

$$R(k=6) = a(d,d_6)\ a(k\pi/4,d-\pi/2),$$
$$= a(0,0)\ a(3\pi/2,-\pi/2),$$
$$= 1.0 \times 1.0 = 1.0 .$$

The continuation measure of this neighbourhood is therefore:

$$C = (L(k) + R(k))/2 = (0.5625 + 1.0)/2,$$
$$= 0.78125 .$$

Given a γ of 0.8 (the preferred value) and T=1 (this neighbourhood is perfectly thin), the final evaluation score for this neighbourhood would be E=0.825. The rings test image curves sufficiently gently so as not to be dominated by such curved neighbourhoods, but they do form a substantial fraction of the neighbourhoods present. This fact should be kept in mind when analyzing the rings image scores.

3.3 Evaluation Experiment Procedures

Since Kitchen and Rosenfeld published the complete evaluation scores of ten edge operators, little value was seen in reproducing all of these results for comparison to the $\nabla^2 g$ results. However, it was considered prudent to fully re-evaluate at least one of the given operators to act as a reference and indicate that the technique had been properly understood and coded. Only the three-level template matching operator could be chosen for this task since the complete set of evaluation experiments were only published for this operator. Kitchen and Rosenfeld designed these experiments in part to explore the process of threshold selection and to determine the ideal choice for γ.

The reproduction of the three-level template evaluation experiments produced results in excellent agreement with those published. It was therefore concluded that the implementation of the Kitchen-Rosenfeld evaluation method was essentially correct. Evaluation of the $\nabla^2 g$ filter could then proceed with confidence that the results could be compared directly to those published.

The Marr and Hildreth $\nabla^2 g$ filter was applied to the test images of the previous section through use of fast convolution via the two-dimensional fast Fourier transform. The filter coefficients were represented to double precision floating-point resolution across the entire image array. The filter standard deviations chosen were exactly those of the last chapter.

The $\nabla^2 g$ filter was designed for the detection of edges through the zero crossings it generates. However, this

evaluation scheme also requires that a magnitude and direction be assigned to each edge pixel. To fulfill this requirement, the conventions of the previous chapter were retained. The magnitude and direction of a zero-crossing segment was defined as the magnitude and direction of the gradient at the segment. Though this is a straight-forward concept in a continuous two-dimensional plane, in a discrete array certain special accomodations must be made. First, a zero crossing is a transition of intensity from positive to negative between horizontally and vertically adjacent pixels. As such, it is not inherently associated with any pixel in the output array. In lieu of creating another set of arrays of size 2N X 2N to explicitly represent the junctions between pixels in the output array (size N X N), a convention was adopted to assign edge magnitudes and directions to pixels of the output array. This convention was to assign the edge attributes to the pixel on the positive side of the zero crossing. The result of this is that edge information will be associated with the borders of dark objects in the input image.

Since a border pixel may have up to four junctions with the negative background, an unconventional approach to solving its gradient attributes was taken. Once such a border point was formed, the gradient was calculated in the following manner. First the slope of all the zero crossings adjacent to the border pixel was calculated using a simple two point difference of the intensities across the zero crossing. This calculation was chosen because of its simplicity, and because its frequency response reasonably approximates that of a derivative across the

$\nabla^2 g$ passband. If two junctions are found immediately adjacent in a clockwise or counter-clockwise sense, the magnitude of the diagonal gradient between them is estimated with the Pythagorean theorem. Finally the gradient attributes adopted by the border pixel will be that of the gradient calculation returning the greatest magnitude. This provides up to eight possible gradient directions which are identified in the subsequent output images by their Freeman direction numbers.

The above approach proves ambiguous only when very narrow object features, one pixel in width, are produced in the output image. In this case, the border pixels could be bounded by two or more legitimate edges of which only one can be recorded. As a consequence, the continuity measure can be expected to suffer. However, as was seen in the last chapter, such thin features only occur for very low standard deviation filters and so are not expected to be a general problem.

The choice of filter standard deviations, σ_f, will parallel that of the last chapter. The reasons are similar; to observe the influence of filters which are too large, properly matched, and too small. Therefore, the evaluation trials will be applied to the filtered rings images of Figure 2.11, where σ_f =16, 6.4, and 1.6.

For the vertical test image, only two standard deviatons were applied, σ_f =6.4, and 1.6. σ_f =16 was rejected since it is hopelessly too large (its zero crossing occurs at a radius of 32 and the image is only 64 X 64). Actually, a single step edge should be resolvable by any filter width. However, the expected

decrease in noise rejection with decreasing σ_f should be reflected in the evaluation scores. Because of the circular connection of the image borders by the convolution method used, the filter actually sees this image as an infinite series of vertical stripes of alternating intensity and edge spacing of T=32. The matched filter size for this spacing would be σ_f =25.6 which is even more impractical than σ_f =16. interestingly, the σ_f =6.4 filter applied to a square wave of T=32 has a β value of five which was shown to be the smallest β for which the filter can produce an ideal step edge response for this kind of image. σ_f =1.6 would correspond to a β of 20 and a resolution interval, I, of 6.4 indicating that most of the image is isolated from the optimum noise rejection influence of the test edges.

To maintain compatibility with the above experiments, the pure noise image was also filtered with σ_f =6.4, and 1.6.

The evaluation trials performed paralleled those published by Kitchen and Rosenfeld for the three-level template operator. This facilitates a detailed analysis of the continuity and thinness of the edges produced as well as the sensitivity to a changing signal to noise ratio.

3.4. Evaluation Results

3.4.1 Rings Image Evaluation

Figures 3.3 through 3.5 display the results derived from filtering the rings image at the three standard deviations. The test image has zero mean Gaussian noise added to produce an SNR of fifty. Analysis of the results follows; reference will be made to the unthresholded edge images of Figure 2.11, chapter 2.

σ_f =16:

The edges produced, seen in Figure 2.11a, correspond only to the extreme inner and outer boundaries of th test pattern. The detailed ring structure and noise are totally smoothed away. It is therefore expected that the edge pixel fraction always remain low, and that thinness scores perfectly.

The histogram of edge magnitudes, Figure 3.3a, indicates a nearly uniform distribution of pixels above 54% maximum grey level. The absence of low intensity pixels is consistent with the absence of noise induced edges.

Figure 3.3b shows that the evaluation measure below 54% threshold remains above, and approximately including, 0.9 for all γ. For γ=0, E remains constant at unity, which is consistent with the observed edge thinness. The maximum epf value of this region, Figure 3.3c, of 0.027, about one third of that expected, reflects the scarcity of edge detail. Above 54%, the remaining curves drop in a complex manner reflecting the loss of continuity on breakage of otherwise perfect edges. Note that the continuity-only curve, γ=1, peaking at 0.9, reflects the inherent

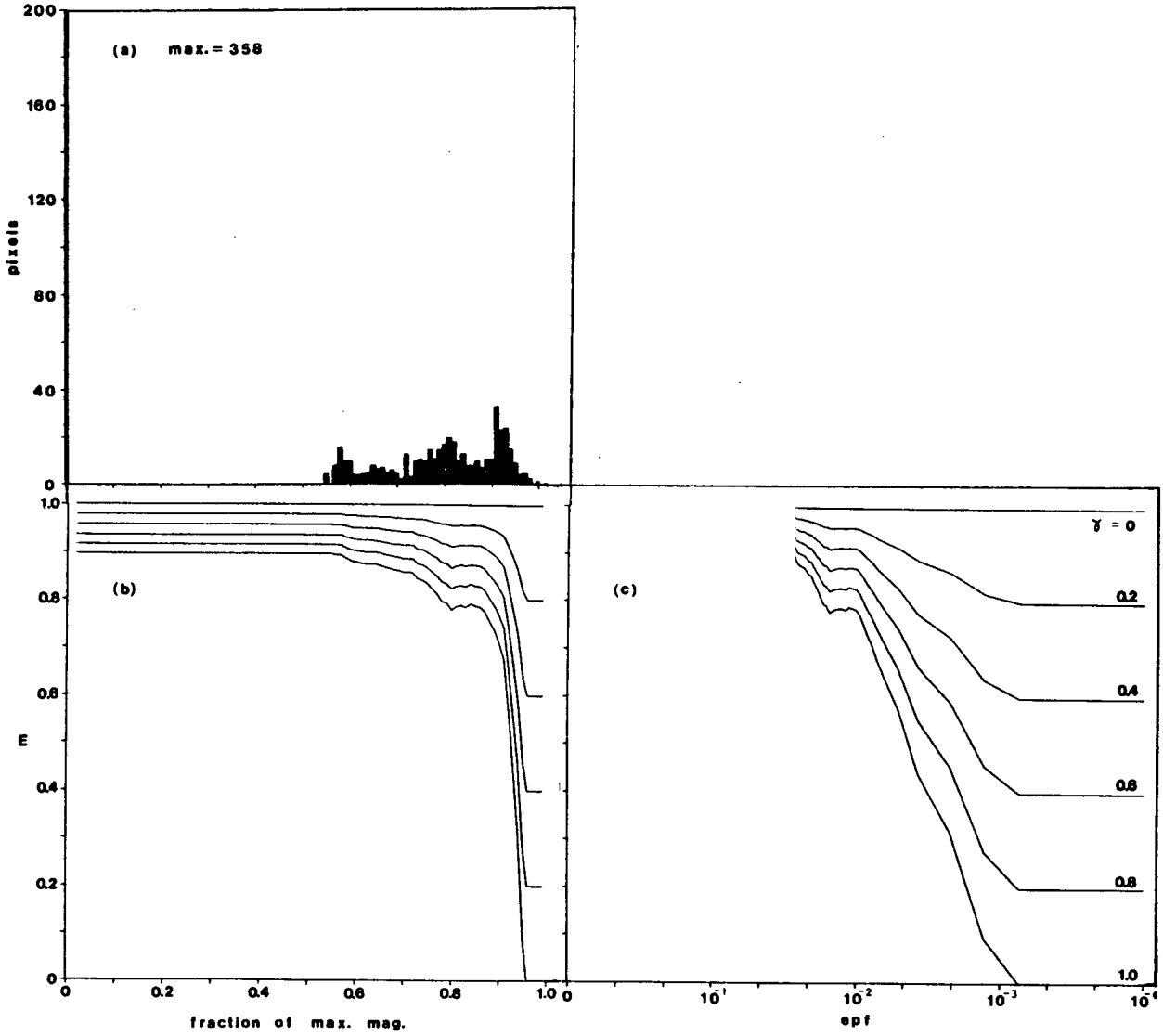


Figure 3.3 σ_f =16.0 $\nabla^2 g$ filtered rings image: (a) edge magnitude histogram; evaluation measure against (b) threshold level; (c) edge pixel fraction

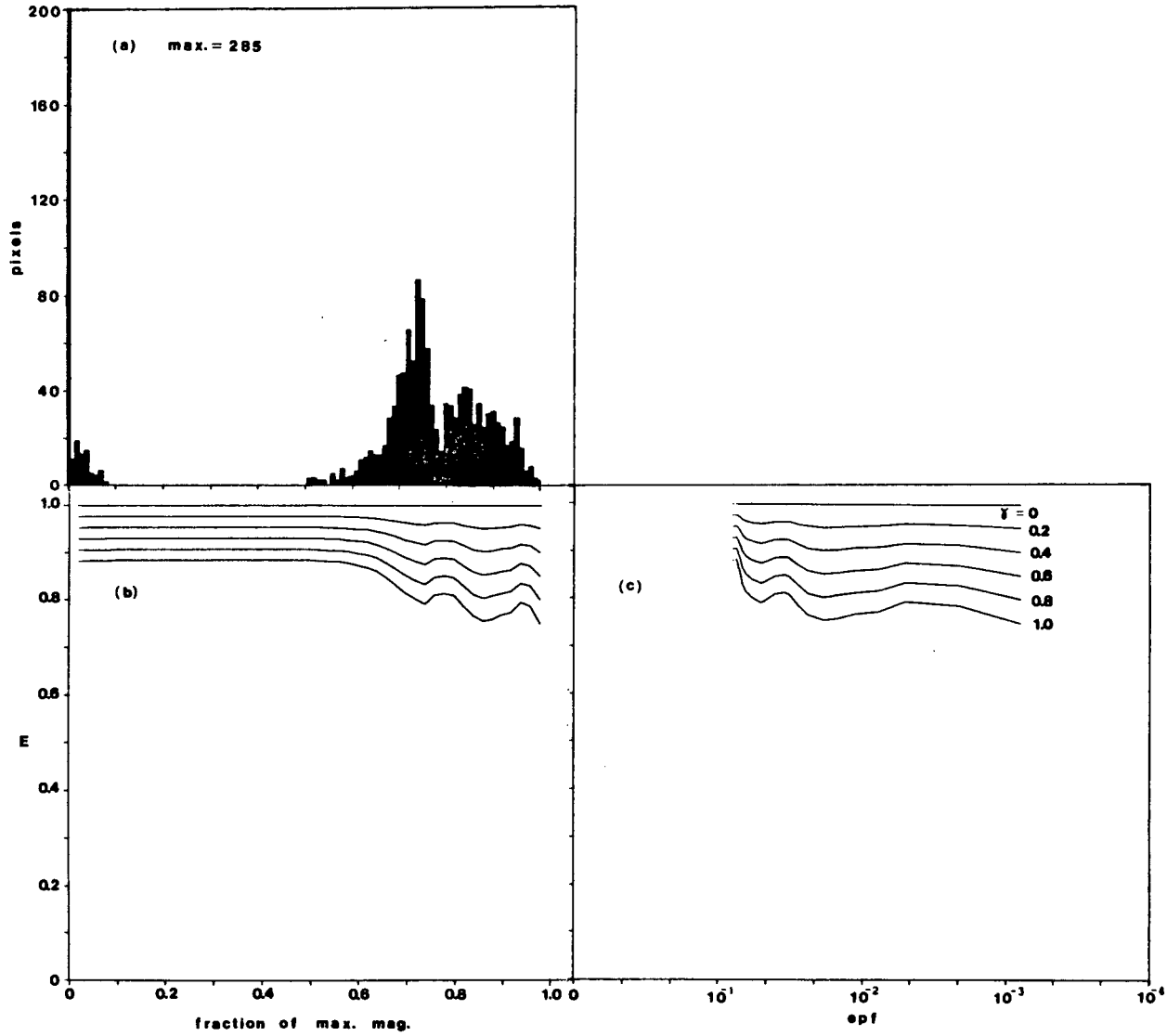


Figure 3.4 σ_c =6.4 $\nabla^2 g$ filtered rings image: (a) edge magnitude histogram; evaluation measure against (b) threshold level; (c) edge pixel fraction

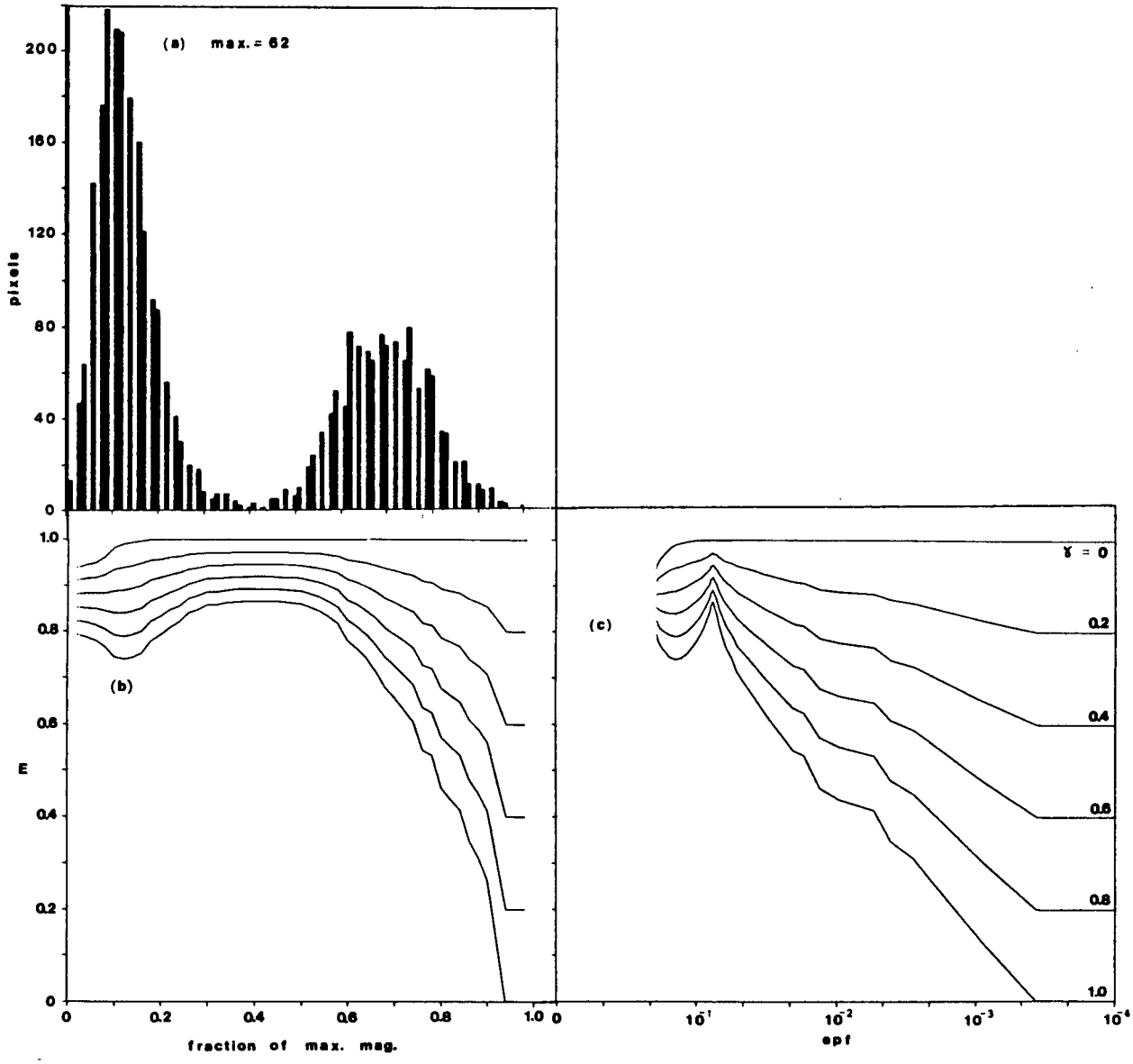


Figure 3.5 $\sigma_f = 1.6$ $\nabla^2 g$ filtered rings image: (a) edge magnitude histogram; evaluation measure against (b) threshold level; (c) edge pixel fraction

bias of this method against curved edges.

σ_f =6.4:

The edge image, Figure 2.11b, indicates that the full structure of the rings pattern was detected. Within the pattern, no noise is seen. A small amount of noise in the form of closed patches is seen in the image corners. All edges are perfectly thin.

The histogram, Figure 3.4a, reveals the beginnings of a bimodal distribution with an empty band between 9% and 50% of maximum level. The onset of the small mode below 9% is an indication of the presence of noise.

The evaluation scores of Figure 3.4b have spread slightly but still remain above 0.88 up to 56% threshold. The perfect thinness of the edges is again reflected in the constant unity value of the γ=0 curve. The noise seemed to have little effect on the low threshold regime of the evaluation measures. Finally, the epf value found just after thresholding out the noise was almost exactly that expected at 0.0741 despite the error in many of the edge positions.

σ_f =1.6:

The edge image of Figure 2.11c shows the full ring pattern resolved to high accuracy but embedded in a field of noise.

Reflecting the pervasive noise, the histogram, Figure 3.5a, has become strongly bimodal with the peak pixel count now in the noise mode at 9%. The many zeros in the histogram are a result of all the edge magnitudes having integer values between 0 and 62

which do not map to every level between 0 and 100%.

The evaluation scores of Figure 3.5b peak over the range of 38% to 44% threshold with γ=1 returning the lowest peak score of 0.865. Thinness is seen for the first time to drop below 1.0, below 18% threshold, attaining a minimum of 94%. Interestingly, the peaks coincide with the valley between the histogram mode a result consistent with that found in Kitchen-Rosenfeld's three-level template evaluations. Figure 3.5c dramatically illustrates this peak occuring at epf=0.075 just above that expected from the rings themselves. Clearly the conclusion here is the same as that of Kitchen and Rosenfeld: if the image must be thresholded, the optimal level coicides with the peak evaluations.

In general, it is observed that the evaluation scores at low threshold remain high for all γ in contrast to the divergence seen in the three-level template results. It can therefore be concluded that the $\nabla^2 g$ filter edges exhibit a local continuity quality comparable to their thinness.

The next series of tests compared the scores against threshold setting for seven differenct signal to noise ratios (1, 2, 5, 10, 20 50, 100). This required choosing a fixed γ setting. The setting chosen was γ=0.8, the same as that used by Kitchen and Rosenfeld. Compatibility with published results was a factor in this choice, but not a dominant one. The principal reason was that the edges were observed to be perfectly thin. The thinness component of the score therefore provided little new information and would tend to upwardly bias the continuity

component. Continuity, however, was of serious concern particularly for the applications envisioned which require continuous edges for region segmentation. The pure continuity score does, however, have a curved edge bias. A γ of 0.8 was therefore felt to compensate for the bias, acknowledge the general edge quality, and provide good indication of the degree of edge continuity.

SNR variation:

These images were only thresholded to the level where epf dropped below 0.02, at which point the test images are quite thoroughly broken.

Figure 3.6a shows the results for σ_f=16. The evaluation scores are seen to be largely SNR independent exhibiting similar shape and small vertical variation. Since the epf range was the same for all the curves, beginning at 0.027 the same level of edge detail was resolved at each SNR. This noise imunity is expected because the resolution interval, I, of 64 about the two resolved rings covers the entire image. The maximum evaluation measure was roughly constant at 0.925 and was attained at maximum epf. The max. E versus SNR curve of Figure 3.6b reiterates the noise immunity found.

The σ_f=6.4 results of Figure 3.7a are still seen to be largely SNR independent in both shape and vertical spread, though some divergence occurs at low epf. The increase in edge detail is responsible for increasing the epf range. The peak scores all occur about epf=0.74 produced by the rings. The max. E curve, Figure 3.7b, displays this SNR independence by remaining flat at

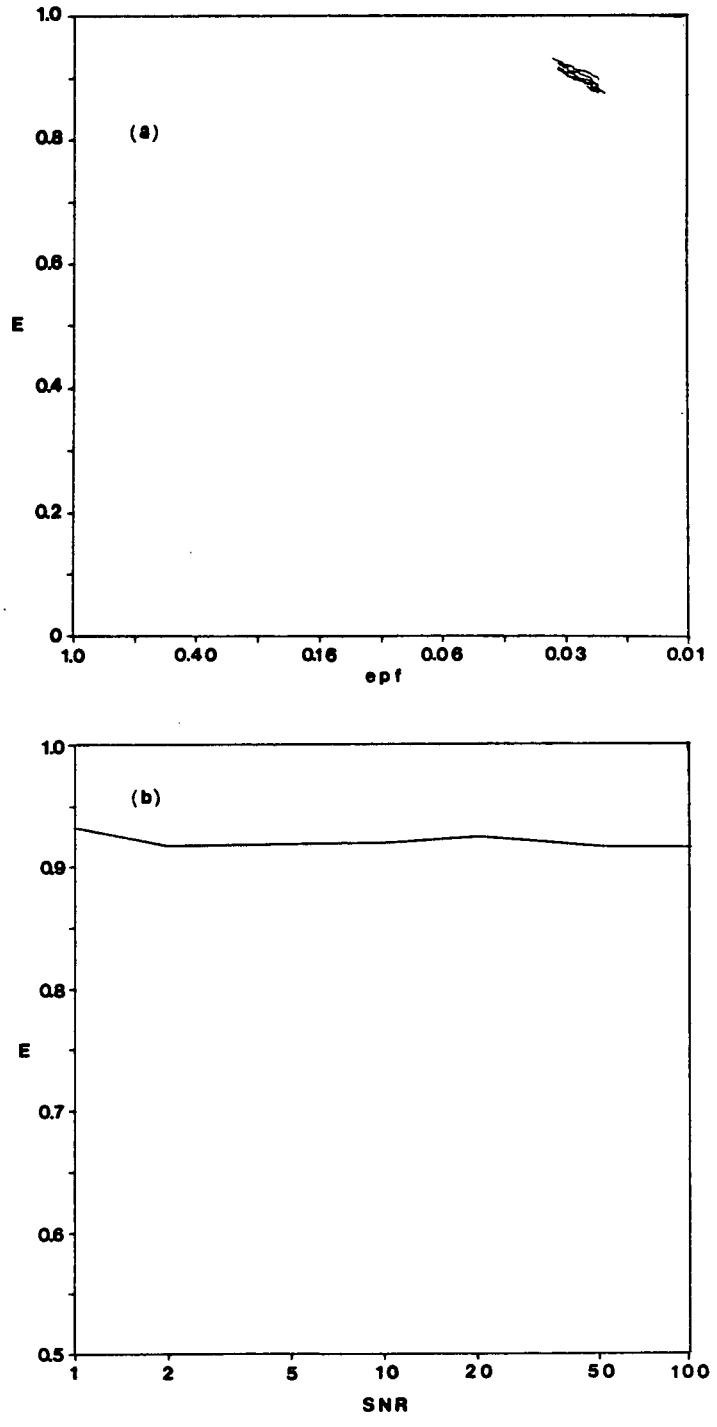


Figure 3.6 σ_f =16 $\nabla^2 g$ filtered rings image evaluation results: (a) SNR= 1, 2, 5, 10, 20, 50, and 100 (from bottom curve to top), δ=0.8; (b) peak evaluation scores against SNR

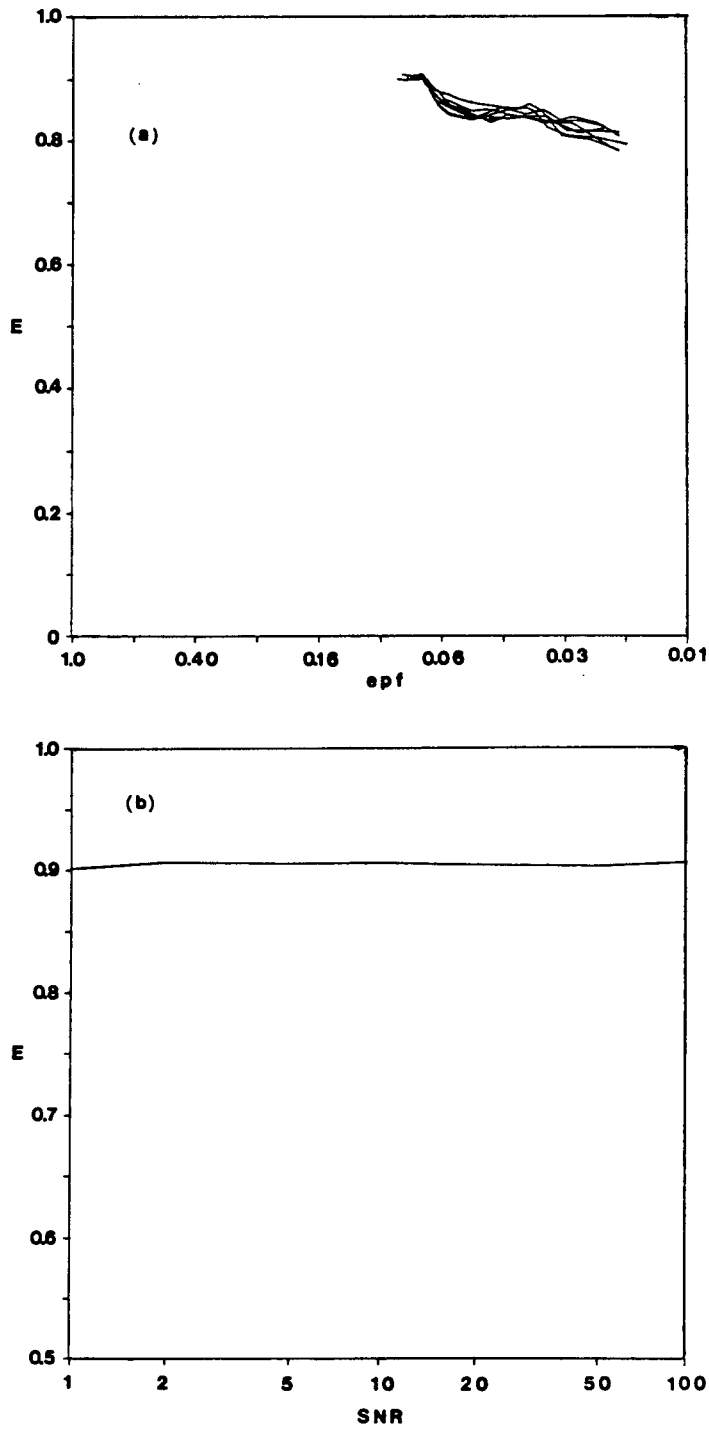


Figure 3.7 σ_f =6.4 $\nabla^2 g$ filtered rings image evaluation results: (a) SNR= 1, 2, 5, 10, 20, 50, and 100 (from bottom curve to top), δ=0.8; (b) peak evaluation scores against SNR

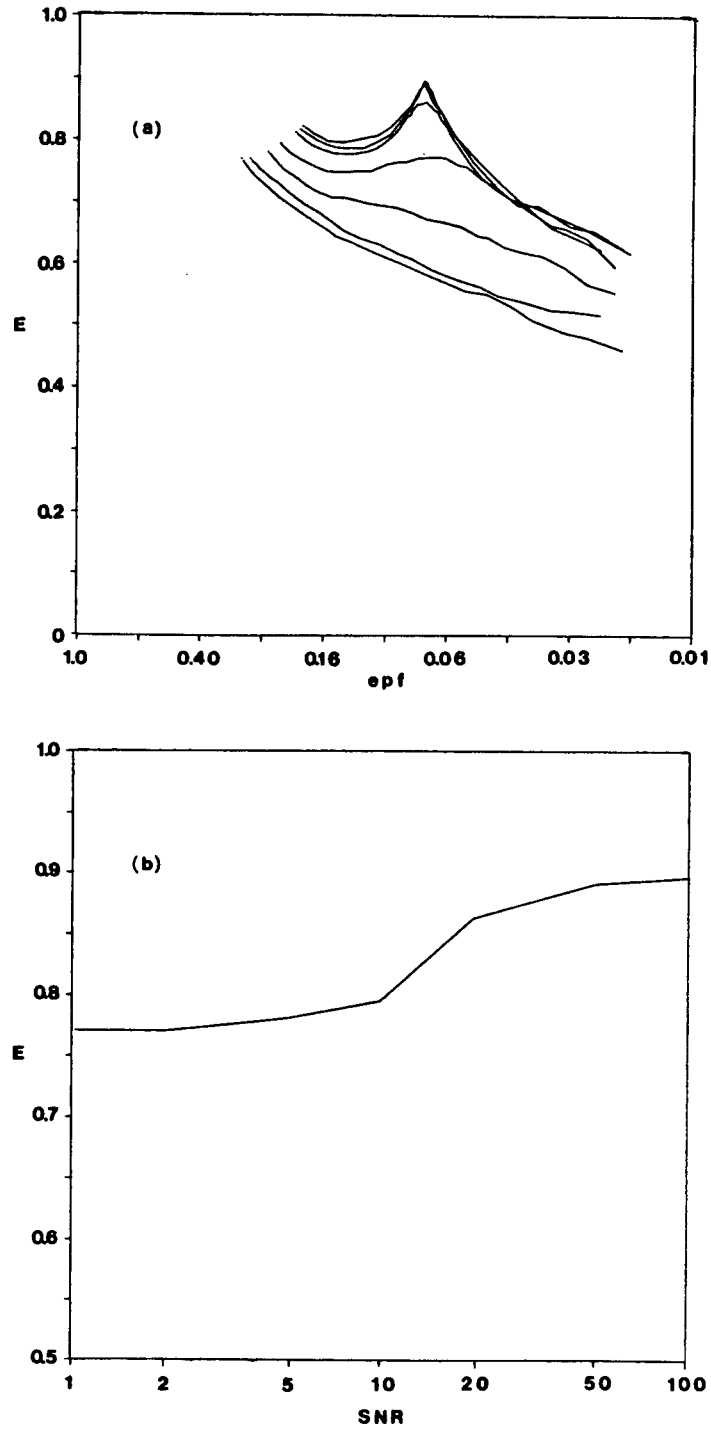


Figure 3.8 σ_f =1.6 $\nabla^2 g$ filtered rings image evaluation results: (a) SNR= 1, 2, 5, 10, 20, 50, and 100 (from bottom curve to top), δ=0.8; (b) peak evaluation scores against SNR

about 0.9.

When σ_f was decreased to 1.6 in Figure 3.8a, the SNR independence vanished. Instead, the curves now show a broad horizontal and vertical spread. The high SNR curves show a definite increase and peak about the critical epf at 0.074. However, the observation that the low SNR curves peak at high epf indicates that the noise has a sufficiently wide range of magnitudes as to render thresholding ineffective in removing it without also damaging the rings' structure. The max. E versus SNR curve reflects the new SNR dependence by ranging 0.77 at SNR=1 to 0.90 at SNR=100. As with Figure 3.5 these results are also beginning to resemble those of the three-level template operator.

It is generally seen that if the σ_f is matched to the edge structure of the image, then the noise rejection can be very high. If, however, σ_f is smaller than required, then thresholding can only safely remove the noise if the SNR is high as witnessed by the evaluation peaks near the critical epf. However, it is also seen that if the noise level is high and the filter too narrow, then it is perhaps impossible to select a general threshold that will remove the noise and also guarantee edge continuity.

3.4.2 Vertical Step Evaluation

The principal interest in processing the vertical edge is that it should be devoid of the curved edge bias present in the rings tests. This permits perfect continuity and thinness scores for perfect edges. Since decreasing σ_f increases the influence

of noise on the filter response which in turn produces curved edges, it is expected that the curved edge bias will dominate over the small vertical edge for small σ_f, high noise images. Tests analogous to those of Figures 3.4 to 3.6 where SNR is held constant at fifty and both γ and the threshold level changed were not performed. Having concluded that the optional γ is 0.8, it is adopted here also permitting the signal to noise ratio trials to proceed immediately.

σ_f =6.4:

The printout of the edge positions and directions (before threshold), Figure 3.9, for all SNRs reveals an image almost devoid of noise. The two edge features always stand out perfectly thin and connected. Only for SNR=1 is there an auxillary feature in the form of a small edge "island" in the middle of the left panel. For all the other images, only the edges are present. The principal influence of the noise appears to be on the straightness of the edge.

The plot of E against epf, Figure 3.10a, shows that for SNR=5 and greater, the evaluation plots are nearly identical each reaching a peak of 0.994 at the maximum epf of 0.018. The score is not perfect due to the slight winding of the edge observed. The curve of SNR=1 deviates markedly from the rest due to the presence of the noise island but still attains an excellent score of 0.958 at epf=0.018.

The max. E versus SNR plot of Figure 3.10b shows a near ideal evaluation for all SNR. Clearly, the elimination of the curved edges bias is responsible for this much improved

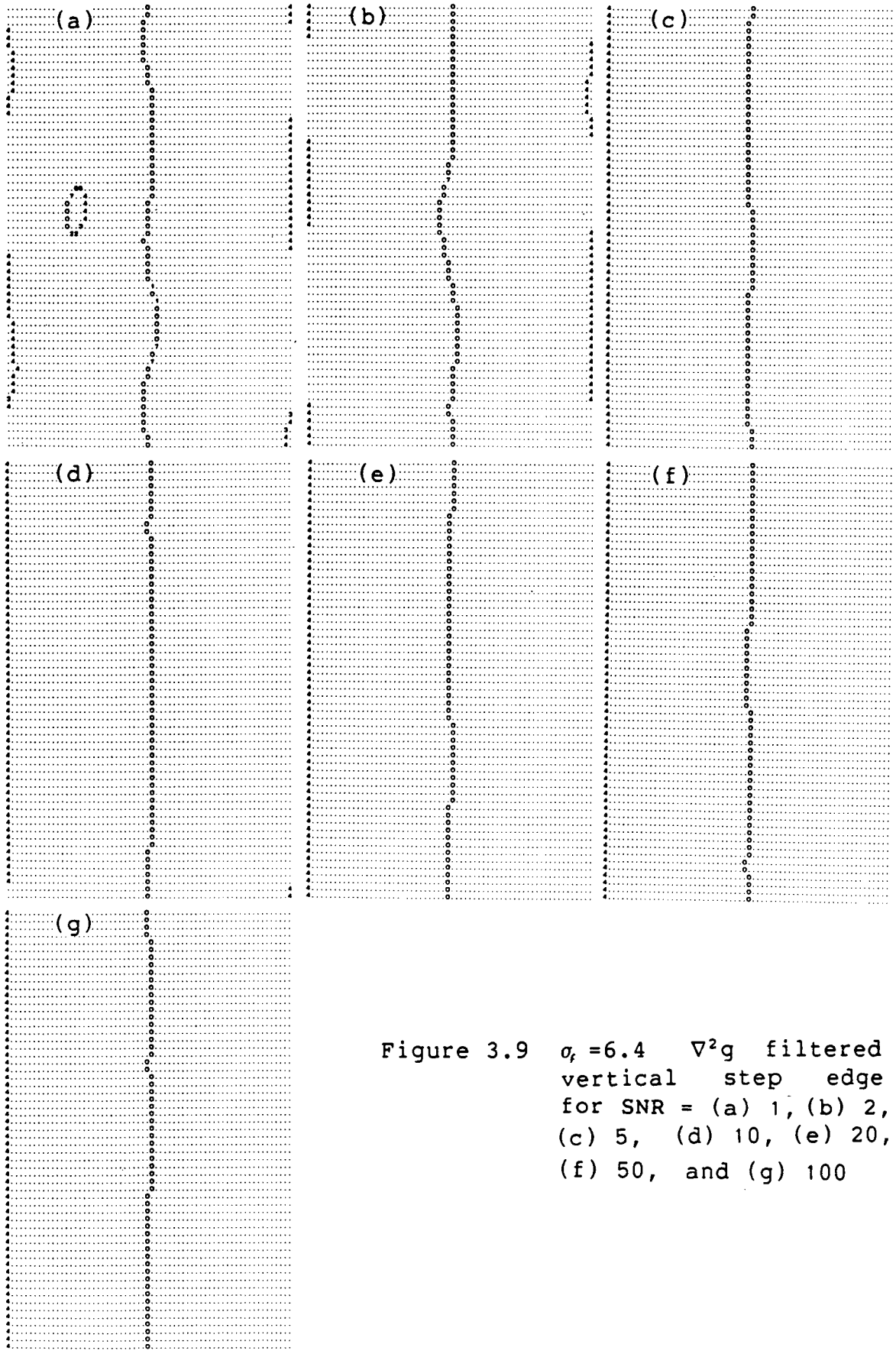


Figure 3.9 σ_f =6.4 $\nabla^2 g$ filtered vertical step edge for SNR = (a) 1, (b) 2, (c) 5, (d) 10, (e) 20, (f) 50, and (g) 100

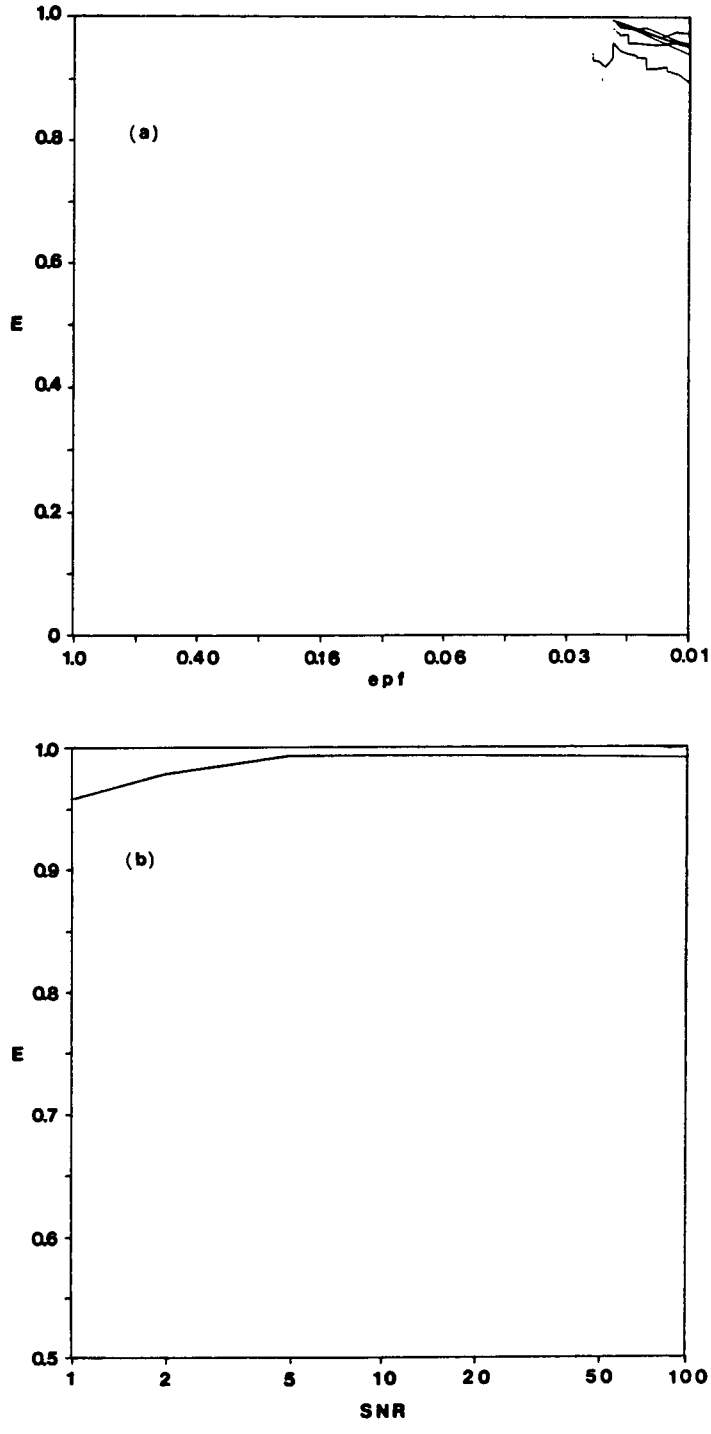


Figure 3.10 σ_f =6.4 $\nabla^2 g$ filtered vertical step evaluation results: (a) SNR = 1, 2, 5, 10, 20, 50, and 100 (from bottom curve to top), δ=0.8; (b) peak evaluation scores against SNR

performance.

σ_f =1.6:

The results here are markedly different from those of σ_f =6.4. The edge image printout, Figure 3.11, shows the absence of a discernable vertical edge below SNR=5 to the untrained eye. This is substantiated by the noise analysis of the previous chapter. By equation (2.48), the lowest tolerable SNR for σ_f =1.6 is 1.6625, or 3.325 by the definition used here. For SNR≥5, the edge becomes increasingly isolated from the background noise, and straighter. However, even at SNR=100, this background noise remains significant and approaches to within two pixels of the edge, but generally stays beyond three pixels.

The plot of evaluation measure against epf, Figure 3.12a, shows a great diversity of scores. Below SNR=10, the scores are seen to decline uniformly from a maximum of 0.77. This is consistent with the poor image quality seen in Figure 3.11. The three highest SNR curves show that the act of thresholding is effective in performing a noise cleaning operation. The peak scores are quite high, up to 0.978, and occur at epf=0.018, about that for an ideal edge. The max. E versus SNR curve of Figure 3.12b also approach that of the three-level template operator with a score of 0.77 at SNR=10 and steadily climbing to 0.978 at SNR=100.

In general, it was observed that the absence of a curved edge bias has resulted in generally increased scores approaching 1.0. Even though the σ_f =6.4 filter was not ideally matched to this edge image, it produced almost total noise imunity for all

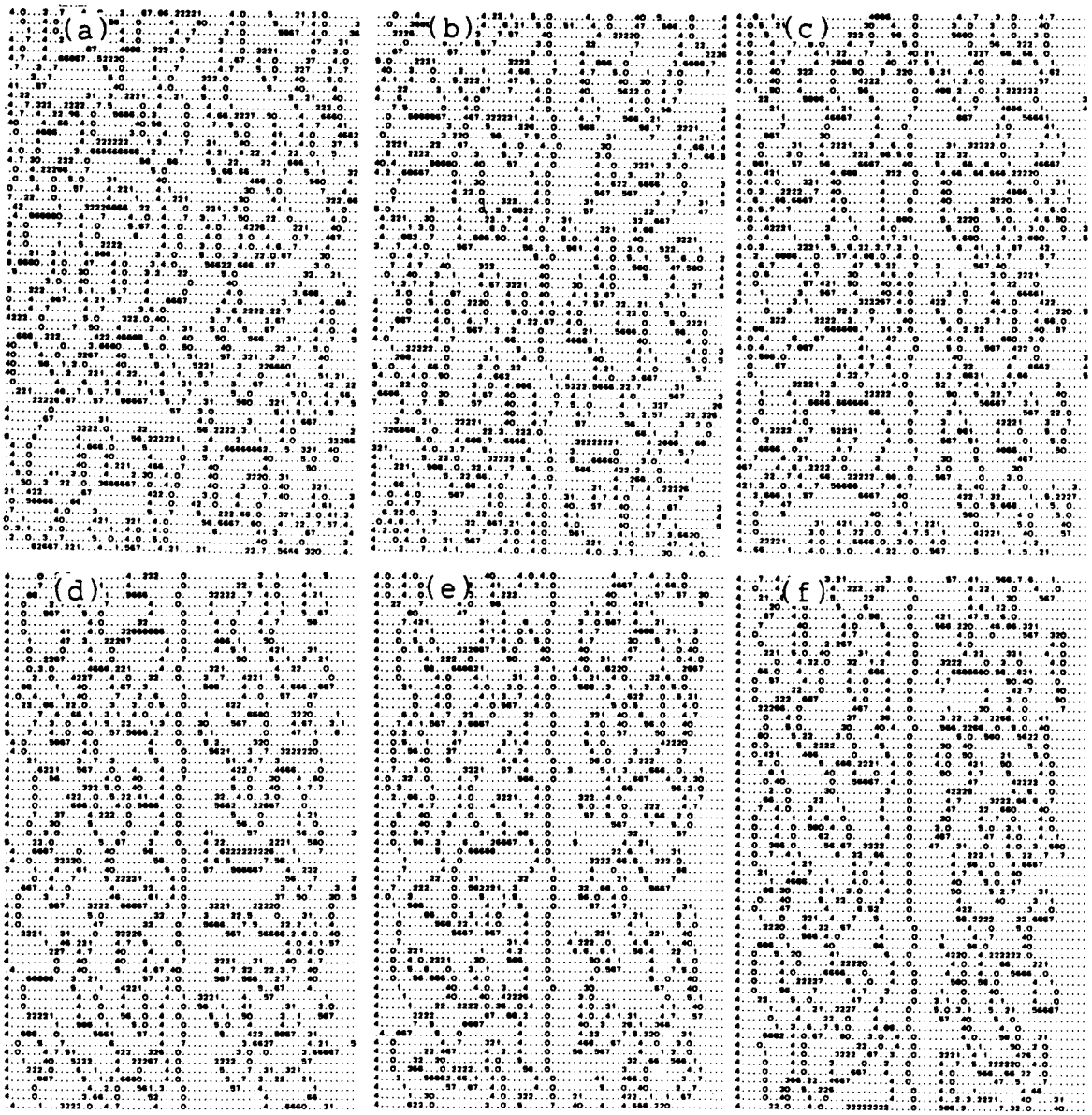


Figure 3.11 $\sigma_f = 1.6$ $\nabla^2 g$ filtered vertical step edge for SNR = (a) 1, (b) 2, (c) 5, (d) 10, (e) 20, (f) 50, and (g) 100

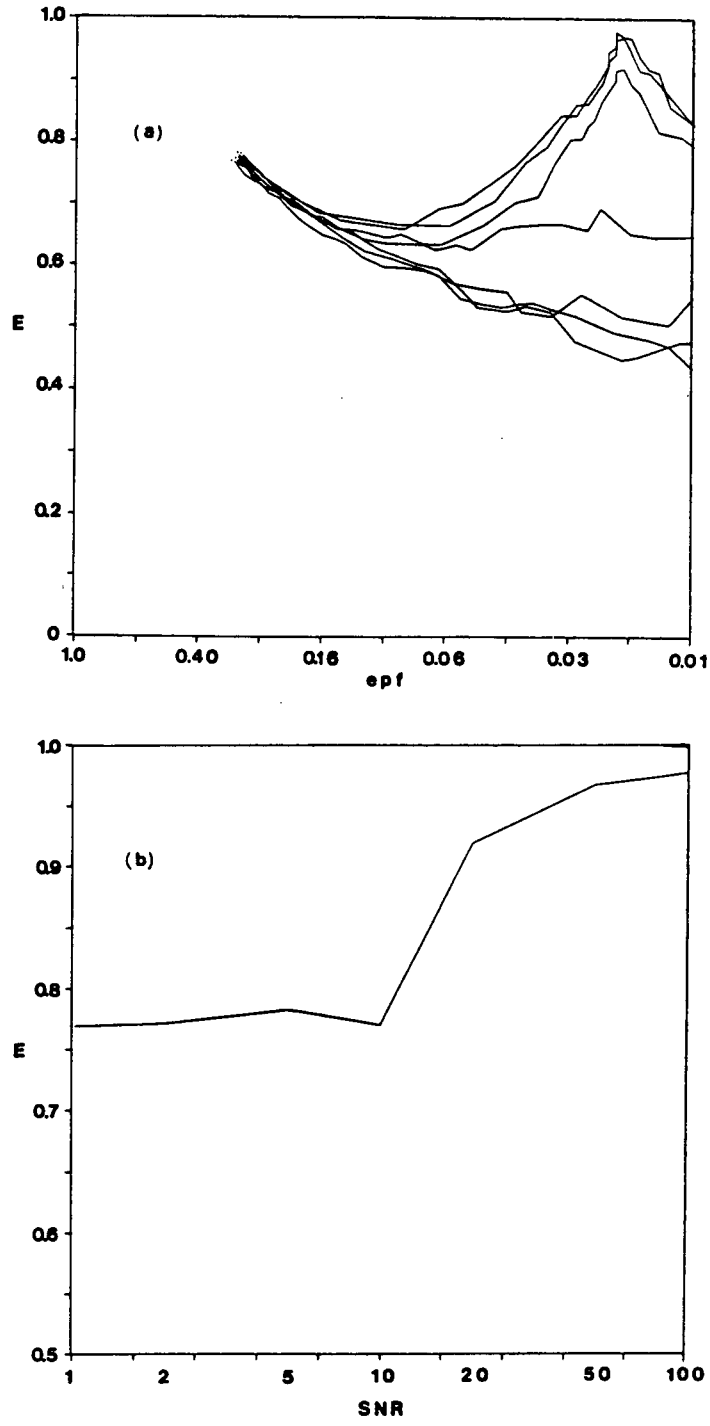


Figure 3.12 σ_f =1.6 $\nabla^2 g$ filtered vertical step evaluation results: (a) SNR = 1, 2, 5, 10, 20, 50, and 100 (from bottom curve to top), δ=0.8; (b) peak evaluation scores against SNR

SNR. The principal influence of the noise on the edge was to contort it from a pure vertical line. Only for σ_f=1.6 at SNR=1 was the test edge observed to be broken.

Even though it was outside of the experiment, the side edge was particularly interesting to observe. Being a pure step, it matched the original model used to design the filter. It is not surprising then to observe that this edge is particularly strong and readily isolated from the background. It is seen to become isolated at a lower SNR and, unlike the test edge, becomes perfectly straight even at intermediate SNR values for both filter variances.

3.4.3 Pure Noise Evaluation:

It was noted by Kitchen and Rosenfeld that even in a pure noise image, there will be a certain occurence of well-formed edges arising due to statistical fluctuations in image density or accidental alignments.

The smoothing property of the $\nabla^2 g$ filter has already been observed. A tendency has also been noted to clump noise into random regions of diameter on the order of σ_f. These edges are highly curved. They therefore present an opportunity to measure the filter's performance in natural images where borders may not be clearly defined, e.g. in remote sensing.

σ_f =6.4:

As for the rings image, the curves, Figure 3.13a, tend not to overlap with changing σ_f. They, however, peak early at maximum epf. The perfect score at γ=0 indicates perfect thinness

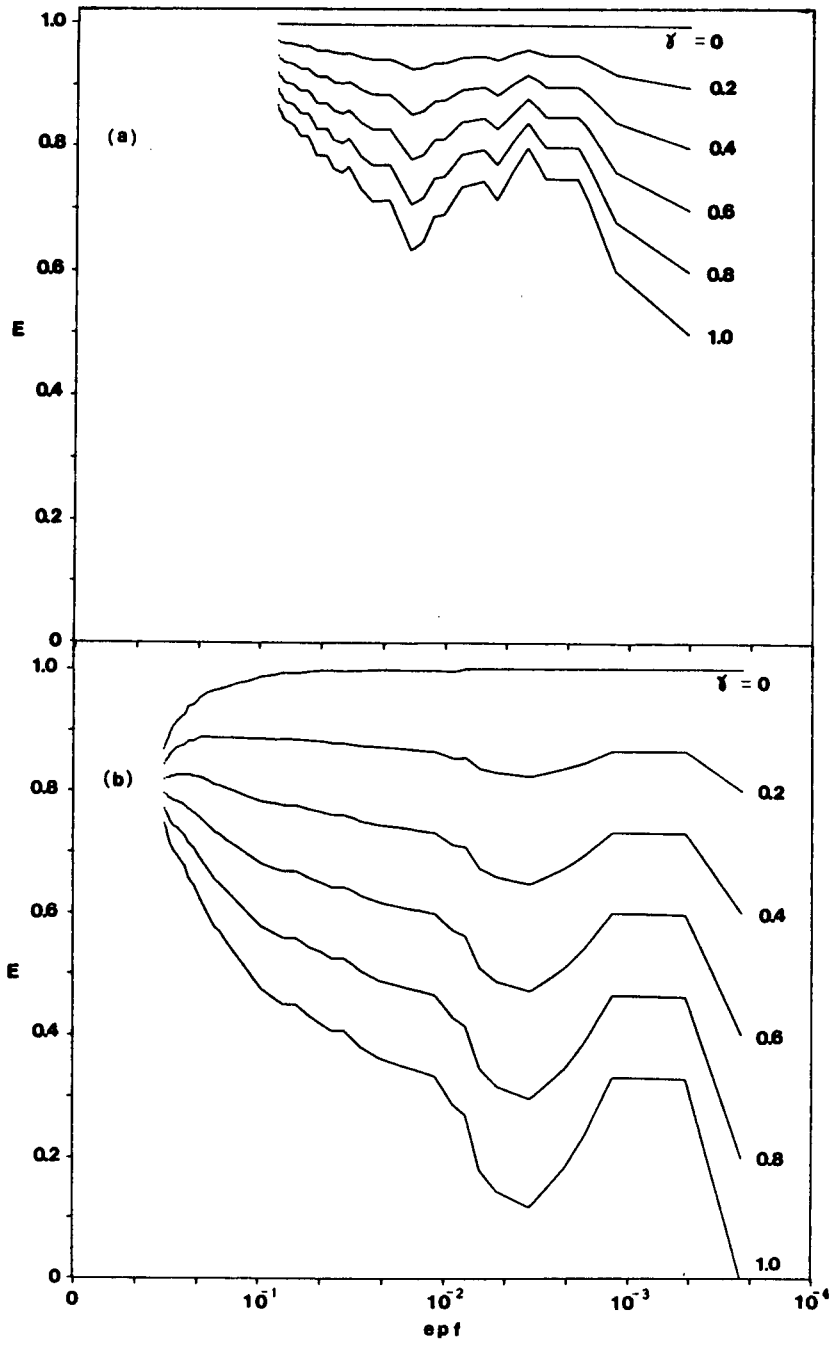


Figure 3.13 $\nabla^2 g$ filtered random noise evaluation results for (a) σ_f =6.4; (b) σ_f =1.6

throughout. The decline with epf of the remaining curves indicate the breakage of edge features with thresholding. Note that there is no optimal threshold point indicated here, the declining scores indicate that all the edge features are equally well-formed.

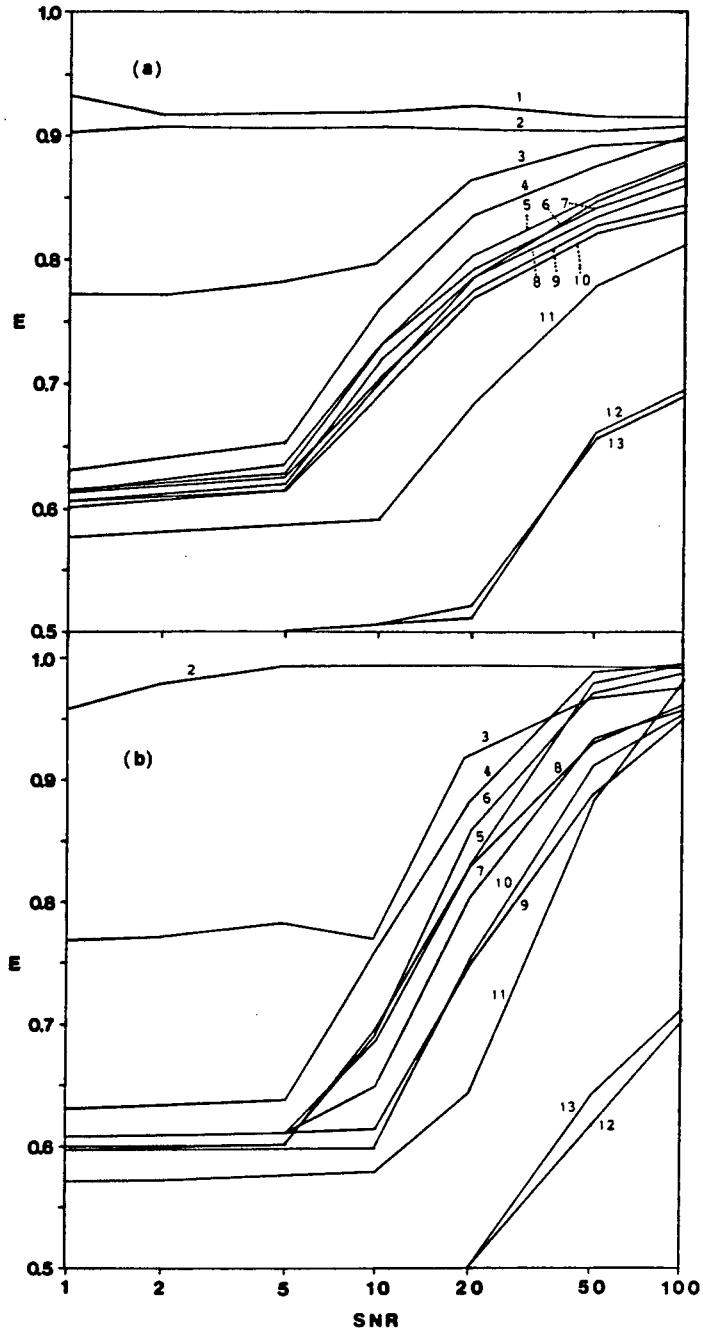
σ_f =1.6:

Again, the curves, Figure 3.13b, do not overlap with changing γ, but they do converge somewhat at high epf. This is due to the loss of thinness, previously observed with this σ_f, above epf=0.01. At high γ, the curves once again peak at the maximum epf level. Here also, then, there is no optimal threshold indicated. All of the edges are equally well-formed, though perhaps a bit cluttered before thresholding.

In general, from the high initial scores for all γ at large epf, it is apparent that this filter does produce well-formed regions in pure noise. From the decline in maximum epf with increasing σ_f it is clear that the widest regions are produced by the widest filter. Also, all the scores exhibit similar tendencies such as an increase in thinness and decline in continuity as the threshold level is raised. However, there is no indication of an optimal threshold level because of the absence of any peaks in the evaluation measure after thresholding is applied. This substantiates the claim of the last chapter that when confronted with an arbitrary image whose properties are not well-known, or are nearly random, then it is best not to apply any threshold at all after filtering with the chosen σ_f.

3.5 Comparison to other Edge Operators

Figure 3.14 shows the results of this chapter superimposed on the evaluation results of Kitchen and Rosenfeld. The plots show all of the curves produced by the $\nabla^2 g$ filter riding well above the others. The near-ideal response seen when the filter standard deviation most closely matches the image details to be resolved, $\sigma_f = 6.4$, is not approached by any of the other operators. However, the fact that the $\sigma_f = 16$ curve scores the highest shows that the good results of the $\sigma_f = 6.4$ curve are not so much the result of an ideal match to image detail, but rather are attributable to the noise cleaning influence of those wide filters and the general local line-like character of the test image edges. It is interesting to note how closely the $\sigma_f = 1.6$ curves match the general shape of the others. Evidently, the shape and amplitude of the noise response curves published is a characteristic of all narrow edge operators. It appears reasonable to predict that further reduction of σ_f will produce further agreement with the results of the other operators, indicating that the $\nabla^2 g$ filter offers little or no advantage where extremely fine resolutions are concerned. However, where a match to image detail of edge spacing greater than two pixels ($\sigma_f \geq 1.6$) is desired, the performance of the $\nabla^2 g$ filter is outstanding.



1. σ_f = 16.0
2. σ_f = 6.4
3. σ_f = 1.6
4. three-level
5. five-level
6. Kirsch
7. Sobel sqrt
8. Prewitt sqrt
9. Sobel sumabs
10. Prewitt sumabs
11. compass gradient
12. Roberts sqrt
13. Roberts sumabs

Figure 3.14 Superimposition of the $\nabla^2 g$ maximum evaluation scores upon those of Kitchen and Rosenfeld : (a) rings; (b) vertical step.

3.6 Conclusions

This chapter sought to prove two assertions: that the $\nabla^2 g$ filter provides superior performance over the majority of other edge operators used; and that thresholding is an unnecessary operation that may simply degrade the performance of the $\nabla^2 g$ operator. The superiority over other edge operators was clearly seen in the comparisons of the last sections. This superiority may be lost where resolution below two pixels is sought. The evaluations of the ring and step edge images show that an optimal threshold point can be found where the evaluation scores peak. However, the $\sigma_f = 16$ filtered rings evaluation and both noise image evaluations showed this peak to occur at maximum edge pixel fraction without threshold. In general, therfore, there is no guarantee that an optimal threshold point can be found that will not disrupt the $\nabla^2 g$ filter's strongest asset, the formation of closed edge segments. Thresholding is therefore not recommended as a noise cleaning method. Multiband filtering which takes advantage of the $4\sigma_f$ wide noise clean strip predicted and observed about strong edge features would be more effective.

Kitchen and Rosenfeld note that their edge evaluation method should not be considered the last word in judging an edge operator. There is a certain value in understanding the accuracy with which an edge operator localizes an edge about its known position. In the context of the $\nabla^2 g$ operator such a measure would serve to confirm the predicted distribution of the zero crossing, σ_z. However, instead of adopting the approach of Abdou and Pratt, or Peli and Malah, which weigh every edge pixel in the image to judge the overall goodness of fit, a more restrictive

test is recommended which recognizes that the test edge is usually fully resolved and isolated from the background sea of noise. This more sophisticated method would search for the output object edge near the true edge position and then follow it about the image, keeping to within $2\sigma_f$ of the true edge. The standard deviation of the edge data will then serve as an indication of the precision of the edge operator. Such a method, left to future work, combined with the results presented here should serve to complete the picture of the $\nabla^2 g$ filter performance.

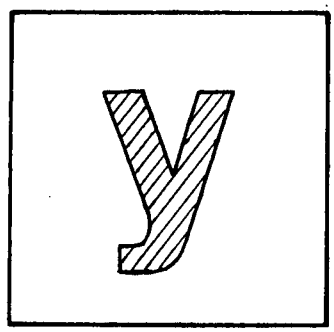
IV. FAST BINARY-IMAGE SEGMENTATION

4.1 Introduction

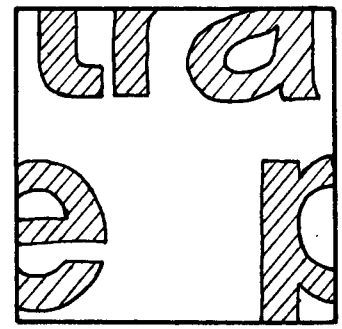
The purpose of the $\nabla^2 g$ filtering stage was to reduce the range of scale found in the input image and enhance those features whose edge spacing exceeds the design minimum. Rather than isolate the edges of the output image, as was done in the previous chapters, a more useful operation is to use the result to binarize the image into "black" and "white" pixels. Binarizing the output image essentially uses the zero level to threshold the filtered image for translation of positive "black" regions to unity, and negative "white" regions to zero. Since printed characters correspond to dark objects on a light background, they will appear as regions of unit magnitude on a background of zeros. The purpose of this chapter is to present a system to separate, or segment, the individual characters seen from the background. In the process, a description of those characters will also be produced to facilitate their recognition.

This description will take the form of positional and relational information of the external and internal borders of the character objects found. A binarized image, as opposed to an edge image, is essential to unambiguously distinguish between, and process, these borders. The task of segmentation can be defined as the operation of identifying those individual regions represented by "1"s in the image. To be considered segmentable, the complete object must be present in the image, separated from the image boundaries by at least one row or column of zeros.

E.g.,



Segmentable

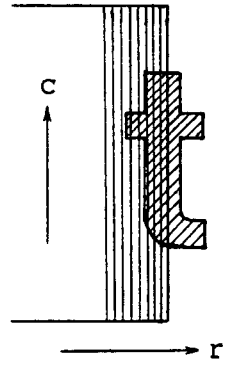


Not Segmentable

Segmentable objects will also be referred to as closed since their outer borders form closed loops.

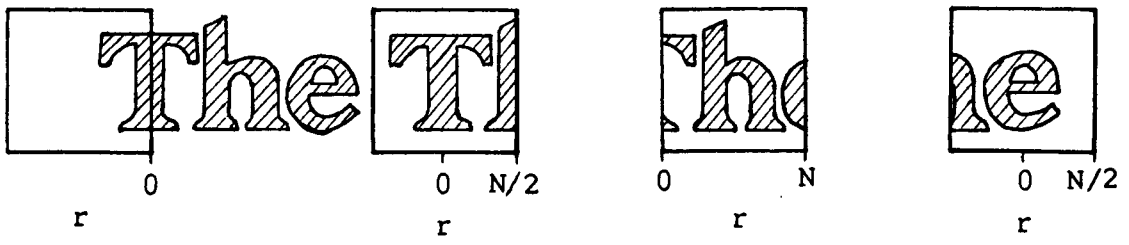
A number of constraints will be considered to be in force in this system. The foremost is that all processing be done in real-time. That is, closed borders must be detected, and their description relayed to the recognition system, concurrent with the image raster-scan. Those borders found not segmentable must be totally ignored.

The raster-scan will be considered to proceed along the local vertical of the printed page, advancing from left to right. For example:



The direction of the scan proceeds along c, with new rows advancing along r. This scanning method and coordinate system was chosen for consistency with a CCD image scanning device, one of the acquisition methods seriously considered. The vertical

extent of c currently stands undefined. However, 64 columns, spanning about two character heights, will frequently be used, again to be consistent with the CCD device considered. It is also possible that the extent of c may include more than one print line. However, only the case of a single line per scan will be considered here. Also note that there is no bound implicit in the row-wise extent of the scan. c is clearly finite, however r may extend indefinitely. The solution is to define a moving image window and bound r to the fixed dimensions of the window. For example, bound r between 0 and N. Then at different periods of the scan, the window and its row coordinates would have the appearance of:



Obviously r is recycled modulo N+1. The extent of r chosen is a flexible design decision. However, it will be shown that it must at best equal the extent of c for the segmentation method to be outlined to operate correctly. Generally, it will be assumed that the extent of r and c are equal.

Before proceeding further, some of the terms used rather loosely to this point, such as "border", must be rigorously defined. With a solid understanding of this image analysis terminology, we turn to a review of past work and then address the segmentation methods endorsed for this system.

4.2 Preliminary Concepts

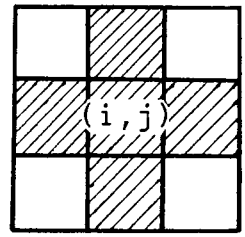
4.2.1 Connectivity

Define a binary digital picture as an integer array of points (i,j). Each of which may assume only the value 0 or 1. It is generally understood that in such a binary picture, 1s represent object points and 0s belong to the background. Rosenfeld [40], [60], has studied, at great length, the properties of such images. His definitions describing the interconnection properties among object and background points are now used widely in the field of picture analysis. Those definitions and properties most relevant to the ideas presented in the body of this paper will be outlined here.

Let $A = \{ (i_1, j_1),\ldots,(i_t, j_t) \}$ be a set of (i,j)s where $t \geq 1$.

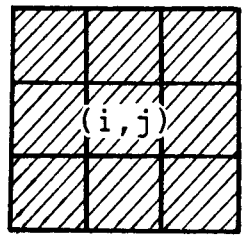
A can be defined as one of the following two paths:

4-path - if for each r, $1 \leq r < t$, $| i_r - i_{r+1}| + | j_r - j_{r+1}| \leq 1$. In other words, that (i_{r+1}, j_{r+1}) either equals (i_r, j_r), or is one of its horizontal or vertical neighbours. Such points are called 4-neighbours and are said to be 4-adjacent as illustrated below.



4-neighbours of (i,j)

8-path - if for each r, $1 \leq r < t$, we have $\max (| i_r - i_{r+1}|, | j_r - j_{r+1}|) \leq 1$. That is, (i_{r+1}, j_{r+1}) either equals (i_r, j_r) or is one of its eight horizontal, vertical, or diagonal neighbours. Such points are called 8-neighbours and are said to be 8-adjacent as shown below.



8-neighbours of (i,j)

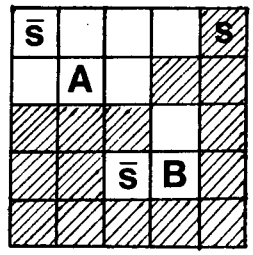
Let S be a subset of picture elements. Define (h,k) and (m,n) of S as being:

4-connected - if there exists a 4-path with (h,k) as the first term and (m,n) as the last, having all its members in S;

8-connected - if a similar 8-path exists between (h,k) and (m,n).

The dark object points of an image are considered to belong to S and are considered 8-connected. The background therefore belongs to $\bar{S}$ and is defined as 4-connected. All points of $\bar{S}$ that are not connected to the border of the picture are called holes in S. If S is connected and has no holes, it is called simply connected.

The principal advantage in using different types of connectivity in S and $\bar{S}$ is that this prevents $\bar{S}$ from being connected across a thin diagonal of S. For example, consider the following image array:



If $\bar{S}$ were defined as 8-connected, regions A and B would be considered as connected by virtue of the two 8-adjacent diagonal points. Therefore B could not be considered to be a hole inside the similarly 8-connected region S. However, if $\bar{S}$ is defined as 4-connected, B would then become a hole inside S, disjoint from A. This is in keeping with our intuitive notion of holes in images.

4.2.2 Borders and Edges

The outline of an object can be described through two features: its border, or its edge. A border will be defined as a simple, bounded, closed 8-path in the object S, which is everywhere 4-adjacent to $\bar{S}$. For very thin objects, one pixel across, the border and object may be one and the same.

An edge was defined by Rosenfeld [60] as a pair of 4-adjacent elements, one in S and the other in $\bar{S}$. The defintion applies regardless of the connectivity of S. When all adjacent edge elements are linked into a simple closed curve, they form the edge of an object. An edge is distinct from a border in that it lies neither within S nor $\bar{S}$ but rather at the junction between those regions. This results in the rather desirable property that all edge points are uniquely defined. This is not always the case with border points.

4.2.3 Chain Codes

Once the border points of the picture objects are found, the question remains as to how best to represent these points for storage in memory. Freeman [49] proposed the use of a simple chain code for the efficient coding of arbitrary shapes, or curves. The method, as applied to a rectangular image array, is fairly simple. Each 8-connected curve point is overlaid with the following 3X3 rectangular array:

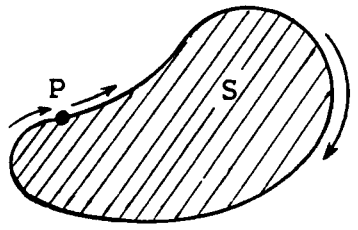
3	2	1
4	X	0
5	6	7

Chain Code Array

The numbers inside the squares, representing the points in the 8-neighbourhood of X, are called direction numbers. These direction numbers quantize the directions of the line segments between next points into 45° increments. Since each curve point can be quantized in this manner, only the coordinates of the first point of the curve need be recorded. The remaining points can then be specified by their direction numbers. The list of connected direction numbers so derived forms a chain code. This representation is more compact than straightforward coordinate storage since only three bits are needed to represent each point. Further reduction in storage can be realized if the curve is known to be slowly varying. In this case, only changes in the chain code of, say, ±45° need be recorded, requiring only two bits per point.

4.2.4 Trace Direction

A concept that will be employed in finding borders is the idea of an edgein and edgeout as defined by Zahn [70]. Consider a point P on the border (or edge) of an object S:



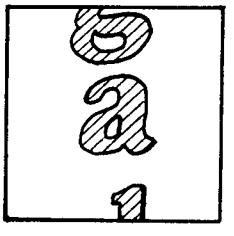
If a global, sequential method was used to find P, and later to proceed from P, we must define a direction for this trace. A right-handed, clockwise direction, as shown above, will be used. Now the edge or border point from which this trace arrives at P will be defined as the edgein to P. Likewise, the next point to which the trace proceeds from P will be defined as the edgeout from P. Freeman direction numbers will be used to describe the locations of the edgein, edgeout points.

You may notice that the term "edge" is used rather loosely here. This is to maintain consistency with the definitions of Zahn. In fact, these edgein(out) points will later be defined to lie on the object's border.

4.3 Review of Past Work

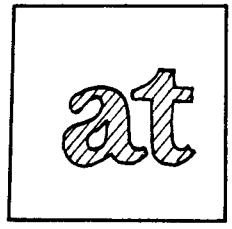
In the simplest character segmentation systems, the concepts of border, edge, and connectivity play little role. Some systems such as that of Mason and Clemens [61] simply scan the image, in a pattern similar to that proposed in the introduction, searching for blank scan rows. Object points found bounded between two such rows are transmitted to the recognition circuitry completing the segmentation process. In the more sophisticated method of Hoffman and McCullough [62], it is recognized that characters may touch or overlap causing a failure rate of up to 35% in a Mason and Clemens-type segmentation approach for 12 pitch sherif-type fonts. The solution was to develop two techniques to optimally sever character assemblies if a blank column is not found when expected. However, the net result is the same: to produce a pair of partitioning columns between which the desired character information lies. Column-wise transmission of this information completes the segmentation process.

These approaches are not satisfactory for two reasons. The first is that they do not address the problem of unsegmentable object points in the image window. These can readily occur when portions of characters from adjacent print lines protrude into the window. For example:



If all columns containing the "a" are to be transmitted, the

protruding ascender and descender must be transmitted as well. This problem must be dealt with by subsequent processing stages, or by narrowing the vertical extent of the scan and precisely guiding the scanner along the print line. Since it is desirable to place all responsibility for character segmentation into one stage, and to relax hardware or operator tracking proficiency requirements, neither of these solutions is satisfactory. A second objection to column-wise segmentation is that it cannot deal satisfactorily with overlapping characters. For example:



The "a" and "t" are clearly segmentable. However, no column can separate them without losing information from at least one of those characters. A solution to the shortcomings of column-wise segmentation is to search the image window for closed boundaries instead.

The approaches for acquiring border information fall into two broad categories: sequential border, or edge, following techniques, and non-sequential border, or edge, processing techniques. The former involves storage of the image in a randomly addressable array, a search for boundary pixels, and then execution of an algorithm to systematically follow the curve of each boundary found. The latter method involves recording each border, or edge, point found during the raster scan, followed by linkage of those points into closed borders during,

or after, the raster scan. Neither of these approaches, however, address the issue of the separation of touching characters. Hoffman and McCullough show this to be a complex issue, so it will not be discussed until much later in this chapter.

The sequential border following methods are the older of the two categories, and those most often seen applied to character segmentation. Their most notable feature is their simplicity. Once the first border point has been acquired, a multitude of methods can readily be conceived for acquiring the next point adjacent to it and so on around the border to produce a right-handed trace.

A pioneering application of this idea was explored by Greanias et al. [13]. They devised a largely analog method for the control of a flying spot scanner to first locate a potential handwritten character boundary, and then follow its outline with tight circular arcs centered on the light-dark boundary transition. Later, Clemens [63] proposed a purely digital method for character segmentation through border following. The search for adjacent border pixels was guided by a set of simple recursive arithmetic expressions. Beddoes and Lunscher [64] explored this segmentation technique and found its principal shortcoming to be an inconsistent view of border connectivity. Borders were considered 4- or 8-connected depending on which of the two possible diagonal directions was under consideration. As a consequence, not all thin, one pixel wide, diagonal lines could be followed.

Both border and edge following methods for 4- and 8-connected borders were presented by Rosenfeld [60]. The edge

follower examined a simple 2X2 pixel area to locate successive edge points along a border. The border follower required a larger 3X3 pixel neighbourhood, the center pixel of which was the current border point. The right-handed trace requires the outer eight pixels to be examined in a clockwise manner to search for the next border pixel. Rosenfeld and Kak [40] modified this method to include the labeling of the border pixels found. This permitted the known borders of an image to be ignored during a rescan in search of new borders. Unfortunately, this labeling process requires an extra two bits per pixel of memory.

Sobel [65] developed a novel, fast border following technique which involved applying a 3X3 operator over the binary image to produce an eight-bit neighbourhood code which in turn replaced each pixel of the input image. This neighbourhood code described the arrangement of object pixels adjacent to the central pixel. After recoding of the image is complete, the neighbourhood codes are read and a table lookup technique is used to quickly guide the method along any borders present, and produce a chain code description as output. The two-pass nature of this technique could possibly be modified to make it suitable for real-time applications.

There are a number of evident drawbacks common to all sequential techniques. Foremost is the requirement to store a complete copy of the input image and provide random access to all of the pixels. This obviously increases the memory requirements of the system, especially when a Sobel-type system is employed. However, memory management also becomes more complex when the moving window of a dynamic image is considered. Clearly some

form of two-dimensional circular list must be employed to store the window which in turn requires communication of the current top and bottom of the list to the border following system. The methods cited operated on static images so memory management was never a consideration. Because of the frequent memory references involved in the search for border pixels, frequently guided by a small sequential algorithm, there is some concern as to whether these techniques could meet the real-time speed constraint. However, D'Amato et al. [66] report a 100 character per second recognition rate in a system that uses a sequential border follower of an unspecified type. Evidently, therefore, these systems can be adapted to our needs.

The non-sequential border finding methods operate very differently. They address the above mentioned memory management problems by simply storing no more than 2 lines of the input image at a time. In fact, the method of Pavlidis [67] stores only one line. To encode the image data, the end coordinates of runs of object-points in the current line are entered into a line adjacency graph (LAG). On completion of an image scan, the LAG data is mapped into a connected string of boundary points which completely describes the input image.

The idea of using a special data structure to connect border points found non-sequentially was also explored by Chakravarty [68]. He proposed that the input image be scanned by a 3X3 operator, a procedure requiring storage of at least two image rows, to search for 8-connected object pixels that can be linked into straight line segments. These line segments are stored in a special list data structure. Concurrent with the scan, these

lists are linked together through the use of pointers to form a description of all borders and lines present in the form of a chain code. No effort, however, is made to achieve closed borders since the method stores thin lines as unique line segments. A true border follower would multiply define the line segment pixels as it traced its way around them.

The opposite problem is found in the approach of Batchelor and Marlow [69]. A 3X3 window is also applied to the input image as it is raster scanned, but this time thin lines are ignored by the system. This is because the system seeks to process object borders found in the 3X3 window with special purpose hardware in real-time, and thin lines represent an ambiguity that had not been resolved. During the scan, the chain code of the border points found is stored in a RAM memory addressed by the points' coordinates. Therefore, even though only two rows of the input image are stored, the initial memory advantage of this system is subsequently lost.

All of the above systems, in addition to the shortcomings cited, are two pass in operation. During the first pass, the image is scanned and processed. During the second pass, the image border description is generated. This is unacceptable in a dynamic image incorporating a moving window. Border descriptions must be produced "on the fly" concurrent with the raster-scan. Modifications to the techniques cited may remove this and other drawbacks; but it was felt the answer lay elsewhere, in a system that inherently detects all borders and presents no barriers to generating their description on the fly.

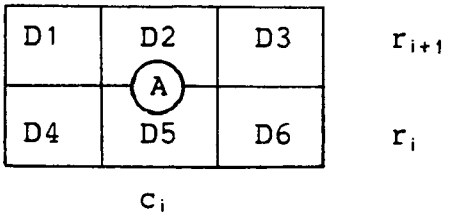
C.T. Zahn [70] devised a binary-image description procedure

founded on the detection and linkage of all edge-points present. The result was a complete, chain code description of any two-dimensional patterns seen. A preliminary method was even provided to show how the object patterns could be reconstructed from the description. Though no suggestion as to the method's suitability for real-time border processing was made, it was felt that this description method incorporated enough flexibility to meet this objective. In the subsequent sections, Zahn's method will be presented and its application to closed-object segmentation explored.

4.4 Zahn's Binary-Image Description Method

C.T. Zahn developed a method of formally describing binary-image patterns which is readily adapted to segmenting such images in a non-sequential way. This technique is based on identifying the edge points between an object and its background and assigning them an edgein-edgeout direction pair consistent with a right-hand trace. Through simple arithmetic relationships this information can be processed to provide a complete description of the borders, both external and internal, of any object totally contained within the image. This description can be provided immediately after a raster scan line admits the complete object into the current image window. Furthermore, explicit storage of no more than two image lines is required.

As already defined, edge points are located between picture elements. To detect the edgein-edgeout directions, Zahn centers each edge point on a 2X3 window:



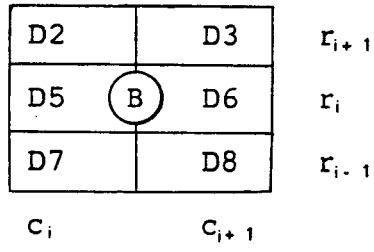
The edge-point at A occupies the coordinates $(c_i, r_i + 0.5)$.

By examination of the window, the edgein-edgeout directions can be readily found using elementary boolean functions. These were found by Zahn to be:

neighbourhood	edgein
$D5 \wedge \overline{D1} \wedge \overline{D2} \wedge \overline{D4}$	1
$D2 \wedge D6 \wedge \overline{D5}$	3
$D2 \wedge D3 \wedge \overline{D5} \wedge \overline{D6}$	4
$D2 \wedge \overline{D3} \wedge \overline{D5} \wedge \overline{D6}$	5
$D1 \wedge D5 \wedge \overline{D2}$	7
$D4 \wedge D5 \wedge \overline{D1} \wedge \overline{D2}$	0

neighbourhood	edgeout
$D3 \wedge D5 \wedge \overline{D2}$	1
$D2 \wedge \overline{D1} \wedge \overline{D4} \wedge \overline{D5}$	3
$D1 \wedge D2 \wedge \overline{D4} \wedge \overline{D5}$	4
$D2 \wedge D4 \wedge \overline{D5}$	5
$D5 \wedge \overline{D2} \wedge \overline{D3} \wedge \overline{D6}$	7
$D5 \wedge D6 \wedge \overline{D2} \wedge \overline{D3}$	0

It should be noted that this particular window cannot detect edges along directions 2-6. To remedy this, an additional window is overlain near A similar to that above but rotated 90°:

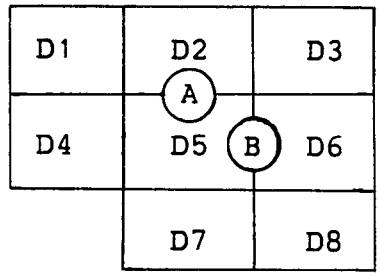


This time the edge-point is centered on B at location $(c_i + 0.5, r_i)$. For this window, a new table of boolean functions defines the edgein-edgeout directions:

neighbourhood	edgein
$D6 \wedge D7 \wedge \overline{D5}$	1
$D6 \wedge D8 \wedge \overline{D5} \wedge \overline{D7}$	2
$D6 \wedge \overline{D8} \wedge \overline{D5} \wedge \overline{D7}$	3
$D3 \wedge D5 \wedge \overline{D6}$	5
$D2 \wedge D5 \wedge \overline{D3} \wedge \overline{D6}$	6
$D5 \wedge \overline{D3} \wedge \overline{D6} \wedge \overline{D2}$	7

neighbourhood	edgeout
$D6 \wedge \overline{D3} \wedge \overline{D2} \wedge \overline{D5}$	1
$D3 \wedge D6 \wedge \overline{D2} \wedge D5$	2
$D6 \wedge D2 \wedge \overline{D5}$	3
$D5 \wedge \overline{D6} \wedge \overline{D8} \wedge \overline{D7}$	5
$D5 \wedge D7 \wedge \overline{D6} \wedge \overline{D8}$	6
$D8 \wedge D5 \wedge \overline{D6}$	7

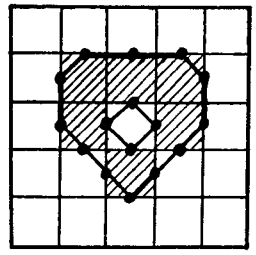
Since the two windows together cover all of the possible edge directions, the two windows can be combined into a single window of the shape:



The positioning of the B point between D5 and D6 will later be seen to be necessary to achieve dynamic linkage of edge-points during a raster scan.

The principal advantage of Zahn's approach lies in the fact that edge-points are uniquely defined. This simplifies the processing of detected edges and implies equal processing time for every edge-point. The principal disadvantage here is that an extra bit is required to store the one half coordinate difference marking the edge position. Since Zahn advocates only the storage of curvature points (edgein $\neq$ edgeout), this may not be significant in light of the storage saved. Another objection is that 90° corners resulting from the intersection of horizontal and vertical object sides are not detected directly. These corners are represented by two 45° corners. Therefore, these and sharper corners are essentially smoothed into a rounded-edge contour not entirely representative of the border shape. This is

illustrated in the following example:

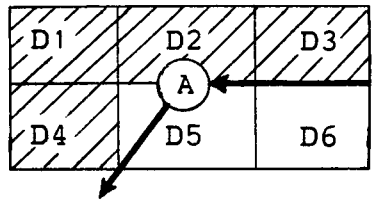


As well as the rounding effect, it is also seen that corners of any type cause the detection of more edge-points (black dots) than border points.

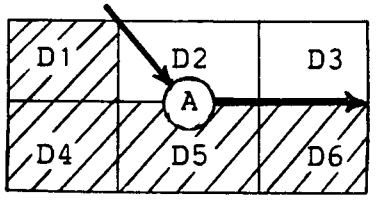
The last objection is largely an aesthetic one. This technique does not generate border points which, at heart, is really what is wanted. A simple modification, however, can accomplish this.

The modification is to associate the edgein-edgeout directions generated not with the between-pixel edge-points, but with the nearest object border point. This initially causes some multiple detection of certain border points, but that's acceptable. The border points are still only detected two at a time.

As an example, consider the following neighbourhood:

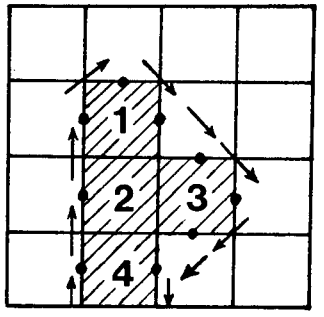


Here, edgein = 4 and edgeout = 5 and these values will be associated with the pixel at D2. On the other hand, the following neighbourhood,



associates the edge directions with pixel D5. A similar modification would be done with edge-point B. The net result is that the edge directions become associated with border points allowing the original coordinate word lengths to be maintained.

That some redundant border point detection occurs can be seen in the following example. Consider the image segment:



The edgeout chain code generated for this edge curve would read:

2 2 2 1 7 7 7 5 5 6 ,

with pixels 1 and 3 being recorded three times. A border following scheme would record each pixel once and generate the code:

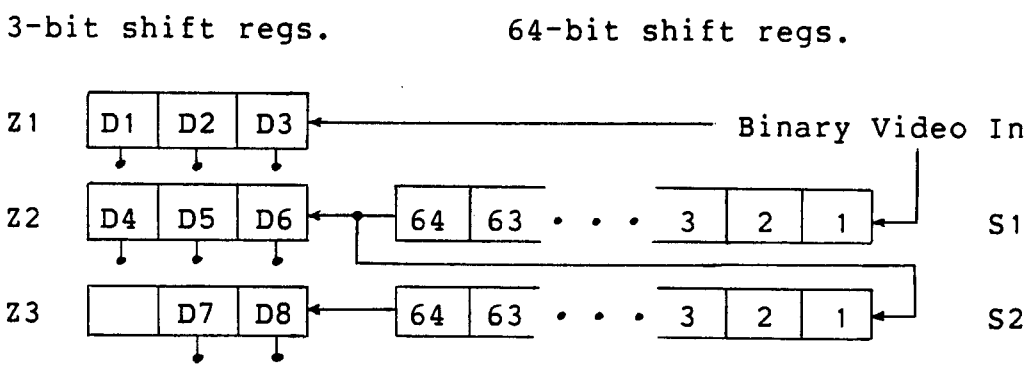
2 2 2 7 5 6 .

Brief examination shows that only the first edgein and the last edgeout need be stored at each pixel. Also, it is seen that the net result of these multiply defined borders is that a pixel ends up pointing at itself via its edge directions and linkages.

Before proceeding, it would be expedient to take a brief look at how the above windows would be implemented in hardware.

Assume, first of all, that the binary image data is being delivered in serial fashion. Also assume, for illustration, that the video image is 64 pixels wide and at least 3 tall.

The proposed implementation would consist of two types of shift registers. One of which would be 64 bits long; two of these would be used. The other would be three bits long and provide access to all three bits; three of these would be used. The basic implementation would then be:



Non-sequential Hardware

All of the shift registers are clocked at the video bit rate. The three-bit shift registers hold the current window, or neighbourhood of points to be examined. The Zahn-based method would access eight of the bits in this window. The two 64-bit shift registers delay the video data by two rows. This configuration results in a 3X3 window sweeping across the image from left to right. Clearly, no true image frame buffer is maintained. The following border point linkage procedure will make extensive use of the order in which this scanning window detects the border points.

4.5 Border Linkage and Closure

The Zahn border point detection method just outlined consists of a local operator applied to every pixel in the image. However, it was shown by Minsky and Papert [71] that a collection of local operations could neither determine connectedness nor closure of simple curves. Some further, global processing was needed. The necessary global information for this is provided in the border point coordinates and the edgein-edgeout directions. In this section, a method will be presented for using this information to determine both connectedness and closure.

It is important to note that in the coordinate system used, r points up in the direction of new image rows which also coincides with the direction of the local horizontal of the character lines. c is directed to the right along a row and represents the character's local vertical.

Linkage:

Using the available r, c, edgein, and edgeout values, pairs of connected border points, (r_a, c_a), (r_b, c_b), can be properly linked by using the following properties observed by Zahn:

(1) $\text{edgeout}_a = \text{edgein}_b$ between all connected points.

(2) $r_{i+1} \geq r_i$ for constant c, and $c_{i+1} \geq c_i$ for constant r. This is a statement of the nature of the scanning process, ie., new border points are always detected at progresssively larger coordinate values. This property will be central to the proper ordering of linked points.

For successively detected border points, (r_1, c_1) and

(r_2, c_2), with $edgeout_a = edgein_b$ along the following directions:

(3) 0-4: $r_1=r_2$ and there can be no other point in this row with c_i such that $c_1<c_i<c_2$. Therefore, consecutive occurences of such points along the same row are automatically linked.

(4) 1-5: $r_1-c_1=r_2-c_2$ and there can be no other point (r_i, c_i) with the same difference such that $r_1<r_i<r_2$ and $c_1<c_i<c_2$. Therefore consecutive occurences of the same differences are automatically linked.

(5) 2-6: $c_1=c_2$ and there can be no other point on this column with r_i such that $r_1<r_i<r_2$. Therefore, consecutive occurences of such points in the same column are automatically linked.

(6) 3-7: $r_1+c_1=r_2+c_2$ and there can be no other point (r_i, c_i) with the same sum such that $r_1<r_i<r_2$, $c_1<c_i<c_2$. Therefore, consecutive occurences of the same sum are automatically linked.

Condition (1) simply states the fact that each border point points to the next border point in succession. Conditions (3) to (6) provide the means for detecting when a given pair border points succeed one another along the same border.

The process of linking all the border points into a closed curve is accomplished through a series of lists.

Conditions (3) to (6) are implemented through a set of eight tables which perform the pairing of linked points. Note that explicit use of condition (2) is made throughout, ie., that new

points arrive with ascending coordinate values.

Direction 0-4:

edgein = 0 = edgeout	
2nd entry	1st entry
r_2, c_2	r_1, c_1

edgein = 4 = edgeout	
1st entry	2nd entry
r_1, c_1	r_2, c_2

These points are automatically sorted by the virtue of c increasing while r is constant in this direction. Therefore, these tables need contain only one entry each.

Direction 1-5:

edgein = 1 = edgeout		difference	edgein = 5 = edgeout	
2nd entry	1st entry	r-c	1st entry	2nd entry
r_2^{-m}, c_2^{-m}	r_1^{-m}, c_1^{-m}	-min	r_1^{-m}, c_2^{-m}	r_2^{-m}, c_1^{-m}
r_2^{o}, c_2^{o}	r_1^{o}, c_1^{o}	0	r_1^{o}, c_2^{o}	r_2^{o}, c_1^{o}
r_2^{m}, c_2^{m}	r_1^{m}, c_1^{m}	max	r_1^{m}, c_2^{m}	r_2^{m}, c_1^{m}

These points are automatically sorted on consecutive occurences of the same difference. Since these directions point to or from new rows (i.e., data not yet arrived), they must be accumulated in tables much larger than the 0-4 direction. One slot is provided for each possible difference. It will be shown that these differences must be subject to modular arithmetc with the modulo determined by the image width. Therefore, even for the largest of images, such a table is comparatively easy to manage. Also, access to the entries is possible through hash-coding, i.e., using the difference result as the address to the table entries.

Direction 2-6:

edgein = 2 = edgeout		c	edgein = 6 = edgeout	
2nd entry	1st entry	coord.	1st entry	2nd entry
r_2^o , c_2^o	r_1^o , c_1^o	0	r_1^o , c_1^o	r_2^o , c_2^o
r_2^m , c_2^m	r_1^m , c_1^m	c max	r_1^m , c_1^m	r_2^m , c_2^m

These points are automatically sorted on consecutive occurences of the same c values. Like the 1-5 tables, these too have many entries. This time, the number of entries corresponds to the maximum number of c values.

Direction 3-7:

edgein = 3 = edgeout		sum	edgein = 7 = edgeout	
2nd entry	1st entry	r+c	1st entry	2nd entry
r_2^{-m} , c_2^{-m}	r_1^{-m} , c_1^{-m}	-min	r_1^{-m} , c_1^{-m}	r_2^{-m} , c_2^{-m}
r_2^o , c_2^o	r_1^o , c_1^o	0	r_1^o , c_1^o	r_2^o , c_2^o
r_2^m , c_2^m	r_1^m , c_1^m	max	r_1^m , c_1^m	r_2^m , c_2^m

This case is similar to the 1-5 direction, only here the entries are ordered according to their coordinate sum. Again, consecutive entries are automatically sorted. The table would also have to contain as many slots as the modular range of sums.

There are a number of implicit assumptions underlying the construction of these tables. The most fundatmental is that the r-coordinate is limited to the same range of values as the c-coordinate. However, r represents an unrestricted number of new rows of image data. r must therefore be recycled in a

modulus at least equal to the maximum c value. It will be shown that all of the previous linkage relations, particularly along 1-5 and 3-7, still hold for a modular r. Another important assumption is that points detected at B in the window are entered first. This is necessary because B values lie farther down in the image and therefore test older points than does A. Also, B coordinates and edges along ±45° diagonals lie between A points and so must be linked into the border first to guarantee proper linkage of A points. B was positioned between D5 and D6 to ensure correct linkage along directions 3 and 7. Finally, the representation of the given tables is somewhat deceptive. In a full implementation, the information stored in the slots may be of a very different character. For instance, pointers for lists to be discussed next may be stored instead. Also, some of the information shown is redundant; for instance, since the 2-6 direction is stored in increasing c values, there is no need to store r, and since all these data structures provide automatic sorting, there is really no need to store the second entry.

Concurrent with the entry of points into the linkage tables, two other lists are maintained:

list-1 - This list would contain r, c, edgein, and edgeout data for later acquisition and processing. This is the primary store for detection data until the border points and/or chain codes are transmitted to the next processor after closure detection. Data in this list is simply stored in sequential order as it arrives.

list-2 - This list can be considered a horizontal extension of list-1 since it has the same number of entries, each of which is intimately associated with one list-1 entry. List-2 is a list of

pointers, the pointers being addresses to other list-1-list-2 entries representing successive border points as they would be linked in a right-handed border trace in the image. By tracing through these pointers, this entire approach achieves global closure detection.

The physical structure of these lists is therefore:

Address	list-1	list-2
1	r_1, c_1, e_i, e_o	-> entry a
2	r_2, c_2, e_i, e_o	-> entry b
3	r_3, c_3, e_i, e_o	-> entry c
:	:	:
N	r_N, c_N, e_i, e_o	-> entry x

where "->" refers to "pointer to". It will be assumed that a 0 in list-2 signifies a null pointer (i.e., the linkage has not been resolved). This is why the list addresses are started at 1.

The entries of list-2 are generated by the linkage tables. In fact, only the list-1-list-2 address need be stored in the linkage tables since list-1 is used to store the general information on border points as they arrive. Also, some of the information in list-1 may be redundant. For instance, in generating and processing a chain code, only one of edgein, edgeout is really needed (usually edgeout). The necessary size of these lists is indeterminate, and depends on the size of the image and the complexity of the objects contained. An attempt will be made to address this question empirically later. When the end of these lists is encountered, the address counter will

simply cycle back to 1 causing old entries to be overwritten. In this way, objects that could not be segmented would simply be forgotten.

The complete processing sequence can now be outlined:

(1) Before acquiring image data, clear all linkage table entries and list-2 pointers to 0. This serves as a general reset preventing false detection of closed borders.

(2) On detection, enter the border points into list-1 and the list-1 addresses into the appropriate linkage tables.

(3) When a linkage is flagged, place the list-1 address of the edgein point into the list-2 slot of the edgeout point. When the edgein point is the first entry, its list-1 address can be found in the linkage table. When it is the second entry, the address of the list-2 slot of the edgeout point is found in the linkage table.

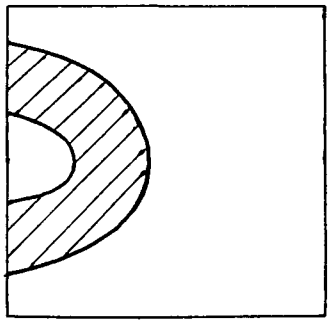
(4) On anticipation of closure (a process defined later), trace through the two lists under the direction of the list-2 pointers. If the trace ends at a null pointer, the current border is not closed. If the trace ends at its starting point, the border is closed and transmission of its points may now take place.

A number of points concerning this processing sequence deserve special attention:

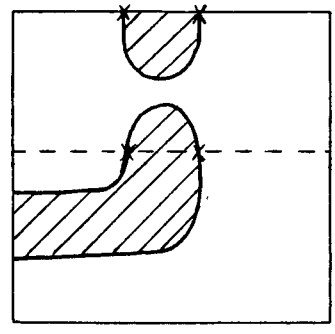
(a) Step 4 is a very much streamlined border tracing method. By following pointers in memory, this approach achieves global closure detection at a speed proportional to the border perimeter and the memory reference rate. This is also the only

substantially sequential procedure in the entire processor and so may be the most critical in determining its overall speed.

(b) To ensure that unlinkable objects (i.e., those which touch the "sides" or original "bottom" of the image) are not seen, special conventions must be invoked when the Zahn window is at the image extremities. The scanning circuitry must flag instances when these extremities are reached. A preferred convention is that when the window is positioned with the A point on the extreme right or left column of the image, or below the first row (at start up), set those window pixels overhanging the image to black. This will cause unlinkable objects to merge with the image borders, and will provide the detection circuitry with consistent information for edge direction decisions. However, to prevent the extreme right column from logging the image boundary and wasting list memory, when on the right column, disable detection of the B point. The net result is that those border points which touch the sides will always point to, or from, null entries. Therefore, if list-2 entries are always cleared on transmission, unlinkable borders can never be detected in step 4. E.g.,



untraceable



could be accidentally linked if list-2 entries are not first cleared

In the long run, the unlinkable borders simply get overwritten as

the finite list slots are recycled.

(c) On a similar note, however, all linkable, or closed borders are seen immediately when complete. This follows from the fact that closed borders leave no null entries in list-2. Anticipating when a border is likely to be closed will be the topic of the next section.

(d) Due to the recycling of the list-1-list-2 memory slots, there is a remote possibility of entering an infinite loop while tracing through list-2. This can occur even though list-2 pointers are cleared with new entries and during border transmission. If an object is unclosable or of such an extent along r that it cannot be closed before recycling of the lists, the pointers at its deepoest level may point to more recent entries. If these recent entries form a closed border on the current scan the trace will encounter an infinite loop with no way of stopping. To prevent this, a counter can be maintained in parallel with the trace, and incremented on each trace step. Should the counter value exceed the list length, the trace has entered an infinite loop and can be stopped without closure being flagged. In practice, this was observed to have occurred once in 6400 scan lines of 128 pixels in width with a list-1-list-2 length of 10,000 entries.

On verification of closure, the border points must be transmitted to another circuit for reconstruction of the object and/or recognition. Transmission is a simple matter of again tracing through list-2 as outlined in step 4, except that the list-1 coordinates and direction numbers are also read and transmitted, and list-2 entries are cleared immediately on being

read. It is here that redundant border points can be removed. A simple comparison circuit or test can be applied to determine if any border point coordinates are repeated. If they are, all but the last repeated will be ignored in the transmission.

It will now be shown that if modular arithmetic is employed, the pairing of border points along the 1-5 and 3-7 directions is assured. Lines along these directions are linked together through common coordinate difference, r-c, and sums r+c. If the register lengths available were infinite, these trivial calculations would pose no problem, but this is not the case. In our tentative implementation, the c dimension would be 64 pixels. To maintain symmetry in the coordinate word lengths, the r dimension will therefore be forced to cycle at modulo 64. The question now arises: can topologically linked border points along the above two lines, but falling on rows r=+31 and r=-32, still be linked by the same simple expressions? The answer is yes. To show this, we will use an arbitrary modulo M (preferably a power of 2) and Figure 4.1.

The figure will be used to illustrate the behavior of the line r+c=K. The region squared off in bold is the region physically seen by the camera scanner. All points outside this region map back onto points within it by virtue of modular arithmetic. For example, points in region 5 share a one-to-one correspondence with points in region 6. Therefore, unscanned portions of characters presently residing in region 5 will subsequently be written into region 6.

For the equation r+c=K, there will be certain regions in the plane where overflow errors would occur because the numbers are

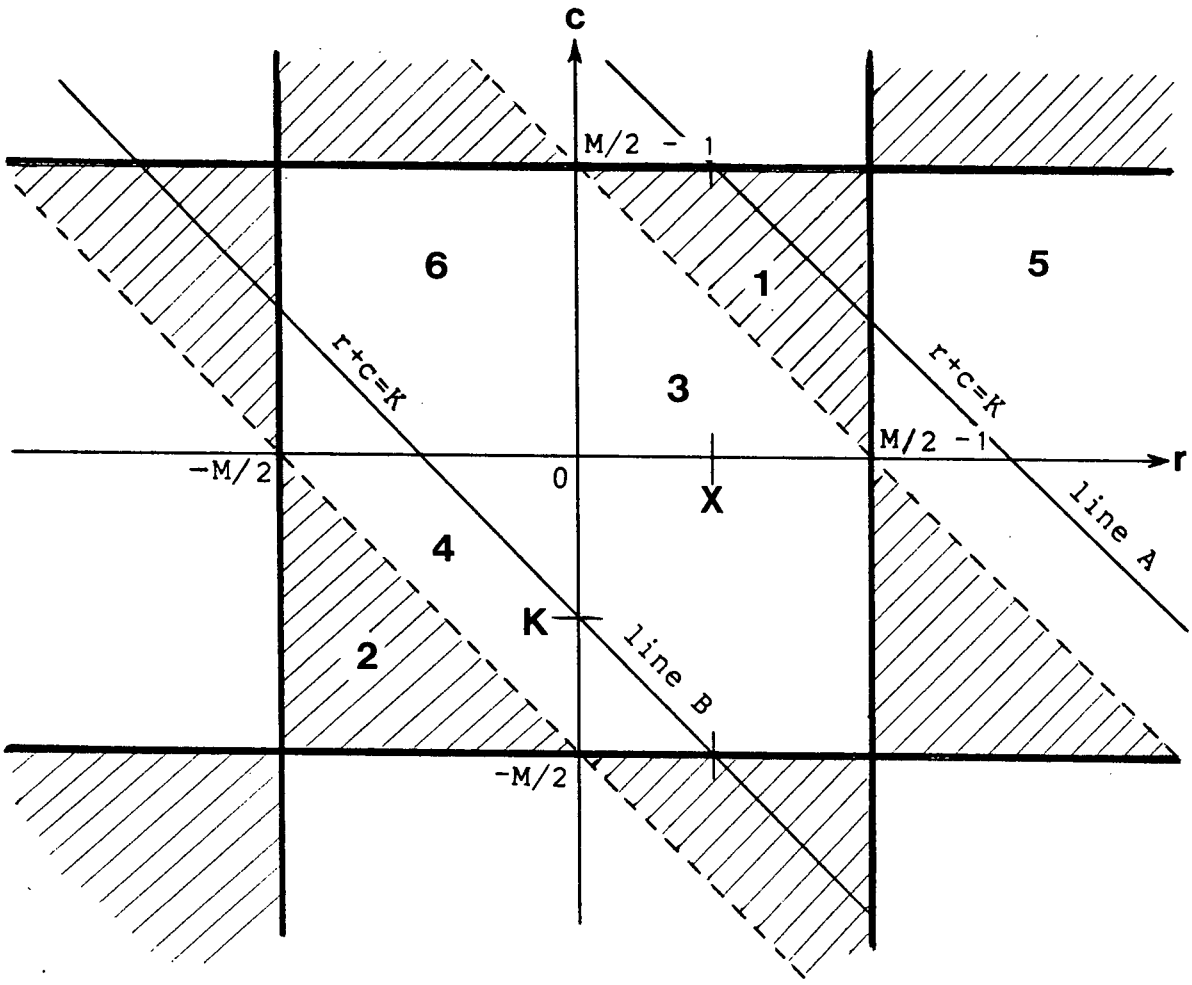
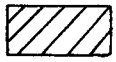


Figure 4.1 Modular image representation

too large (labelled  in the figure). These regions are defined as:

In region 1:

$r + c > M/2 - 1.$	The resultant sum here would be represented as a negative number.

In region 2:

$r + c < -M/2.$	The resultant sum here would be represented as a positive number.

Careful examination of these regions reveals that the sums generated in region 1 are exactly those generated in region 4 with a one-to-one mapping; and sums generated in region 2 are exactly those generated in region 3, again by one-to-one mapping.

Therefore, points on the overflowing line segment of line A in region 1 will map exactly onto the non-overflowing line segment of line B in region 4, and yet both lines will satisfy the relation r+c=K by virtue of modulo M. Furthermore, points in region 5 which represent the continuation of the lines outside the positive modulus limit will map into region 6 and again directly onto line B. Therefore, even ignoring overflow, points which lie on the line r+c=K in region 1 will still satisfy the equation r+c=K in region 5.

Also it might be supposed that if a line segment, B, already exists and is currently unlinked, it may inadvertantly be linked to a new line segment, A. This will in fact happen under only special, tolerable circumstances. Since lines A and B are parallel, of slope -1, and separated in both dimensions by the modulus M, line B will always be completed, or linked, before line A is first detected. This is because line B leaves the

visible region at r=X if it is not first connected, and line A also can start no earlier than r=X because of the above mentioned constraints on the lines.

The only exception to the above is the special case where there are two unlinkable objects present: one at the top of the image (using the currently depicted orientation) and one at the bottom, and the bottom one has an edgeout along line B at r=X and the top one has an edgein along line A also at r=X. However, this is not a serious situation because, since both polygons are clearly unlinkable, at worst, the algorithm will link them into one large unlinkable object. (This is guaranteed if list-2 entries are cleared as new edge points are entered.) Furthermore, since the length of the two lists is finite and also modular, a large unlinkable object will become overwritten in time like any other and be forgotten.

In summary then, modular arithmetic can be used in the labelling and processing of border points. Overflow errors or flags can be ignored in the calculation of r+c and r-c (the argument is the same as above by symmetry) for determining border point linkage, and yet correct results are assured.

4.6 Closure Detection

Actual detection of border closure requires the list-2 trace outlined in step 3 previously. However, since this step is time-consuming, some reliable method must supplement this trace to anticipate if closure detection is likely. The discussion of two such techniques forms the topic of this section. The first, a "region counting" technique, physically labels or colors all object regions seen. When a certain label is no longer seen on a scan it is assumed closed and a list trace is done. The second approach is less sophisticated and less reliable, but easier to implement. Called the Euler number approach, it strives to detect probable closure by detecting local changes in image topology within a given row. Both techniques operate non-sequentially.

4.6.1 Region Counting Approach

This technique was given its name because it could be used effectively as a means of counting the number of connected object regions in the image. It was adapted from a method for determining region connectiveness by Rosenfeld and Pfaltz [72]. However its very ability to count regions requires that it is able to detect closure, and it is in this connection that it is applied here.

The technique requires the maintenance of a small image map (no larger than the two shift registers needed for non-sequential detection) of a certain fixed depth, say four bits per pixel. A cross reference table with as many entries as levels in the image map (16 in this case) is also needed and a pair of flags per

level entry. The method operates in the following manner:

1. On startup a counter is initialized to some number, say i=0.

2. Image points enter the image map in shift register form as before but now the image is also viewed through the following 2X3 window:

<- indicates the source of new image points.

3. Point A is considered the current point. Now examine the last point, L:

   (a) If L=0 and A≠0, a new object point has been detected. Examine B and C:

   (i) If B and C=0, increment i and store it in the shift register at A. Also look up the value of i in the cross reference table and set a pair of region-seen and region-active flags.

   (ii) If B or C≠0, give A the number found in B or C and set the region-seen bit for that number in the cross reference table.

   (b) If L≠0, set A to the value of L and also examine B and C:

   (i) If B and C=0, carry on.

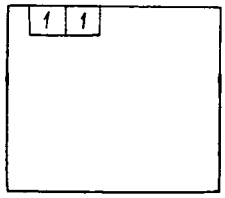
   (ii) If B or C≠0≠L, look up B or C in the cross reference table and set their region-seen flag. Also set the C or D cross reference flags alongside A in the table and A's flag alongside C or D. This identifies both numbers as belonging to the same region.

4. On the completion of the scan of a row, the cross reference table is systematically examined to locate which regions were seen during the scan. When a set region-seen flag is found, a trace is conducted through its cross references to set their region-seen flags true also. Caution must be exercised to prevent infinite loops during this trace. When complete, the entire set of region-seen flags is again examined to determine if any of the previously active regions were not seen in this scan, as indicated by a true active-region flag but false region-seen flag. If this is the case, then a trace through the list-2 entries is made to find the closed border. On completion, clear the cross reference table entries that were now found to be closed.

5. Continue the image scan. On incrementing the region counter, i, observe a certain modulo (in this case 16), however, skip i=0 since this is the null label.

The operation of this technique is perhaps best seen with an example. Consider an image with at most four possible, and cross referenceable, regions:

1. One row of image date (1s) first enters the image:

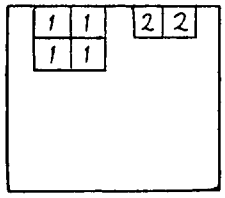


Image

Region	Cross Ref.				Seen	Active
1	X	0	0	0	1	1
2	0	X	0	0	0	0
3	0	0	X	0	0	0
4	0	0	0	X	0	0

Cross Reference Table

2. At a later time, another region row appears in the image:

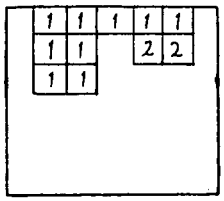


Image

Region	Cross Ref.				Seen	Active
1	X	0	0	0	1	1
2	0	X	0	0	1	1
3	0	0	X	0	0	0
4	0	0	0	X	0	0

Cross Reference Table

3. With the arrival of new image data, region 1 is seen to merge with region 2:



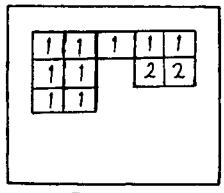
Image

Region	Cross Ref.				Seen	Active
1	X	1	0	0	1	1
2	1	X	0	0	1	1
3	0	0	X	0	0	0
4	0	0	0	X	0	0

Cross Reference Table

The merger of region 1 with 2 is recorded by cross referencing region 2 and setting its "region-seen" flag.

4. The object now breaks away from the top image border forming a closed, simply connected object:



Image

Region	Cross Ref.				Seen	Active
1	X	1	0	0	0	1
2	1	X	0	0	0	1
3	0	0	X	0	0	0
4	0	0	0	X	0	0

Cross Reference Table

Having broken away, no regions are seen on this scan, causing the "region-seen" flags to remain reset. A subsequent examination of this table by the processor will reveal that two active regions, linked by a cross reference, were not seen. A trace through list-2 will then be ordered to find the closed border.

The main attraction of this method is that it immediately flags the arrival of a closed object. When implemented properly and thoroughly, there is no ambiguity about closure.

The principal disadvantage of this technique is its complexity. To be workable with real character images, many more

features will have to be implemented than have been shown here. On the surface, we see that there is a need to utilize a number of shift registers for storage of the region numbers. There is also the need to store and maintain the cross reference table and region counter. Tracing through the cross reference table to find which active regions were not seen is a sequential operation, and therefore time-consuming. Provision must also be made to prevent infinite loops during a trace.

There are also a number of disadvantages which are not immediately obvious. It is difficult to fix a modulus for i, the region counter. In a sufficiently complex, distorted object with background noise, the number of counted and merged regions can become quite large. This, in turn, would force the cross reference table to become quite long.

Since detection of closure is not confirmed until a full row has been scanned, there is no optimum way to search list-2 to find the connected outer border and the borders of any holes it may contain. This necessitates scanning all of list-2 and keeping a third, binary list to record those list-2 entries already examined.

To circumvent these many implementation problems another, somewhat less reliable, method was developed. However, what it lacks in reliability it gains in simplicity.

4.6.2 The Euler Number Approach

An Euler number is a topological property of binary images and is one of the few such properties that is locally countable. A very thorough discussion of Euler number evaluation can be found in Gray [73]. Basically, an Euler number, E, represents the difference between the number of objects in an image, B, and the number of holes, L. ie.,

$$E = B - L \ .$$

This number can, however, also be found using readily countable image elements. ie.,

$$E = n_0 - n_1 + n_2$$

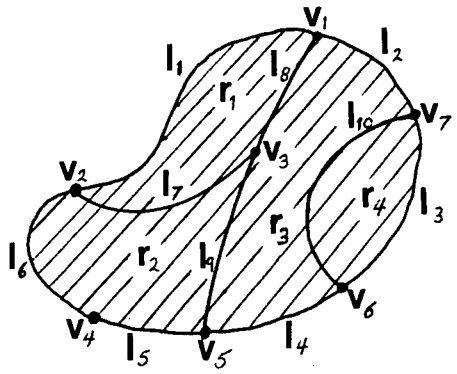
where: n_0 = the number of vertices,

n_1 = the number of line segments,

n_2 = the number of object regions contained by the line segments.

The vertices are defined as places where two or more line segments meet. The following two examples should clarify the concept:

Eg. 1:



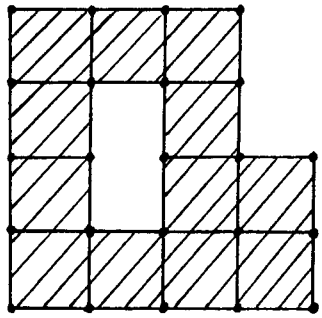
v - vertex
l - line
r - region

$n_0 = 7, \quad n_1 = 10, \quad n_2 = 4$

$E = 7 - 10 + 4 = 1$

Since there is only one object, without a hole, E=1 as expected.

Eg. 2:



$$n_0 = 23, \quad n_1 = 35, \quad n_2 = 12$$

$$E = 23 - 35 + 12 = 0$$

Note that line segments do not criss-cross holes.

The Euler number is a locally countable property and can be found for a complete image by summing the contributions from all mutually exclusive image segments. It is known that the Euler number, like any locally countable property, cannot detect the individual values of B or L. However, the change of the Euler number, ΔE, in a dynamic image can be readily determined locally. This information can, in turn, then be used to signal that a new closed object or hole may have appeared in the image.

To implement ΔE detection, two approaches have been developed for this dynamic image case. In method A, 1s topped, the image is considered to be framed in 1s regions from which the object appears as droplets that later break free. In method B, 0 topped, the top of the image is considered to consist of a row of 0s below which the object appears from "nowhere". These methods are illustrated in Figure 4.2.

Setting the two left and right sides of Figure 4.2 to 1 is done to prevent objects which touch the image sides from influencing the Euler number. This is because such objects are

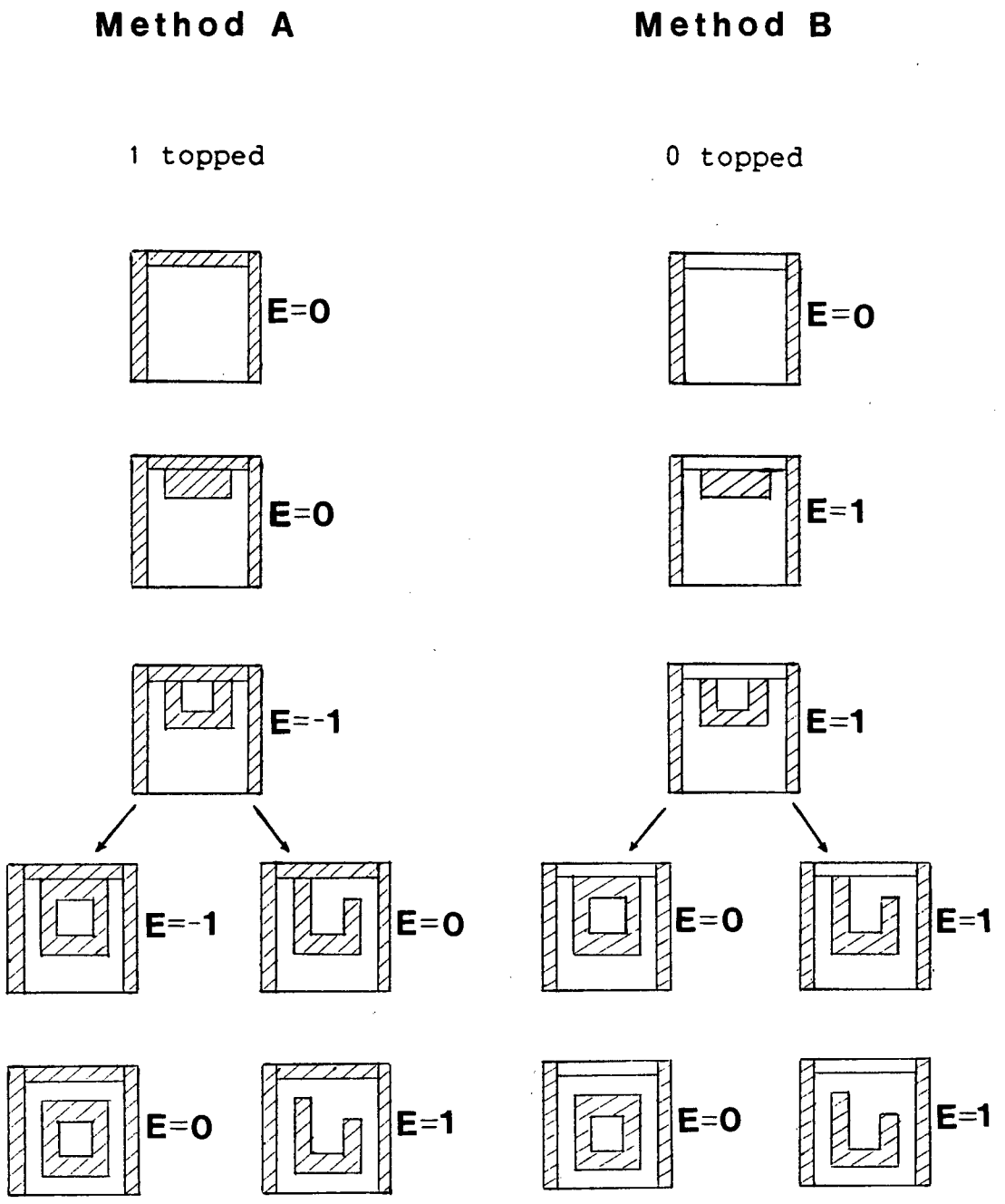


Figure 4.2 Euler number variation in 1s topped and 0 topped image representations

considered unlinkable. Examination of the Euler number at the various steps in the example shows that the two methods do not behave the same. The 0 topped method B, for instance, only detects the appearance of a new feature, body or hole, but does not detect its separation into the frame. The 1s topped method A, however, detects the first appearance of a new hole, and the separation of a new body into the image frame. It also detects the merger of an interior hole with the outside in a similar way as it detects separation.

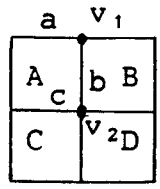
In summary, the Euler number changes as:

	ΔE	
	1 topped	0 topped
new object appears	0	+1
new hole appears	-1	0
hole broken	+1	0
hole established	0	-1
object separation	+1	0
2 objects merge	0	-1

In comparison it is seen that the two methods complement each other. Neither's Euler number changes in the same manner as the other during an event, and all but only two of the ΔE pairs is unique. Unfortunately, it is these non-unique pairs, which reduce the reliability of the method. Of course, we have not considered the case of multiple objects entering the image field. The net result then is that ΔE can be confused when it is evaluated one row at a time.

The overall observation is that method A is useful in detecting a separated, or possibly separated, object, and method B is useful in detecting established holes.

To detect these changes in the Euler number, a 2X2 square local neighbourhood is examined:

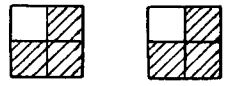


In each method, the evaluation of ΔE is done with respect to pixel A, and the total ΔE for a row, or portion of a row, is found by summing all the local values. ΔE is calculated by considering if a vertex is present at v_1 or v_2, whether a line is present at a, b, or c, and whether A is 0 or 1. Methods A and B differ largely in how they consider a and v_1. Method A always considers there to be a line and vertex at a and v_1. If A=0, method B never considers there to be a line at a and considers v_1 to be a vertex only if B=1. Occasionally, a line or vertex must be cancelled in order to avoid being counted twice.

Since there are sixteen possible boolean expressions for this window for each method, the detailed analysis will not be presented. The final result, however is:

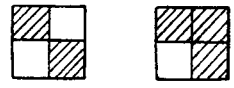
Method A (1s topped)

$\Delta E_1 = +1$ for 

$\Delta E_1 = -1$ for 

Method B (0 topped)

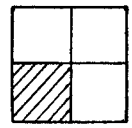
$\Delta E_0 = +1$ for 

$\Delta E_0 = -1$ for 

The beauty of the method rests with the fact that all of the above conditions can be determined with simple boolean tests, and ΔE accumulated using a simple up-down counter.

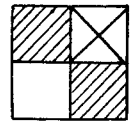
Since many objects, or segments of objects, can be detected on a single row scan, it is best to stop and test the accumulated ΔE at certain strategic locations.

For the 1s topped case, this place was chosen as the right corner of an object's boundary. This would be signalled by



This configuration must be present on the right side of a separated object. If at this point, the accumulated ΔE_1=+1, then such a separation may be probable.

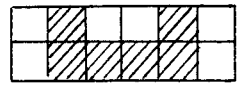
Likewise for the 0 topped case, ΔE_0 would be tested if the following configuration appears:



(X indicates don't care). This configuration must appear at the upper right corner of a closed hole. An accumulated ΔE_0 of -1 to this point indicates that a closed hole is probable.

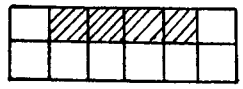
Since the two ΔE counters are bounded to the range of -1, 0, and +1, provision must also be made to ensure that they are not incremented or decremented outside of this interval. This can happen with the presence of multiple unseparated objects or holes. For example,

1s topped:



A ΔE_1 of -1 is accumulated after this unseparated hole has been scanned;

0 topped:

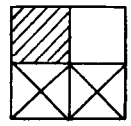


A ΔE_0 of +1 is accumulated after this unseparated object has been scanned.

In each of the above examples, the method is working correctly in that the correct ΔE per scan will be accumulated. However, the true objectives of this approach are being overlooked. That is, accumulate the ΔE individually for each object encountered during a scan line to detect closure. To accomodate the presence of multiple objects provision must be made to reset the Euler number counters whenever an incomplete object or hole has been encountered. This will require one more boolean test for each method.

Method A (1s topped):

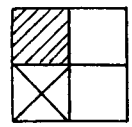
Reset ΔE_1 for



This indicates that an unseparable object has been encountered, and that an unseparable hole may be encountered.

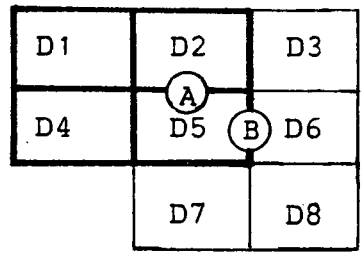
Method B (0 topped):

Reset ΔE_0 for



This indicates that an unseparable object has been encountered.

Before summarizing the complete closure detection procedure, it would be expedient to show how this method would integrate with the border detection scheme. The current 2X2 window would overlap the border detector's window as follows:



The current window is outlined in bold. The point to be noted is that each of the two above stop and test conditions occurs when there is a pixel 4-adjacent to the previous A edge. If either ΔE counter indicates the presence of a segmented object or hole, then it is a necessary condition that this A edge also belongs to a corner where edgein ≠ edgeout. Consequently, in all of the border representations to be discussed later, it will be recorded in the linkage tables and its list-1 address can be held available in a register. Therefore, when a closure test condition is flagged, initiate the trace through list-2 beginning at the previous A point entry. Since this fast trace is a sequential process, it may be expedient to keep a short table of list-2 start points which are to be traced at the end of the row when more time is available. Either way, we have established a simple method of indicating possible closure, and finding the point in list-2 where the verification trace is to be begun.

4.6.3 Euler Number Closure Detection Procedure Summary:

1. On beginning the scan of a new image row, clear two ΔE counters (ΔE_0 = 0 topped, ΔE_1 = 1 topped).

2. On scanning the row, test the 2X2 Euler window simultaneous with border point detection:

   (a) On $D5 \wedge \overline{D1} \wedge \overline{D2} \wedge \overline{D4}$ true, increment ΔE_1.

   (b) On $D4 \wedge D2 \wedge \overline{D1}$ decrement ΔE_1.

(c) On $D2 \wedge \overline{D1} \wedge \overline{D4} \wedge \overline{D5}$ increment ΔE_0.

(d) On $D1 \wedge D5 \wedge \overline{D4}$ decrement ΔE_0.

(e) On $D1 \wedge \overline{D2}$ reset ΔE_1.

(f) On $D1 \wedge \overline{D2} \wedge \overline{D5}$ reset ΔE_0.

3. Record the list-1-list-2 address of the previous point just detected at A in a short stack when:

(a) $D4 \wedge \overline{D1} \wedge \overline{D2} \wedge \overline{D5}$ AND $\Delta E_1 = +1$,

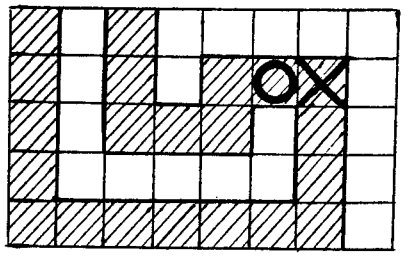
(b) $D1 \wedge D5 \wedge \overline{D4}$ AND $\Delta E_0 = -1$,

then reset the ΔE responsible for one of these expressions returning TRUE.

4. At the end of a row pop the list addresses from the stack and, one at a time, trace through list-2. Transmit the border points when closure is detected.

It should now be clear that the principal advantage of this method is its simplicity. The additional hardware required by this method is two counters of only two bits each, a short stack of maybe ten entries, and some additional logic. The exact same window registers used by the border detectors are used here.

One disadvantage of this method is that it cannot guarantee the closure of the borders it has detected. This is because of the local nature of the method. It only has a memory for the topology of the row of points it has seen up to step 3 above. It can say nothing about the body of the image, or the row points yet to arrive. A simple example would be the following:



The scan here proceeds from left to right. The 1s topped detector would flag an outer border at pixel X and the 0 topped detector would flag an inner border at pixel O. However, it is obvious that neither border is closed, and so a list-2 trace would fail. It is difficult to say how often such false alarms would occur per row. An attempt will be made to count the false alarms produced by a series of test images. However, considering that many objects are not that complex in structure and in any case, a list-2 trace can be done very quickly, the delay resulting from false alarms is not expected to be very great.

There is a possibility that this closure detection scheme will detect the same closed border more than once in a given row scan. It is therefore important on transmitting the border points to not only clear the list-2 entries during transmission, but also to check whether they are clear before starting transmission. If they are clear, the transmission algorithm can proceed to the next closed-border stack entry. Failure to check for cleared entries may lead to an infinite loop, or erroneous data being transmitted.

In conclusion, the Euler number closure detection method provides a cheap and easy method to anticipate whether a given border is closed.

4.7 Border Point Representation

To preserve generality, this discussion has deliberately avoided the question of how the border points are to be represented in storage and during transmission. Thus far, the information associated with border points has been somewhat redundant, consisting of both coordinates and direction numbers. In the interests of minimizing memory size and data-channel bandwidth, a more compact border point representation must be chosen. Basically, three methods are being proposed as feasible candidates: coordinate storage, chain code storage, and a hybrid of the two.

Representing the border points by their coordinates in the image window is probably the most obvious approach. Here only the (r, c) coordinates of each point need be stored in list-1, and once closure is detected, the closed string of these points is sequentially filed in a subsequent storage buffer. The principal drawback of this method is the large volume of data involved. For a 64X64 bit window, each r-c pair would require a twelve bit word size. Besides being long, this size is intermediate between the 8 and 16 bit word sizes in common use today.

The idea of chain code representation was introduced in the fundamental concepts. The idea is simple, instead of storing the r-c coordinates of each point, just store the edgeout direction number at each point. This number by definition points to the next pixel in the chain. At most only the coordinates of the first point in the chain need be stored. However, this is not really necessary since the readout of list-1 can start at any arbitrary location. Therefore, the start point, for instance, in

border reconstruction, can be placed at some normalized position. In either case, the advantage of chain code representation is that each point is simply characterized by a three-bit direction number. It can even be argued that if the border is gradually curved only one or two bits is needed to represent the change in direction number. The net result is that border point storage is now very compact. At most, only one quarter the memory of coordinate storage in a 64X64 image is needed.

A further advantage of this approach is that certain transformation operations on the border are greatly simplified. Translations in the X or Y directions are simply a matter of moving the chain's anchor (the first point's coordinates) about by the appropriate amount. Rotation of the border by increments of 45° simply involves adding 1 to each chain code entry per increment. However, one should be cautioned that rotations by increments other than 90° results in spatial distortions of the border. Coordinate storage, on the other hand, would require these transformations be done to every point with rotations involving 2X2 tensor operations (though without attendant distortions).

A combination of the two previous techniques forms the hybrid representation. This method of border point storage was first formulated by Zahn. He observed that borders are completely defined by those points where edgein ≠ edgeout. These he called "curvature points". This is because all those points which have edgein = edgeout lie along a straight vector directed between curvature points. Therefore, the storage of the coordinates and the edgeout of all curvature points would

completely characterize a border. This would require about 15 bits per point, but if the border forms a very angular polygon, as many characters do, then curvature points would comprise about one-tenth of the points in a border. This would yield a very compact 1.5 bits per border point. However, if the border is noisy or circular, this advantage would be lost.

The transformations applied to this representation would also be a hybrid of the two preceeding. Translations would be performed only on the coordinate component of the data, whereas rotations would be applied to both. Rotations would be particularly complex since the coordinate components can be rotated by a variety of degrees but the direction numbers by only increments of 45°. This would probably yield very irregular borders for non-90° rotations.

The hybrid representation has one strong asset that makes it the most attractive of the three. The edgein ≠ edgeout decision can be performed at the lowest level by dedicated hardware monitoring the edge directions found in the Zahn window. If edgein = edgeout, no further processing is done and that edge-point does not get logged in list-1 and the linkage tables. A great deal of processing time and memory space can thereby be saved.

Hybrid and chain coding can also provide elementary topological information about the border stored. This information in the form of change in curvature point direction number with border length could facilitate character recognition. By providing information on curvature, the degree of bend at different positions on the border can be calculated and compared

to tabulated values from characters. Normalized, such information would be size and orientation independent.

Structural information such as the area contained within a border is also obtainable from coordinates of the curvature points in the hybrid representation. These coordinates are situated at the vertices of a polygon. When read out in the linked sequence, the contained area A can be calculated with, [66]:

$$A = (1/2)\sum_{i=1}^{N-1} (\Delta c_i \, \delta r_i - \Delta r_i \, \delta c_i)$$

where:

N = the number of curvature points ,

$\Delta z_i = z_i - z_o$,

$\delta z_i = z_i - z_{i-1}$.

If area information is vital, dedicated hardware can readily evaluate the above expression.

In conclusion, it appears that the hybrid approach to border point representation and storage represents the most efficient utilization of memory, and the most useful scheme to supplement optical character recognition.

4.8 Object Reconstruction

After the detection of an exterior border confirms that a closed object has been found, the subsequent goal of a binary preprocessor is frequently to reconstruct that object in an image buffer. This reconstruction process utilizes the border points as a guide to restoring the dark object interior points as 1s. A variety of procedures can be used to implement restoration depending to a large extent on the border information available, and the means used to represent this information. The method presented utilizes both internal and external curves and makes no reference to any information source other than those curves. Topological properties of simply connected objects are explicitly employed and any of the three previous border data representations can be used as input.

The procedure presented here is intended to supplement border tracing methods, such as the non-sequential schemes, which provide complete external and internal border information. Zahn showed that such information completely characterizes the object to be reconstructed and so no further reference to the original image is necessary. Three means were presented in the previous section for representing the border points. The method to be presented here was adopted from Zahn's proof and is applicable to all three, but the chain code representation will be used for illustration.

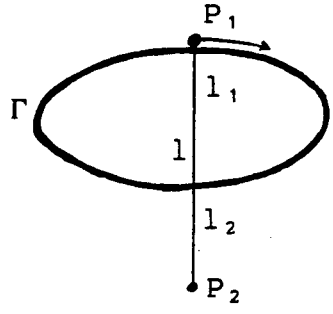
The foundation for this reconstruction method is a corrollary of the Jordan Curve Theorem [74, pp. 13-16]. This states that if P_1 is a point enclosed by a curve Γ and P_2 is some other point in the plane, then P_2 is outside Γ if a line

drawn from P_1 to P_2 intersects Γ an odd number of times. Likewise, if P_2 is inside Γ, such a line intersects Γ an even number of times.

To see how this theorem offers a means for object reconstruction, the corollary is rephrased in the following form:

If P_2 is external to Γ and P_1 is internal to Γ, then an odd number of the line segments of the line, l, between P_1 and P_2 extend from Γ to P_2. If P_1 is also external to Γ, then an even number of line segments of l can be drawn from Γ to P_2, an odd number of which intersect Γ an even number of times, including the starting point on Γ.

A simple example will illustrate how this corollary can be applied to object reconstruction. Consider a closed, simply connected object with external contour Γ. Also consider two points P_1, P_2, both external to Γ with P_1 adjacent to Γ and P_2 far displaced from Γ. A line segment drawn from P_1 to P_2 intersects Γ an even number of times:



This line is subdivided into two segments l_1, l_2; l_1 totally internal to Γ and l_2 external to Γ. Now cause P_1 to trace the outline of Γ in a clockwise fashion. During this trace, P_1 must cross l at least once. At this point, the line segment from P_1

to P_2 is identically l_2. If this trace is stopped when P_1 is adjacent to but not occupying its initial position, then the family of all line segments from P_1 to P_2 generated during the trace will include l_1 exactly once and l_2 exactly twice.

If now all points in this plane are initially labelled 0, we can assign an operator to the P_1-P_2 line segment which complements all those points on the line (i.e. changes 0 to 1 and vice versa). When such an operator accompanies the above trace, we find that, when complete, l_1 points have been complemented once yielding 1s and l_2 points have been complemented twice and so remain 0. However, these lines are simply examples of an infinite family of lines generated during the trace. The net result produced by the operator, therefore, is to convert all points interior to Γ to 1 and leave all those exterior to Γ as 0. The object points inside Γ have therefore been reconstructed.

Note also that if Γ is a curve internal to some other curve Γ_2 which does not contain P_2, then a similar trace about Γ_2 will also complement the points inside it reseting the interior of Γ to 0, revealing it to be a hole inside the larger object outline, Γ_2. This then will be the mechanism for reconstructing objects from border information.

The previous example fixed the line from P_1 to a stationary anchor at P_2. In fact, the theorem and its results apply just as well if P_2 is permitted to move freely along, say, the bottom edge of the image in such a way that the lines from P_1 are all parallel and vertical downward. In this way, all points below Γ in the reconstruction window are complemented. This permits a

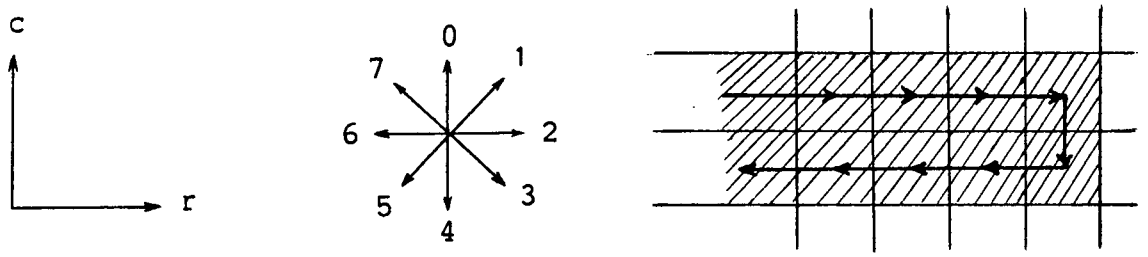
feasible proposal for object reconstruction. As the object outline is written into the reconstruction buffer, following the right-handed trace convention, complement all of the buffer points below each new point until border reconstruction is complete.

Before proceeding, however, there are two problems in adapting the theorem ignored in the example, which must first be resolved. In the example, Γ was considered part of the object and P_1 part of the background. The convention was adopted that all points on the line, 1, excluding P_1 were complemented. On close examination, we now find the curious result that when P_1 is between Γ and P_2 intersecting l_2, it encounters points that will be complemented an odd number of times (ie., set to 1). The net result is that points on l_2 adjacent to Γ are transformed to object points, essentially thickening Γ by one point over these regions. In the event that P_1 is complemented along with all pints of 1, then the thickening is simply transferred to the top portions of Γ. If P_1 was allowed to ride on Γ, the result would be that only portions of Γ would be transformed to object points. There is no consistent way to remove this behavior.

This problem was ignored in the example because P_1 was simply considered an infinitesimal point. However, when points become pixels, the problem cannot be ignored.

The other problem was that P_1 was never considered to follow a path parallel to L. That is, a path where Γ forms a straight line parallel to 1. This implies that points on 1 should be complemented an infinite number of times as P_1 moves along Γ. The result of this operation is indeterminate.

Both problems can be solved by imposing a set of rules to guide the reconstruction. The nature of these rules is readily visualized when using chain code border representation. Consider the following example displayed alongside our coordinate and direction number conventions:



To reproduce the image segment on the right by complementing downwards, apply the following rules to the current border point:

For external borders:

complement the current point if edgeout = 1 to 4,

do not complement the current point if edgeout = 5 to 0.

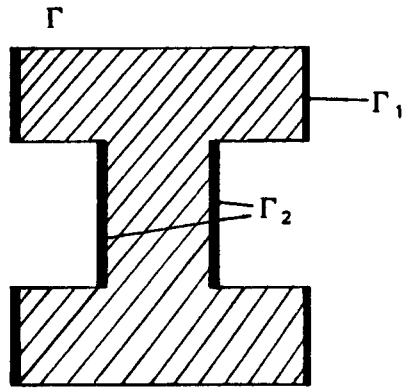
For internal borders:

do not complement the current point if edgeout = 1 to 4,

complement the current point if edgeout = 5 to 0.

The policy is reversed for internal borders because the borders still belong to the object and not the hole.

The above cures the Γ thinning problem. The problem of moving P_1 along a straight line manifests itself here as a problem of moving the trace along a vertical border segment. This is best illustrated with the following example:



Those pixels marked in bold correspond to the places where 1 is tangent to Γ at a line. These are places that either already are or will be complemented by a trace following the upper boundary. Γ can be further subdivided into two problem areas: Γ_1 representing convexities, and Γ_2 corresponding to concavities.

To eliminate the problem in the vertical lines, don't complement when:

edgein = 0 AND edgeout = 0

edgein = 4 AND edgeout = 4

The Γ_2 concavities pose a special problem since they are already complemented from above. The interiors of the vertical lines will be dealt with by the previous method, however, to prevent complementing below the corner points in the concavity, don't complement when:

edgein = 5, 6 AND edgeout = 4

edgein = 4 AND edgeout = 2, 3

edgein = 1, 2 AND edgeout = 0

edgein = 0 AND edgeout = 6, 7

Internal borders are processed using the exact same rules. Since all of these rules are simple boolean expressions, they can

be readily implemented in hardwired logic. The only additional memory burden is the storage of the old edgeout to be used as edgein above.

When reconstruction in a finite, physical buffer is considered, the problem of where to start the reconstruction traces arises. For an outer border in hybrid representation, this is not a serious problem. The buffer need only be designed to accomodate the largest expected character, and the characters are reconstructed upper-left-justified in that buffer. Two pieces of information are required. These are the column position of the trace start point supplied by the closure detector and the left-most column position encountered during transmission of the curvature points. The difference in these values will give the column position in the upper row from which reconstruction may start.

Since internal borders are detected before outer borders, two complications arise. The first is where to position them. This is best solved by storing internal border information until an external border arrives. The external border can then be reconstructed followed by the internal border(s) which start at a position determined by the difference in initial trace point cooridinates. The second problem arises when the internal border is inside an object which cannot be segmented. In this case, the correct outer border will never arrive. To avoid this internal border being added incorrectly to a subsequent object, the values of its extreme upper, lower, left and right coordinates must be compared to those of the outer border. If they all fall within the outer border's extreme points, then reconstruction can

proceed normally. Otherwise, this internal border must be discarded. Situations can be envisioned where this technique may fail. However, in character images, these situations are expected to be too rare to justify the cost of a more sophisticated method.

How the reconstructed image is handled now is up to subsequent processors. For instance, a recognizer may perform a template match on the image data and then find it satisfactory and so clear it when finished. On the other hand, it may decide something more is needed, eg., the body below the dot on an "i", and leave the image to have other parts added later, or it may order some transformation on the image such as rotation or scaling.

Since dedicated hardware can be devoted to reconstruction, the operation promises to be fast. However, many more points in the buffer must be referenced to perform complementing than there are points inside the object. Therefore, it may be essential that the time interval between successive closed objects be very much larger than the memory reference period. Fortunately in optical character recognition, this seems to be the case.

Further details concerning implementation of the reconstruction procedure will not be presented. Since such a system was not investigated through simulation, it was felt that such a discussion would be too speculative, adding little to what has been presented already.

4.9 Touching Characters

A serious investigation of the segmentation of touching characters did not form a part of this thesis research. However, since Hoffman and McCullough [62] reported up to 40% incidence of touching characters in experiments involving 12-pitch type, it was felt that the matter does deserve some discussion. Rather than attempt an exhaustive examination of possible methods to separate touching characters, only one possible solution will be presented instead. This will demonstrate that a solution to the problem could be incorporated into the preprocessor design.

A hierarchical approach is proposed for the segmentation of the text in a line. The first level of this hierarchy will address the segmentation of words, the second level, the segmentation of characters. This is a major departure from the concentration on characters that has dominated this chapter. However, the segmentation of words as a unit produces at least two notable benefits. First, words, not characters, are the elemental semantic components of language. It is the words on a page that a reading machine must articulate accurately. Secondly, the segmentation of words provides a set of measurements whereby to group and parse the characters found at the high resolution level.

Compared to characters, words are easy to segment. Even in the handwritten case, words are set apart by a clear space. The size of this space for typewritten text is at least one character-space. Therefore, there is no question of the foregoing segmentation scheme being able to segment words as a unit provided that all internal details could be forced to blend

together. From qualitative experience, it is noted that the spacing between characters within a word, and disconnected components within a character (e.g., the dot on "i"), are of a distance comparable to the average limbwidth of a character. Therefore, if the standard deviation of the $\nabla^2 g$ filter is chosen to resolve a minimum of, say, twice this spacing, all details within the word are blurred together; but words as a unit, separated by more than two limbwidths, would be resolved.

When processing this blurred image, the segmentation system may ignore internal borders, retaining only the external borders. When a closed external border is detected, the recognition system is signalled to indicate that all of the characters found by the high resolution system since the last closed border now constitute a word. The primary usefulness of this information is that the characters found can now be grouped with confidence into word units. Such a system could, for instance, distinguish between a solitary "I" representing the first person singular pronoun, and an "I" occurring in the first position of a word such as "International". Also, positional errors arising during the separation of touching characters will be prevented from accumulating beyond word boundaries. This filtering and segmentation of words must be done by a dedicated system operating in parallel with a similar system dedicated to resolving characters.

As the character boundaries are found by the high resolution system, they are passed on to the recognizer. In the event touching or poorly resolved characters occur, the recognizer should indicate a low recognition confidence. Those borders are

then stored for later processing. When a word is flagged, the spatial ordering of the characters found within it immediately becomes known. In the case of poorly resolved characters, this knowledge could be used directly or coupled with di- or trigram statistics to improve recognition confidence. Similar knowledge could also be applied to assist in the parsing of touching characters.

This parsing could proceed in the recursive manner outlined by Casey and Nagy [75]. For the case where the characters are reconstructed in a buffer, successive trials are made to split the characters at a column and send each segment to the recognizer in turn. If the result scores high confidence, including the positional and statistical information, for both characters then the segmentation is accepted. However, if low confidence is indicated, then the column is moved to increase the window on one character and decrease it on the other, and the recognition process is repeated until one of the characters is parsed too narrowly. In this case, the best guess must be taken. If more than two characters are connected, then some estimate of expected character width must be employed to select the multiple parsing columns. The method would then continue as before with, however, one column at a time being shifted to determine the optimal position.

This segmentation approach is subject to the same criticism that was directed at Hoffman and McCullough. There is no optimal way to segment overlapping characters. To address this criticism, the reconstruction step is abandoned in favor of operating on the border representation directly. The outer

border can readily provide the extreme horizontal and vertical dimensions of the character cluster. With this information, a variation of the Casey and Nagy method can be invoked. The row coordinate at a suitable division point (usually the width of the narrowest character from the left or right) is noted. The outer boundary of the cluster is then scanned to find the nearest point with row coordinate greater than or equal to this value. Once found, a second search is made for a similar point, but for which the trace is proceeding in the opposite direction. This is indicated by the edgein-edgeout pair being directed into opposite half-planes from those of the first point. If the distance calculated between these points is sufficiently close to a limbwidth then these points represent a good guess for segmentation. Should they be too far apart, then this segmentation row guess is not near the touching point and so must be shifted.

To complete the segmentation, the compound outer border is split into two smaller borders. This is done by "splicing" together the two close points just found by placing them at the head and tail of two new border point lists representing the borders on opposite sides of the splicing region. These new borders are then transmitted to the recognizer to be assigned a confidence measure. This process is outlined in Figure 4.3 below:

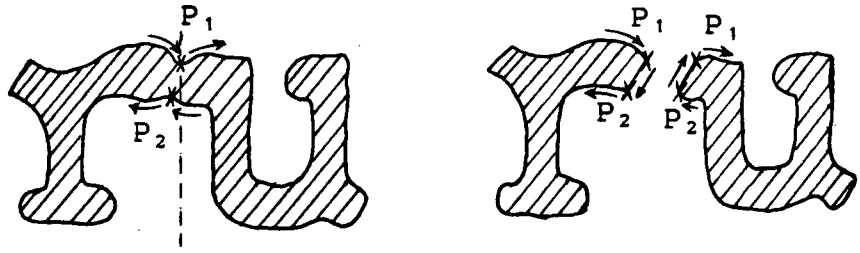


Figure 4.3 Touching character separation

If the segmented result is rejected by the recognizer, then the operation must be performed again, but with the initial row guess moved. For this reason, it is important that only a copy of the outer border data is spliced and the original left intact for subsequent trials. Note, however, that not as many recursions are expected from this approach because some prescreening already takes place in the judgement of suitable closeness between the two points to be spliced.

In the event that the characters touch at two widely separated and vertically displaced places, the above technique will reject all trial splicing pairs. An internal border will probably form in the gap between the two touching areas. This internal border must therefore be included in a subsequent repeat attempt at segmentation. This time, splicing at both places simultaneously is required.

The above has been merely a preliminary glimpse at how to address the touching character problem. It is hoped that it will serve as a fruitful starting point for further research. No doubt a more rigorous investigation, including simulation trials, will produce refinements, or replace these suggestions entirely

with a superior method.

4.10 Segmentation Summary

The complete binary-object segmentation procedure advocated in this chapter can now be summarized in the following steps:

(1) Before scanning the input, initialize all of the linkage tables, all of the list-2 entries, and the Euler number counters (ΔE_0, ΔE_1) to zero. The two image-data and the three Zahn-window shift registers are initialized to unity to frame the image bottom and sides with object points. Also, initialize the current list-1-list-2 address to unity.

(2) Raster scan the image from left to right shifting the binarized pixels into the two sets of shift registers. Begin processing the image data only after shift register S1 is filled. Maintain a count of the row and column coordinates at pixels D2, D5, and D6 with the row count cycling at a modulo at least equal to the image width. Also maintain a set of flags to indicate when the first or last image column enters D2. For the first column, black out (set to unity) D1 and D4, at the last column, black out D3, D6, and D8, and disable detection of a B point.

(3) Using the modified Zahn border point detection procedure, log the coordinate and edgeout value of all points where edgein $\neq$ edgeout into list-1, simultaneously clearing the list-2 slot. B points are processed before A points.

(4) Log the list-1 address of the current point into the appropriate linkage table where it is designated as a first entry.

(5) If a linkage is flagged as a result of the current point also being a second entry, then if the first entry being linked to is an:

(a) edgeout - log the current point's list-1 address in the list-2 slot pointed to by the first entry.

(b) edgein - log the list-1 address found in the first entry into the list-2 slot of the current point.

Clear the linkage table slot just flagged.

(6) Increment the list-1-list-2 address when processing of each A or B point is complete. When the available addresses are used up, cycle back to unity.

(7) Perform the closure anticipation operations on D1, D2, D4, and D5 to alter the Euler number counters and/or detect possible closed-border points.

(8) If a possible closed-border point is identified, then retrieve the last A point found from its delay buffer and push it onto a trace stack. It may be appropriate to maintain separate stacks for internal and external borders.

(9) When processing of the current Zahn-window is complete, place the current A point's (if there is one) list-1 address into the special delay buffer used in (8).

(10) Continue shifting the pixels and repeating steps (2) to (9) until the flag indicates the last column has been processed.

(11) At the end of the scan, pop the trace stack entries and use each to initiate a trace through list-2. Maintain a count of the number of entries traced. If:

(a) the trace leads to a null pointer, then discard the current trace stack entry.

(b) the trace leads back to the initial address, the current border is closed. Push this trace entry onto a transmission stack.

(c) the counter exceeds the maximum list-1-list-2 address, then the trace is caught in an infinite loop. Discard the current trace entry.

(12) Pop the transmission stack entries and trace through list-2 again. This time retrieve and transmit the information in list-1 to the recognition/reconstruction circuitry while also clearing the list-2 pointers. If the list-2 pointer is found to be null before the transmission is complete, abort the transmission and flag the error.

(13) On reception, the border information can be buffered and the object reconstructed when its outer border arrives, or the border information can be processed directly for recognition.

(14) Continue scanning the document until complete. Then

(a) Terminate the shifting of image data to ensure that any remaining unclosed objects are not closed accidentally.

(b) Return to step (1) to perform a complete system reset.

At first sight, this segmentation procedure may seem excessively complex to attain raster-rate performance. However, the complexity is reduced by observing that a number of these processing stages are mutually exclusive and thereby admit to parallel execution. Furthermore, this entire procedure consists largely of simple logic operations and data transfers which are

ideally suited to implementation on fast, dedicated hardware.

Once the system has begun to deliver image data, the processing of the Zahn window can be represented by the following state diagram:

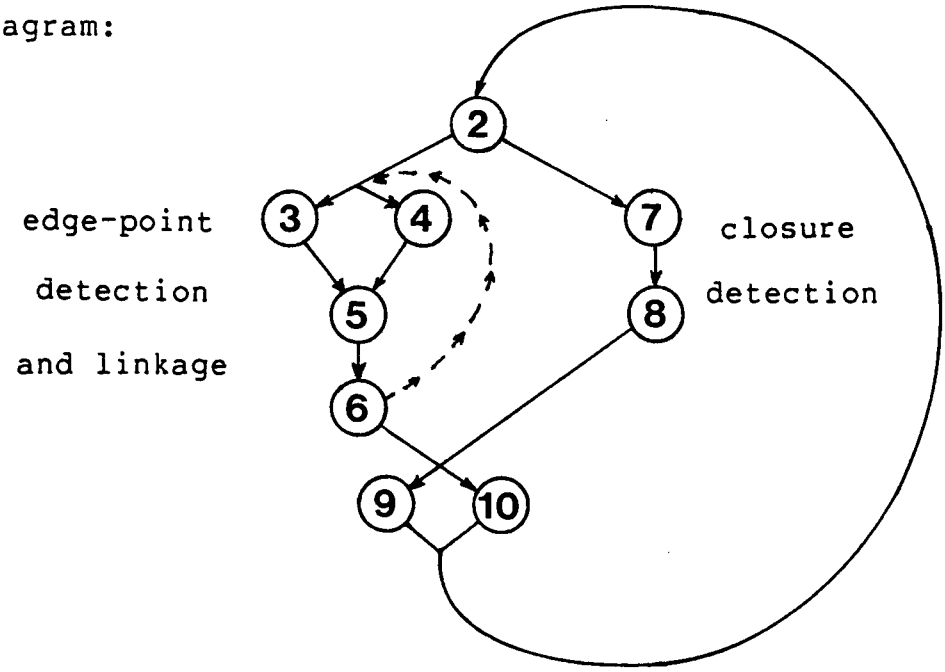


Figure 4.4 Segmentation state diagram

The edge-point detection and linkage, and closure detection sequences, are the most clearly parallel operations. There is no sharing of memory or processing resources between them. Within the edge-point detection and linkage branch, the operations of logging the edge data in list-1, (3), and logging of the list-1 address into the linkage tables, (4), can also proceed concurrently. What the dashed line attempts to show is that if both an A and a B edge is present in the current window, steps (3) to (6) must be performed twice in succession.

When processing of the window is complete, the buffering of the A point, (9), and the shift of image data, (10)-(2), can also proceed concurrently.

When a given row scan is complete, the system must dedicate

itself to the verification of boundary closure and its transmission, (11) and (12). It appears unavoidable that these steps must be executed sequentially. It may be possible to overlap these operations with those of processing the Zahn window through interleaving of the list-1-list-2 references. However, this has the one serious drawback that closed boundaries found during step (11) may become overwritten before or during transmission in step (12). It therefore appears that sufficient delay must be allowed between row scans to permit steps (11) and (12) to run to completion. What constitutes sufficient delay depends on the size of the image array, the length of list-1-list-2, and the speed of the technology employed.

The processing of border data in step (13) is an operation totally exclusive of all the others. It essentially constitutes the second processor in this image analysis pipeline, with stages (2) to (12) constituting the first. Since this stage involves no reference to the memory resources of the earlier stages, and since it is expected that there are less segmentable objects present than image rows, processing can proceed at a more leisurely pace than before. If new border data is transmitted before processing or reconstruction of previous data is complete, then it can simply be buffered until it is needed. The exact timings of this stage depend on the implementation of the border analysis/reconstruction system and on the expected complexity of the objects to be processed.

4.11 Segmentation Simulations

Throughout this chapter, discussion of the segmentation system's hardware requirements has been purposely vague. Partly this is due to the fact that these issues are implementation dependent. That is, they depend on the input image width, the maximum scan rate, and the bandwidth of the technology employed. However, it is also due to the fact that the complexity and size of the objects scanned can have a major influence on the system's requirements. Because so many contributing factors are present, it is difficult, if not impossible to analytically predict what those requirements are. Some empirical insights, however, may be gained through simulations. In this section, such simulations will be performed by implementing the segmentation system in software, and then applying it to a set of binarized images.

The simulation involved implementing the segmentation procedure, excluding reconstruction, in PASCAL to accept 128X128 binary images as input and produce a list of all segmented-border curvature points as output. Since this simulation was designed to measure such quantities as the necessary list-1-list-2 memory slots, and stack depths required by these images, these resources were made exceptionally large to ensure that they would not overflow. In the case of list-1-list-2, ten thousand memory slots were made available. The stacks were each given thirty slots.

Concurrent with the segmentation-related activity, the following statistics were also gathered:

(a) Total number of A points, B points, and simultaneous

detections of A and B points.

(b) Total number of anticipated outer and inner borders, the verified number of outer and inner borders which together provides the number of false alarms.

(c) Total number of closed border points transmitted.

(d) Image statistics: total image area; total edge points present.

(e) Record of the occupancy (number of memory slots used) of list-1-list-2 at the end of each row scan.

(f) Histogram record of the number of points traced while testing for closure at the end of each row.

(g) Histogram record of the number of segmented points transmitted at the end of each row.

(h) Histogram record of the occupancy at the end of each row of the three stacks used: stack of anticipated outer borders; stack of anticipated inner borders; and a stack for verified closed borders to be transmitted.

The choice of test images initially presented some difficulty. Foremost, these images were to present as general a set of objects as possible to provide a comprehensive understanding of the segmentation method's operation. Furthermore, these images were also to be indicative of the range of scale and image quality expected of the $\nabla^2 g$ operator. This last point required that the $\nabla^2 g$ operator be applied to binarize the input image, which in turn left open the question of the appropriate filter standard deviation. Since this system was designed to process character images, these initially seemed suitable. However, printed characters come in many shapes and sizes, so choosing any one would bias the results, and choosing a

wide selection of fonts, including all characters and a range of print sizes, would generate a vast number of test images, more than would seem necessary for the issues addressed here. Besides, it would be desirable if the results could reflect the system performance for any kind of image, including real world scenes. In the interests of generality, then, the test images should contain a complex set of borders enclosing objects of a variety of sizes, but with no particular bias to any particular class of images. Only pure noise binarized by a range of $\nabla^2 g$ filters appeared able to meet these requirements.

Fifty such binarized noise images were generated for these simulations. These were produced by initially generating ten 128X128 images containing independent Gaussian noise of mean 128 and standard deviation 16. These noise statistics were chosen simply to be compatible with those used during the evaluation trials of the last chapter. Five $\nabla^2 g$ standard deviations, σ_f, were applied to binarize the images: 0.8, 1.6, 2.4, 3.2, and 4.0. Referencing the square wave edge model, these filters will resolve edge detail with a minimum spacing (for β=1.25) of: 1.0, 2.0, 3.0, 4.0, and 5.0 pixels. These edge spacings present as wide a range of object sizes as can be expected in character, or real-world images. Since these images are derived from random noise, it was observed that these spacings are in fact close to the average values obtained. Therefore the results produced during segmentation can be classified as arising from five distinct levels of input image complexity.

Figure 4.5 reproduces two representative candidates from each of these five test image classes. The severe complexity of

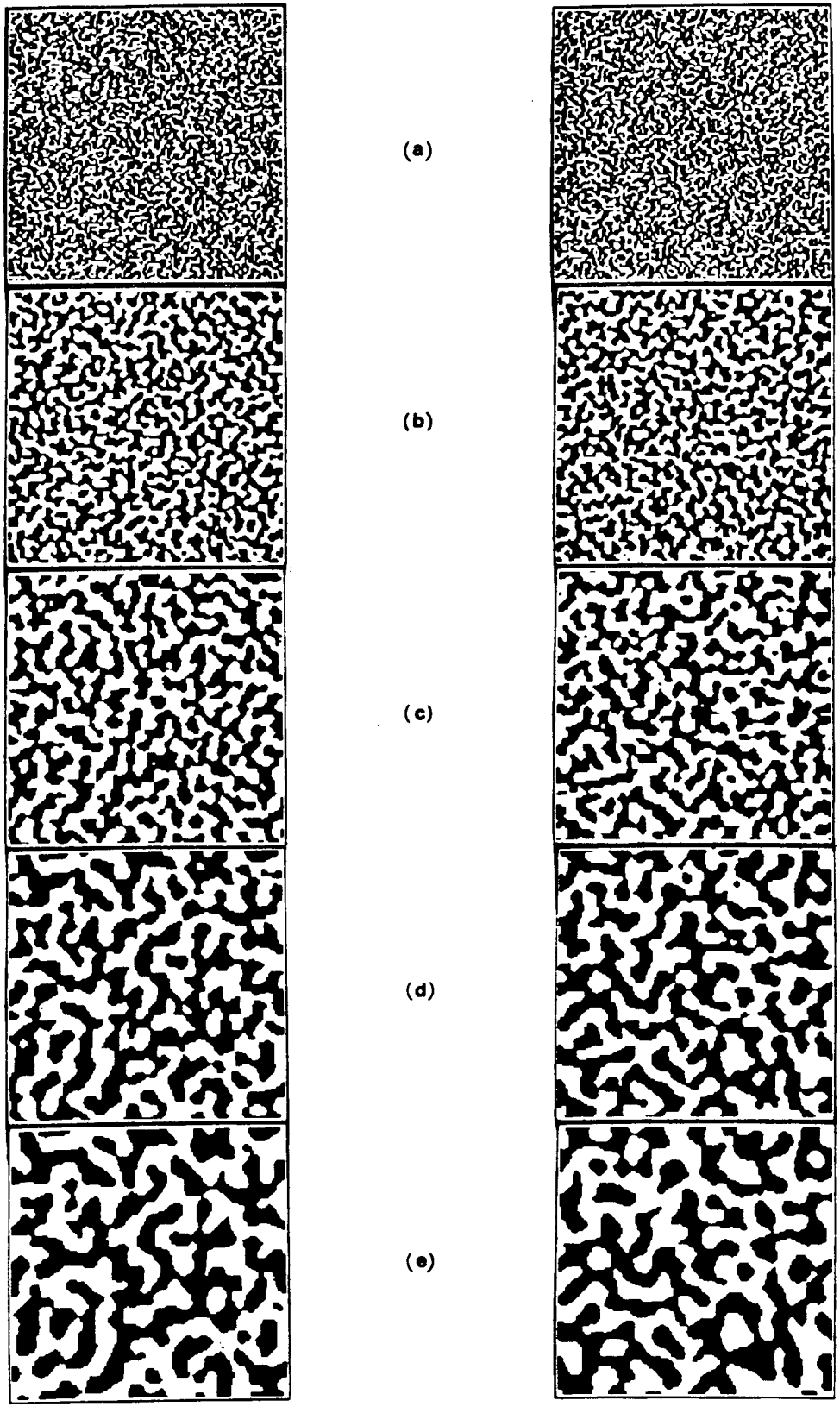


Figure 4.5 Sample test images: (a) σ_f =0.8; (b) σ_f =1.6; (c) σ_f =2.4; (d) σ_f =3.2; (e) σ_f =4.0

the σ_f=0.8 image is immediately apparent. This complexity far exceeds that expected in text images, and exceeds that of most real-world images. At the other extreme, the σ_f=4.0 image contains far larger, less structured objects than expected in text images, but it is consistent with certain low resolution real-world images. The σ_f=1.6 image is probably the most representative of the object complexity expected in text and real-world images. Though the amount of detail present is still much higher than text images are likely to produce, it contains object limb widths and boundary curvatures similar to those which compose printed characters.

Together, these five test image classes present 6400 lines of image data to the segmentation algorithm. The statistics will be grouped according to class with each grouping therefore representing the results seen in 1280 image lines. The results of the number of points traced and transmitted, and the stack occupancies at the end of each row will be combined to provide an indication of the system operating requirements across all levels of input image complexity.

Before examining the results of the simulation experiments, let's examine the system requirements imposed by a worst-case situation. Such a situation would be produced by a checkerboard image at the pixel level as shown below for even N:

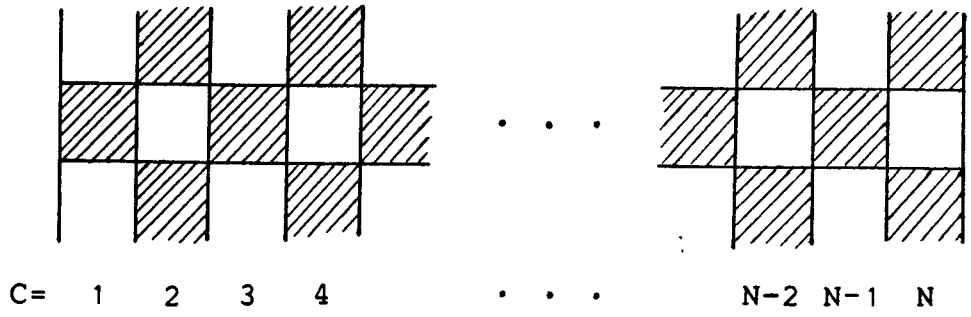


Figure 4.6 "Checkerboard" worst-case image

Even though this image has no closed outer borders, it contains a maximum density of holes, and inner borders. A number of observations can be made immediately:

(1) Every junction of two pixels is an edge point with edgein ≠ edgeout, so all Zahn window positions, except at C=N, produce both an A and a B point to be logged. Therefore 2N-1 points are logged into list-1-list-2 at the end of each scan.

(2) Every hole, except at C=N, is detected as a possible inner border; therefore the internal-border trace-stack contains a maximum of N/2 entries at the end of a row scan.

(3) With four border points per hole, an average of (2N - 2.5) points would have to be traced and 2N-4 points transmitted at the end of each row.

A maximally detailed image containing only closed outer borders would be less dense than the above worst case, but the densest rows will contain the same number of closed borders and border points as (2) and (3) above. It is important to note that it is readily possible that a given row scan in another image may close a border that contains more points to be traced and

transmitted than indicated in (3). Substituting N=128 will permit comparison of this worst-case for the results actually obtained.

Table IV presents some of the results obtained alongside those of the worst-case situation. The decreasing image complexity with increasing σ_f is reflected throughout the table, most notably in the decrease of edge points despite a nearby constant dark image area. The worst-case is revealed throughout to be an unrealistic model about which to design the system. It is also seen that the σ_f=0.8 image class is sizably more demanding of the system than the other classes.

It is observed that in general, the A and B points occur with approximately equal incidence at a combined rate ranging from 0.58 points per column (ppc) down to 0.08 ppc. Simultaneous occurence of A and B points also declines from 24.8% of the points seen down to 3.2%. These results stand in contrast to the worst-case figures where 2 ppc are produced with essentially 100% simultaneous occurence.

The curvature point arrival rates suggest that there may be some merit to pipelining the edge-point detection and linkage process. In the first stage of such a pipeline, the curvature points would be detected and logged in list-1. In the second stage, the list-1 addresses of the curvature points would be buffered for linkage processing. Provided that this system could process an average of about 0.5 ppc (or one point every two pixel shifts), all points seen should be properly linked shortly after the end of the row scan, minimally delaying the subsequent closure verification stages. Such an implementation would permit

	Worst -case	σ_f =0.8	1.6	2.4	3.2	4.0
area	8192	8186.6	8196.7	8191.1	8190.9	8153.2
edges	32512	14645.7	7710.7	5229.0	3930.9	3183.1
edges/row	225	114.420	60.240	40.852	30.710	24.868
A pts	128	37.275	15.194	9.009	6.238	4.905
B pts	127	36.887	15.156	9.094	6.433	4.921
Total	255	74.162	30.350	18.103	12.671	9.827
A AND B	127	9.198	1.759	0.648	0.291	0.159
anticipated outer	0	2.438	1.144	0.588	0.347	0.242
anticipated inner	63.5	6.038	1.369	0.672	0.377	0.255
actual outer	0	0.271	0.186	0.096	0.044	0.043
actual inner	63	3.484	0.347	0.105	0.046	0.030
false alarms	0.5	4.720	1.980	1.059	0.634	0.423
closed pts transmitted	252	56.796	10.670	3.932	2.053	1.963
unclosable pts	3	17.366	19.680	14.171	10.618	7.863

Table IV. Segmentation simulation results including "checkerboard" worst-case

use of slower hardware and would be more suited to cope with those few instances where A and B points arrive simultaneously.

The results for the closed borders anticipated by the Euler number closure detection method show that the σ_f=0.8 image class exhibits a 2.5:1 bias in favor of internal borders. This may be a statistical anomaly since the bias rapidly disappears for the other classes.

It is observed that there is an almost order of magnitude decrease in the borders anticipated per row from the worst-case (63.5) to σ_f=0.8 (8.5) followed by a further order of magnitude decrease by σ_f=4.0 (0.50). Of the borders anticipated, the percent that can actually be closed and transmitted ranges from 44% (σ_f=0.8) to 15% (σ_f=4.0) in contrast to 99% for the worst-case situation. It would appear that the efficiency of the Euler number closure anticipation method declines markedly with a decrease in image complexity. However, this apparent decline could also be attributed to the increase of unclosable curvature points in proportion to those forming closed borders with increasing σ_f. This causes a decrease in the number of closed border points to total curvature points ranging from 76.6% (σ_f=0.8) to 20.0% (σ_f=4.0) which lends support to the premise that the method's efficiency is not strongly related to the image's complexity.

The largest data structure in the system is the dual list-1-list-2 array. It is important that this array log all the curvature points likely to be encountered in an image window before recycling. Otherwise, segmentable objects within that window may become over-written before being detected as closed.

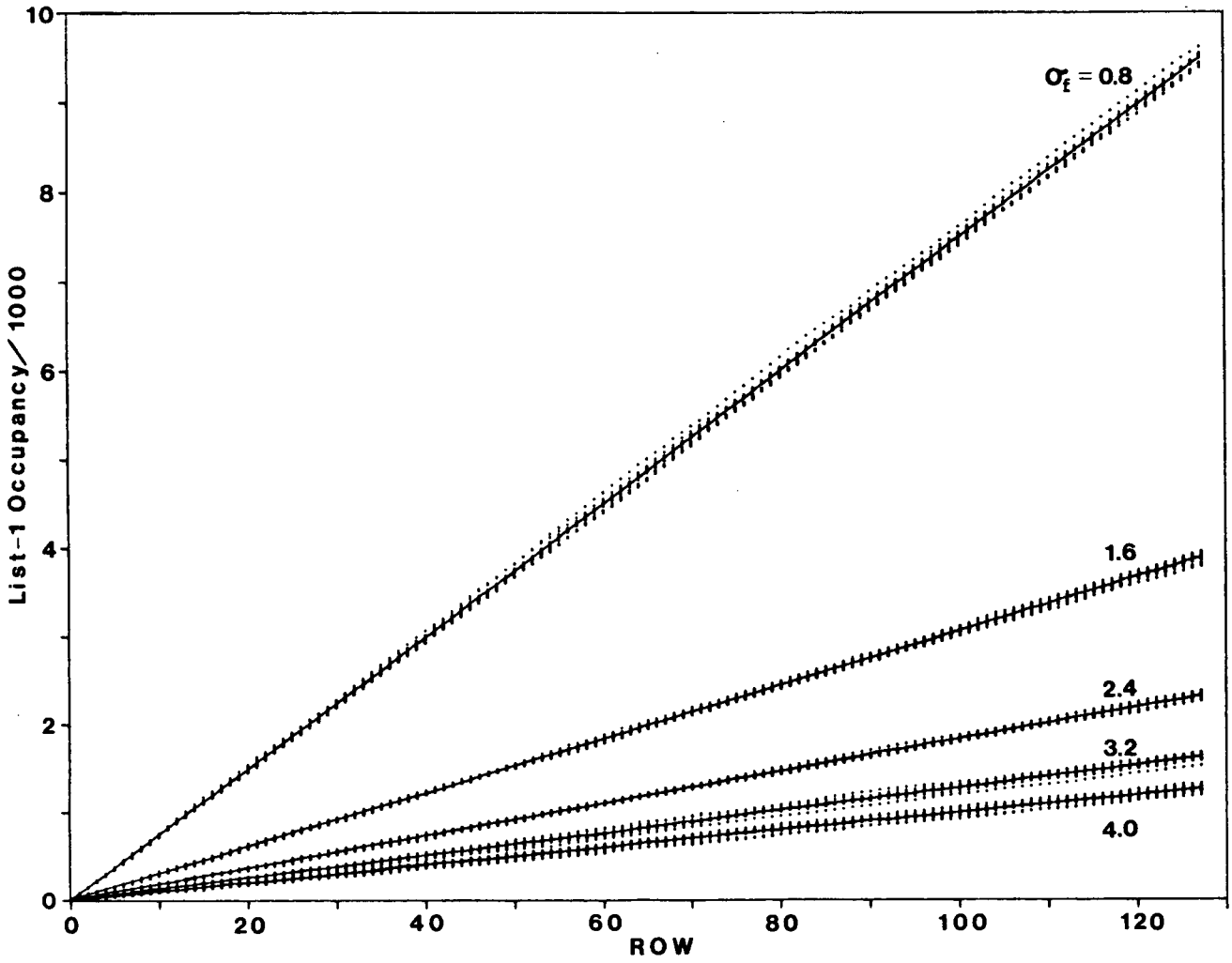


Figure 4.7 List-1 occupancy against row count

A record of the occupancy of list-1 at the end of each row for all of the test images is shown in Figure 4.7. There are two important points to note about this plot. The individual points, representing the results of a given image, scatter very narrowly about the lines representing the average occupancies per class. The average occupancy, therefore, serves as a reliable indicator of the list-1-list-2 requirements of a given resolution class. Secondly, the occupancy shows an almost exactly linear dependence on the row count for a given class. This implies that for a given image complexity class, the list-1-list-2 size requirement scales directly with the image area. Therefore, a 64X64 image window needs only one quarter of the list-1-list-2 memory space of the 128X128 images.

The maximum values attained in Figure 4.7 represent the list size requirement for that image class. It does not mean, however, that objects are present with a size that extends across the entire length of the image. Rather, it means that if such objects were present, these list sizes represent the minimum required to guarantee their segmentation. The requirements extend from about 9500 memory entries at σ_f=0.8 to 1300 entries for σ_f=4.0, with 3900 entries for that expected from printed characters at σ_f=1.6. The requirements for the worst-case equal that of the number of edges present at 32,512 entries. The choice of which list size to adopt must be motivated largely by cost. That is, the cost of the memory hardware, as compared to the cost of losing the occasional segmentable object. It is evident from the narrow scatter of the results in Figure 4.7 about their average values that the σ_f=0.8 requirements represent a maximum that is unlikely to be exceeded. Indeed, even the

worst-case requirements are deceptive since the image is dominated by single pixel holes, so that segmentation is actually guaranteed after only 383 entries.

The efficiency of logging only curvature points as opposed to all edge or border points is evident on comparison of the maximum values of Figure 4.7 to the total edges present in these images as given in Table IV. As a percentage, the curvature points range from 65% of the total to 41%. This indicates that, in general, the intermediate storage requirements of this form of border representation are about one half those of a more complete representation that logs every point. Consequently, also, half the delay while verifying closure and transmitting borders can be expected.

In order to estimate the delay incurred at the end of each row scan, the number of points traced and transmitted at the completion of each row were recorded. The results are presented in histogram form in Figures 4.8 through 4.13. Since the closure verification trace and border transmission operations occur in sequence, the steps taken in these operations should be summed to provide a basis for a delay estimate. The maximum sums seen range from 1122+581=1703 for σ_f=0.8 to 298+130=428 for σ_f=4.0 . Interestingly, the steps counted for σ_f=1.6, 1177+411=1588, are not much different from the otherwise more complex σ_f=0.8 image. The worst-case values of 255+252=507 are misleadingly small due to the edge-points being produced by a large number of single pixel holes rather than extended features.

Examination of the histograms reveals that the maximum trace and transmission counts are rare events of a magnitude far above

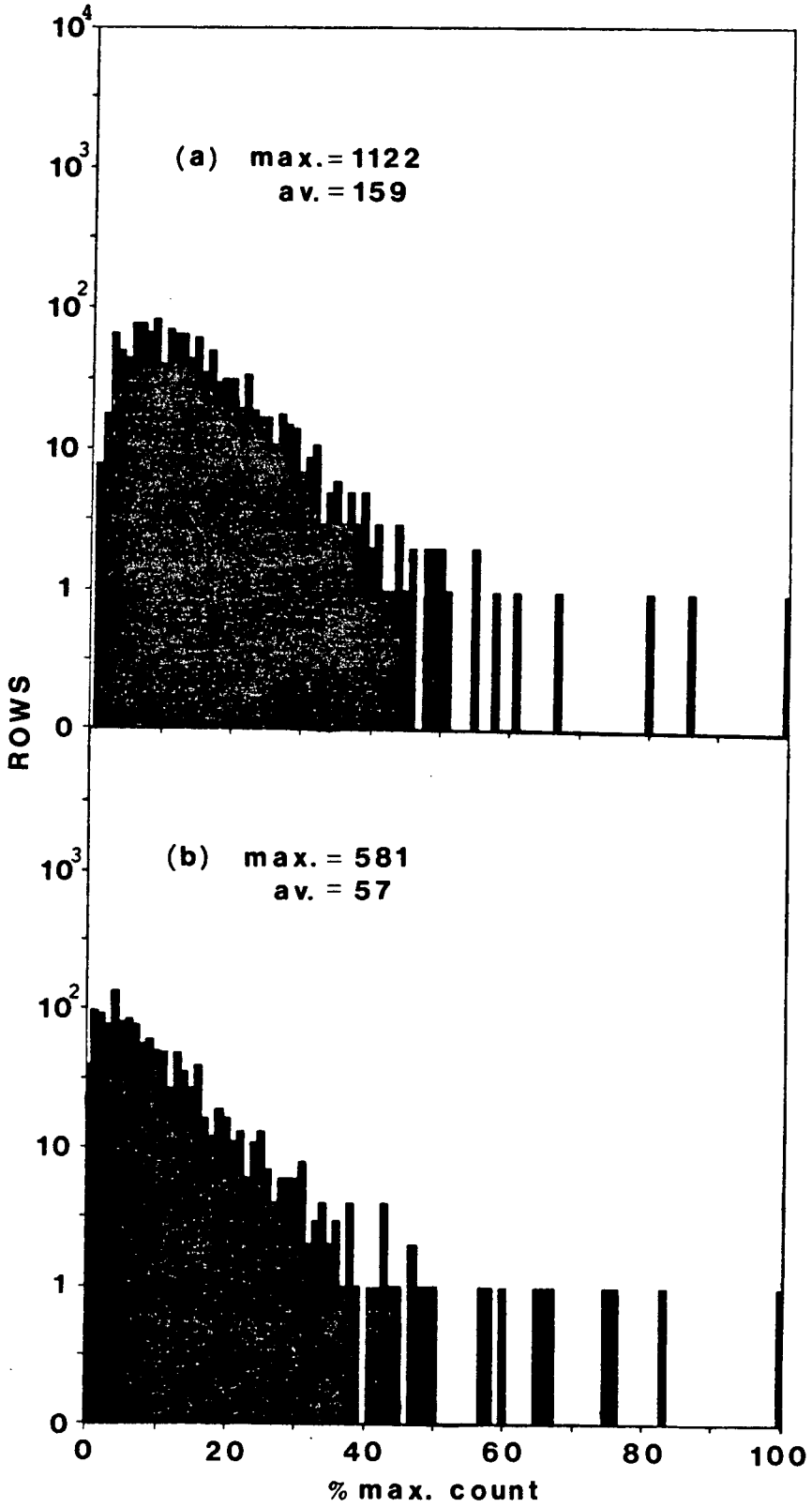


Figure 4.8 σ_f =0.8 histogram: (a) number of points traced per row; (b) number of points transmitted per row

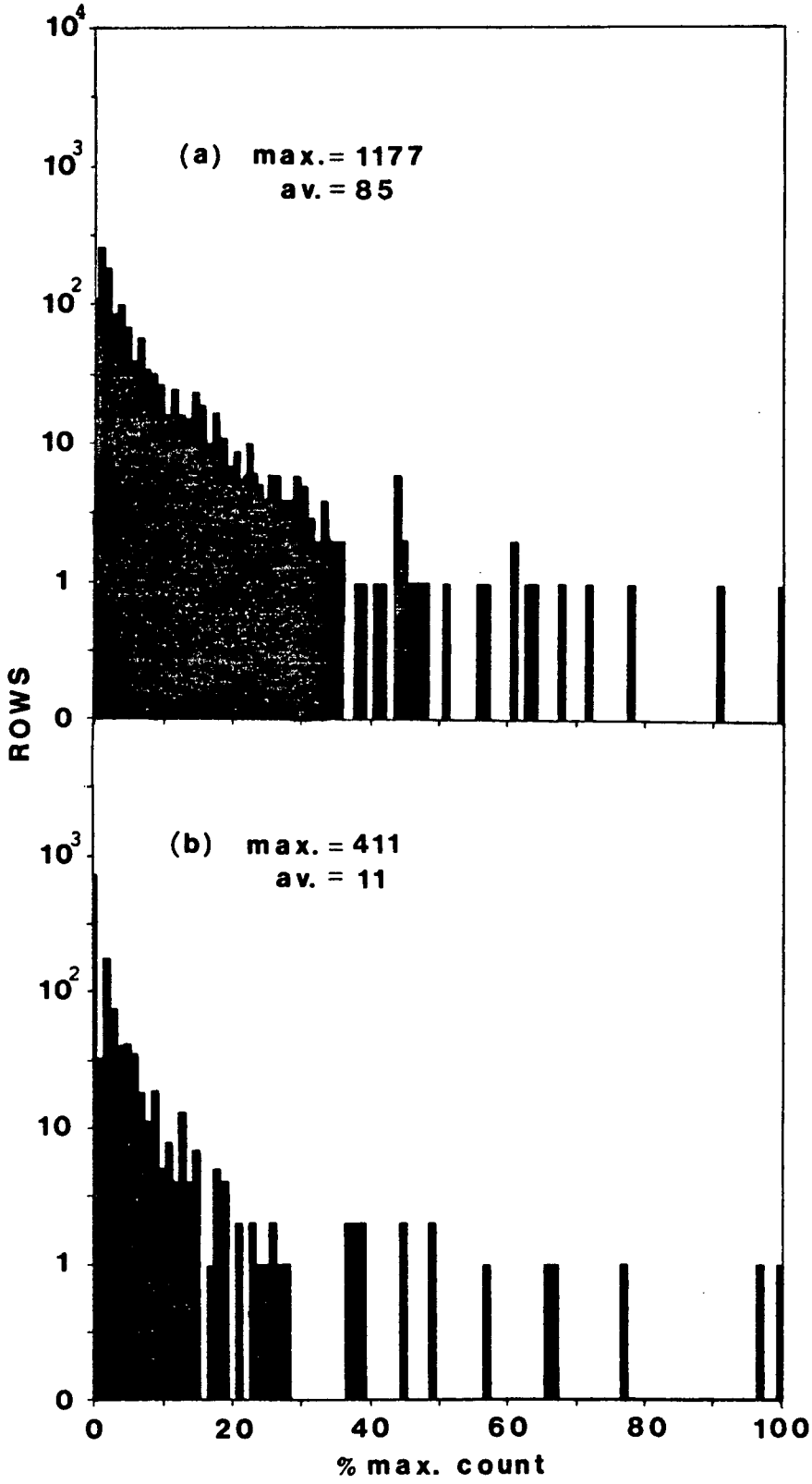


Figure 4.9 σ_f =1.6 histogram: (a) number of points traced per row; (b) number of points transmitted per row

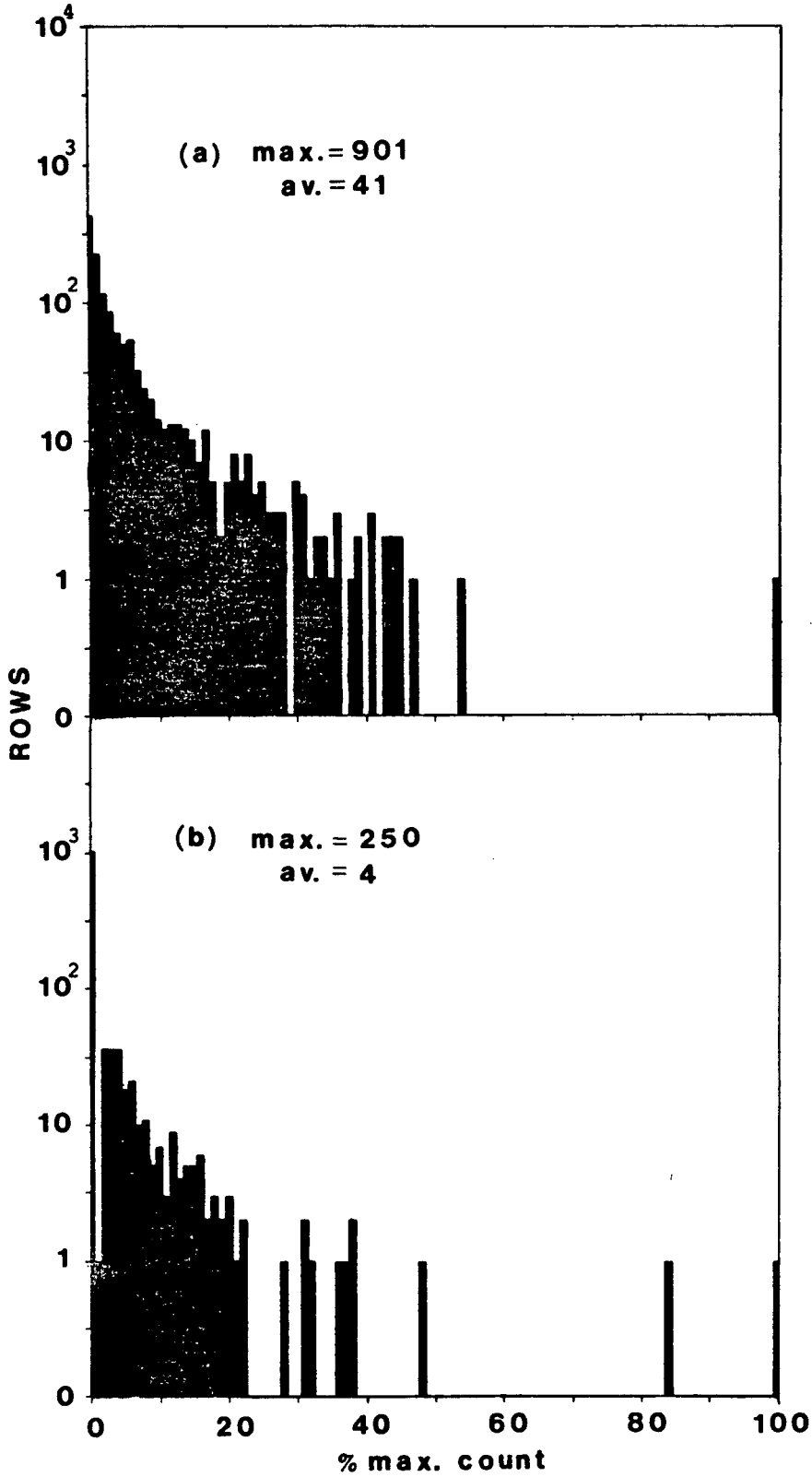


Figure 4.10 σ_r =2.4 histogram: (a) number of points traced per row; (b) number of points transmitted per row

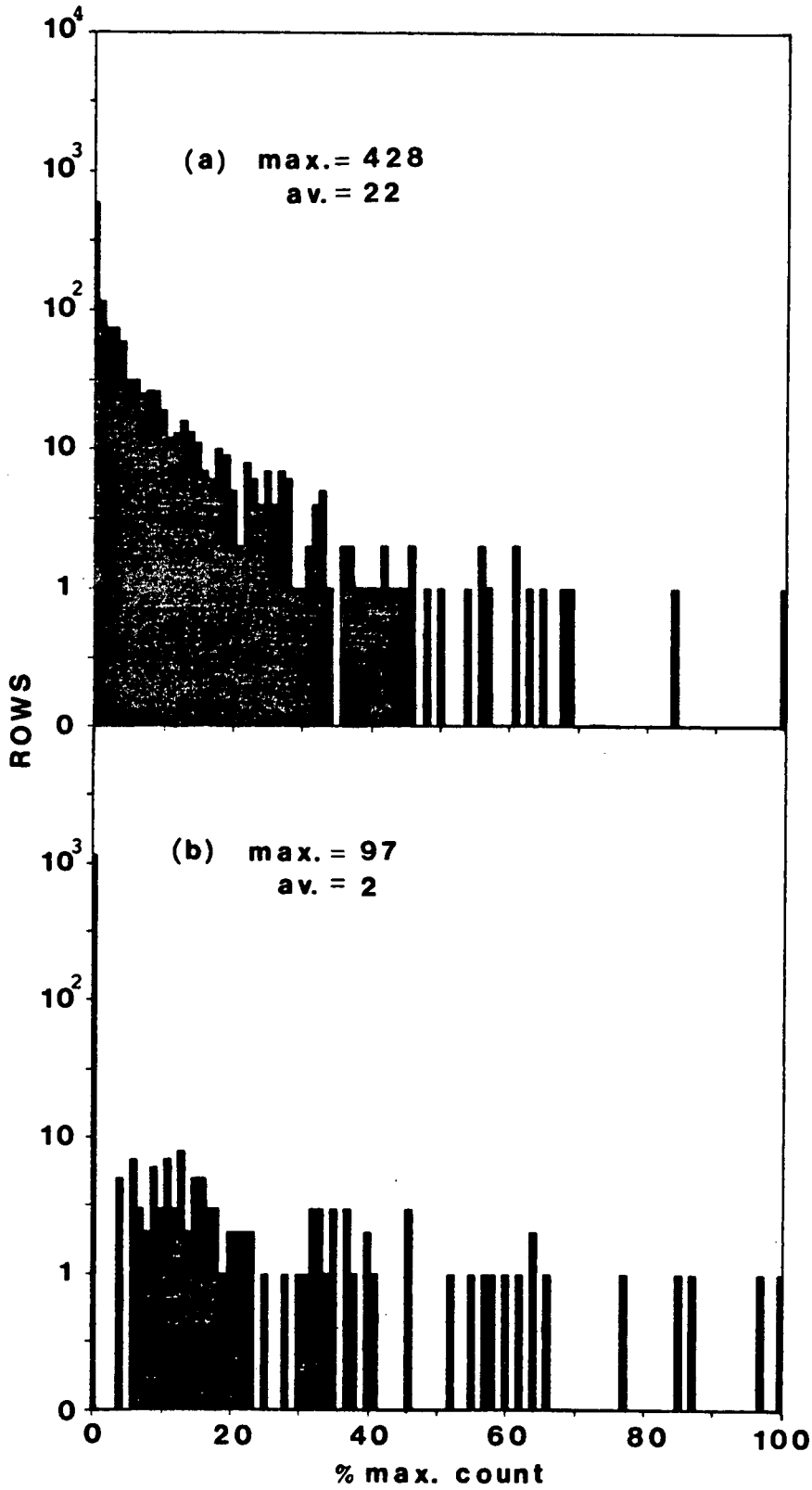


Figure 4.11 σ_r =3.2 histogram: (a) number of points traced per row; (b) number of points transmitted per row

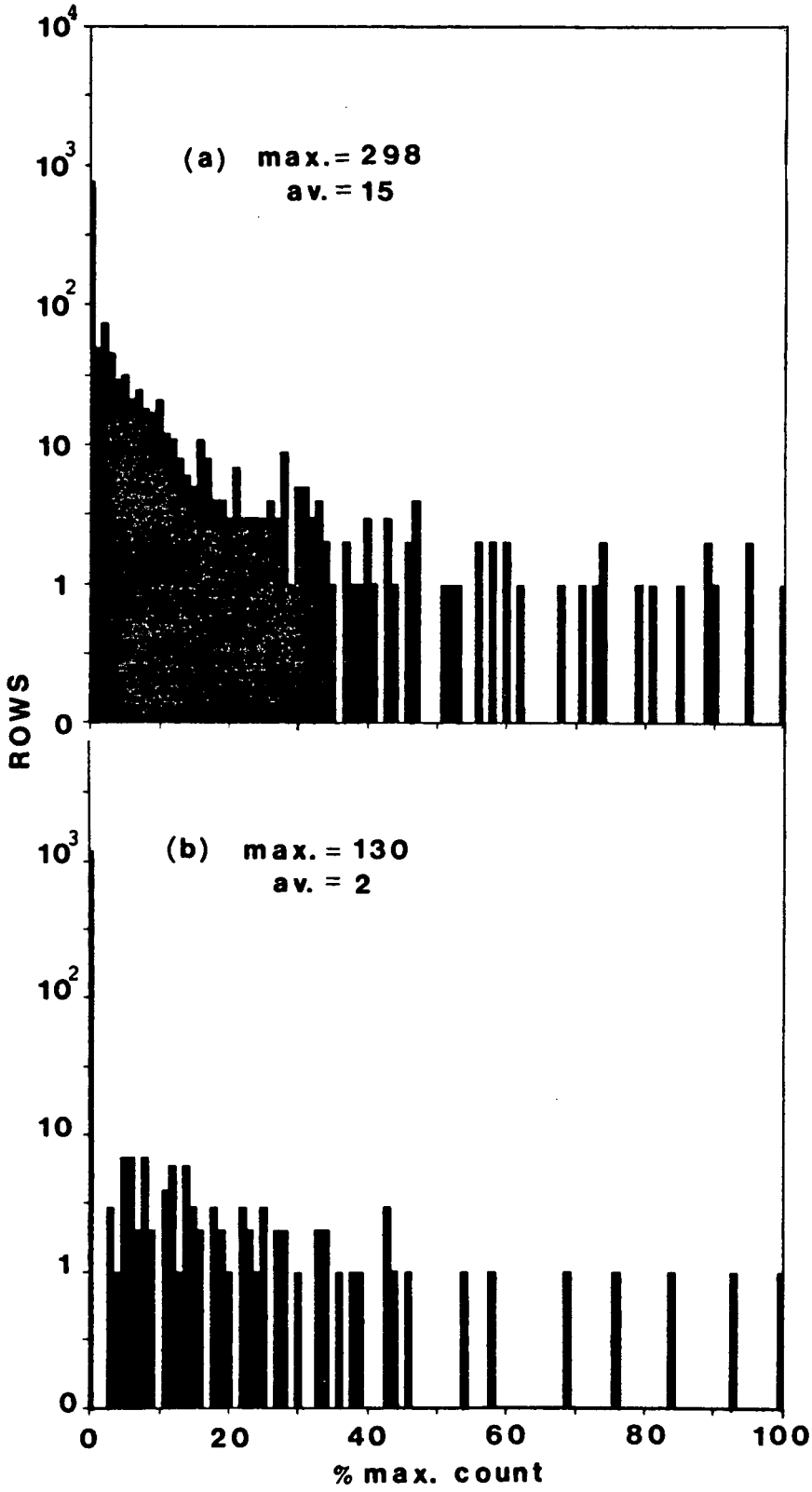


Figure 4.12 σ_f =4.0 histogram: (a) number of points traced per row; (b) number of points transmitted per row

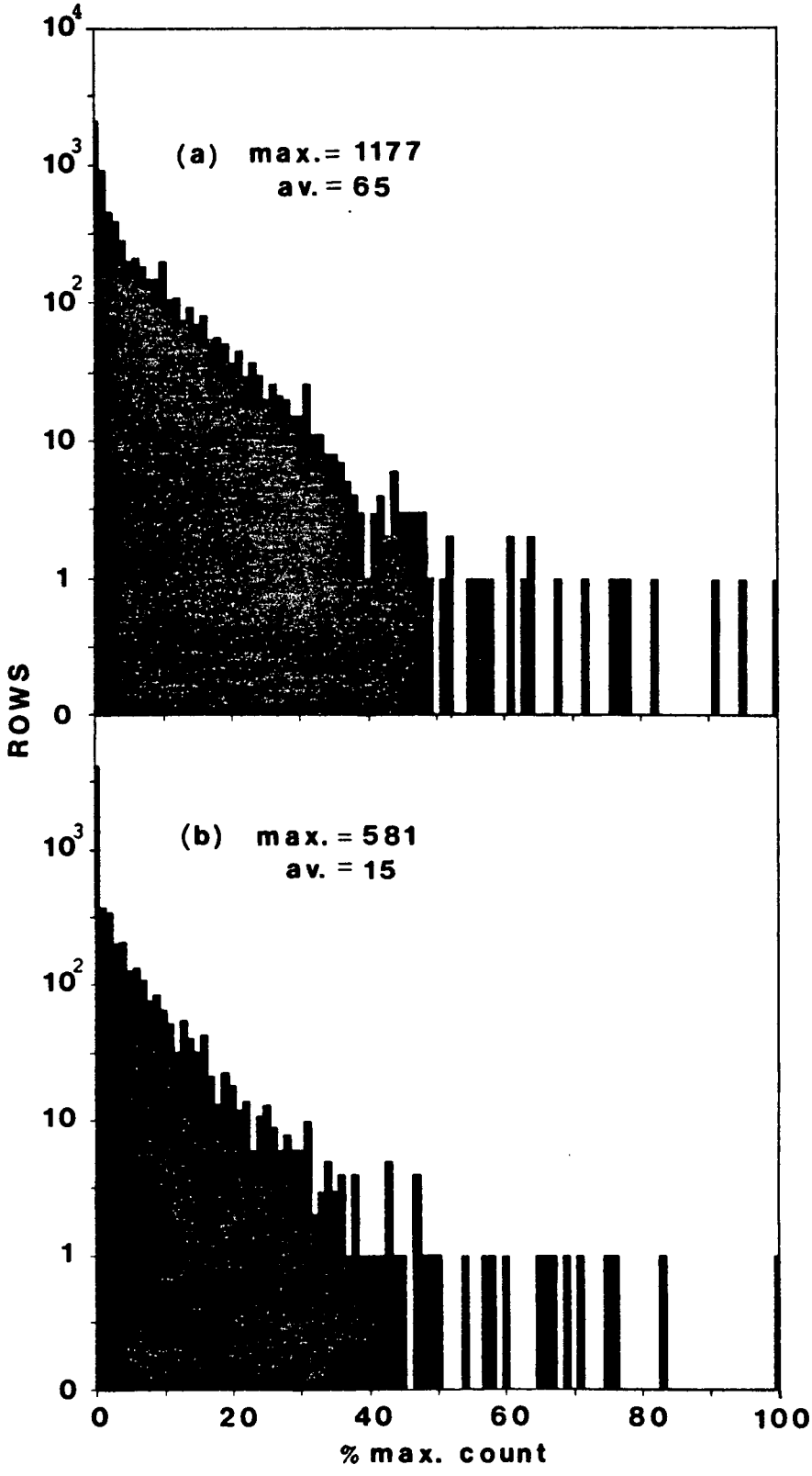


Figure 4.13 Total combined histogram: (a) number of points traced per row; (b) number of points transmitted per row

the mean values. In fact, the mean values are generally less than 10% of maximum. This is most succinctly illustrated in the histograms of Figure 4.13 which combine the results of all the image classes. The maximum count of points traced after a row was found to be 1177 but the mean was 64.5, 5% of the maximum. Similarly, the maximum points transmitted was 581 but with a mean of 15.1, only 2.6% of the maximum. However, the processing algorithm presented required that all points be traced and transmitted at the end of a given row scan. This requires designing the system solely about the maximum points expected to be encountered at the end of a given row scan. These simulations suggest that to be 1177+581=1758 or, with a safety margin, approximately 2000 points. The hardware employed must therefore be capable of processing 2000 memory references in the time between single row raster scans.

An alternative approach allows advantage to be taken of the much lower mean values by abandoning the requirement to complete the closure verification and border transmission tasks between scans. This approach must, of course, involve concurrent execution of these tasks with the curvature-point detection and linkage task. As a result, the system architecture is further pipelined. The earlier observation that at most 0.5 curvature points will be detected per raster shift permits parallel execution of tracing and transmission with detection and linkage since list-1-list-2 memory references can be interleaved. The principal objection of this approach was that the entries being traced or transmitted could be overwritten and lost due to new points entering list-1-list-2. However, the list-1-list-2 size requirements are now understood, and seen to be very predictable

for a given image complexity class. Increasing the list size above what is required for this class can prevent unwanted data loss during parallel execution.

For example, the σ_f =1.6 image class was seen to require at least 3900 list-1-list-2 memory positions with about 30 positions being filled per row scan. Even though the maximum points traced and transmitted per row was 1177+411=1588, the means were 85+11=96. Therefore even if the trace-transmit system averaged only one memory reference per shift, with no processing between scans, it would generally complete its task during the subsequent row scan. However, the rare maximum case would require 1588/128=12.4 raster scans. Therefore, to accomodate this eventuality the list-1-list-2 size needs to be increased by 12.4X30=372 entries, or about 10%. Since the single memory reference per raster shift may be unrealistically slow, and there may be time for processing between scans, the actual memory increase may be much less. Therefore, the cost saving of such a fully parallel system as associated with slower hardware and greater assurance that all operations will run to completion without conflict may be considerable.

The only serious side-effect resulting from parallel execution of tracing and transmission is that the stacks storing the pointers to possible internal or external borders need to be referenced by both the tracing and closure-anticipation systems. Since, even for the σ_f =0.8 image class, new entries for these stacks arrive relatively infrequently memory reference conflicts pose no serious difficulty. The difficulty arises from the tracing system requiring access to the oldest points on the

stacks while at the same time the anticipation system is adding the newest points. There are basically three solutions to this problem:

(1) Ignore the problem and process the stacks as usual. Since on average, the borders must be traced faster than detected, the stacks will always be emptied eventually. However, the addition of new points will add to the delay in reaching the stack bottom which in turn may require a sizable increase in list-1-list-2 memory to prevent these old borders from being lost during the interim. Also, the borders will be transmitted all out of order in relation to their causal appearance creating havoc with the border post-processing system.

(2) Employ multiple sets of stacks with the closure anticipation system switching to a new set at the begining of each new row scan. The closure verification system then processes the stacks in the usual way from the oldest to the newest. The stacks are recycled when the last in the series is filled. The number of extra stacks needed is determined in exactly the same manner as the extra list-1-list-2 memory was determined: the number of image rows scanned while processing the row with the maximum points to trace and transmit equals the number of extra stacks needed. In the previous $\sigma_t = 1.6$ example, that number would be 13 sets (rounded up from 12.4). This method has the attraction of processing the anticipated borders in the usual manner without modification, except for the addition of a system to change the stacks. The only drawback of the system is that a great deal of the memory reserved for the stacks will be wasted since the stacks are seldom used to full capacity.

(3) Replace the stacks with a set of circular lists. The closure anticipation system will now place new pointers on the top of each list while the closure verification system removes old pointers from the bottom. The causal order of borders traced and transmitted will remain correct and the system used to control the lists is similar to a stack except that two pointers are maintained for the top and bottom of the lists, and these must be cycled in a finite block of memory. The amount of memory required for each list would be determined by the product of the mean occupancy of its corresponding stack and, as before, the number of image rows scanned while processing the row with the maximum points to trace and transmit, plus a suitable buffer to accomodate large deviations from the mean. For example, if the mean occupancy was 2 and the row count, for $\sigma_f = 1.6$, was 12.4, the requirement would be about 25 entries. Including a buffer of 10 gives a total size of 35 entries. Clearly, this is not a very demanding memory requirement.

The drawback of this system lies in maintaining the correct causal relationship between internal and external borders. Internal borders generally belong within the temporally next closed outer border. The reconstruction or analysis system may rely on this causal ordering to correctly process the borders. When tracing and transmitting all points between scans, this ordering was guaranteed. However, to ensure this ordering using circular lists, the system must monitor the values of the pointers stored in those lists. The most recent pointers have the highest list-1-list-2 addresses, cycling at a fixed modulus. The circular list for internal borders must therefore be read

first until the next pointer found exceeds the pointer of the next external border pointer. The external border list will then be processed until the next pointer exceeds the pointer of the next internal border, and so on. The closed borders will then be detected in the correct order for transmission. Unlike the previous cases where closed-border pointers were stacked to await transmission, after the closure verification stacks had been read, this system can transmit the borders as soon as closure is verified. This is because these circular lists cannot be fully read in a guaranteed interval of time since new pointers are always being added. Also, immediate transmission of closed borders provides extra assurance that the data in list-1-list-2 will not become overwritten. This simplified transmission system may cancel out the extra complexity required by the verification system.

Which of approaches (2) and (3) is favored depends largely on the relative cost of the hardware employed. Approach (3) seems the most elegant, and the immediate transmission of closed borders is very attractive. However, system (2) is basically a simple extension of the earlier stack processing method, but the necessary waste of memory it would entail may cause it to be economically unattractive. The amount of memory required by each of these systems can be estimated by the stack occupancies found during the simulations.

The simulation used a stack each for anticipated inner and outer borders and a single stack to store pointers to verified closed borders. Each stack was read and emptied after each row scan and the number of entries found recorded. The result was a

record of the stack occupancy in terms of the number of rows generating a given, number of entries within each image class. The results are shown in histogram form in Figures 4.14 to 4.19. The most remarkable observation is that none of these stacks is very long. The worst-case example would have required up to 64 internal-stack positions and 63 transmission-stack positions. However, the worst case in the simulations, σ_f =0.8, Figure 4.14, requires only a maximum of 14 positions for the internal-stack and 13 for the tranmission stack. The σ_f =4.0 findings, Figure 4.18, show only three positions to be necessary for each of the three stacks. This small memory requirement for the stacks in any realistic situation is not worth optimizing. To declare a minimum of twenty entries to be needed for each stack would certainly prove adequate and provide a comfortable safety buffer for unexpectedly complex input images.

For a simple three stack system without parallel processing of verification and transmission, these stacks would only occupy 60 memory locations. For a parallel system with multiple stacks, case (2), the example given would require 780 locations. Optimization could now be considered, but this memory size is still not serious. Design of the circular list structures required an estimate of the average stack memory requirements. Figure 4.19 presents the combined occupancy results for all image classes. The means for the internal-, external-, and transmission-stacks was found to be 1.8, 1.0, and 1.0. These results are roughly comparable to the σ_f =1.6 results but less than the σ_f =0.8 results. Rather than adopt the overall mean, for illustration the σ_f =0.8 figures will be used, but with the external-stack mean equal to the internal-stack mean since their

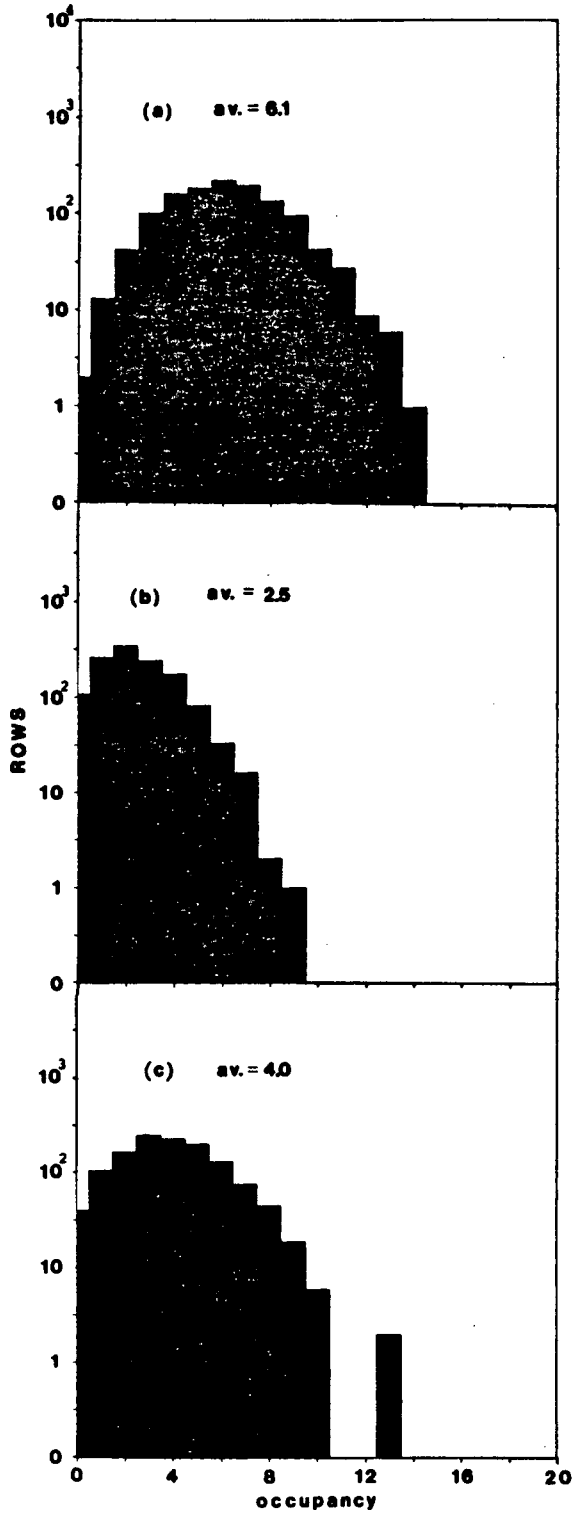


Figure 4.14 σ_r =0.8 stack occupancy: (a) external border stack; (b) internal border stack; (c) transmission stack

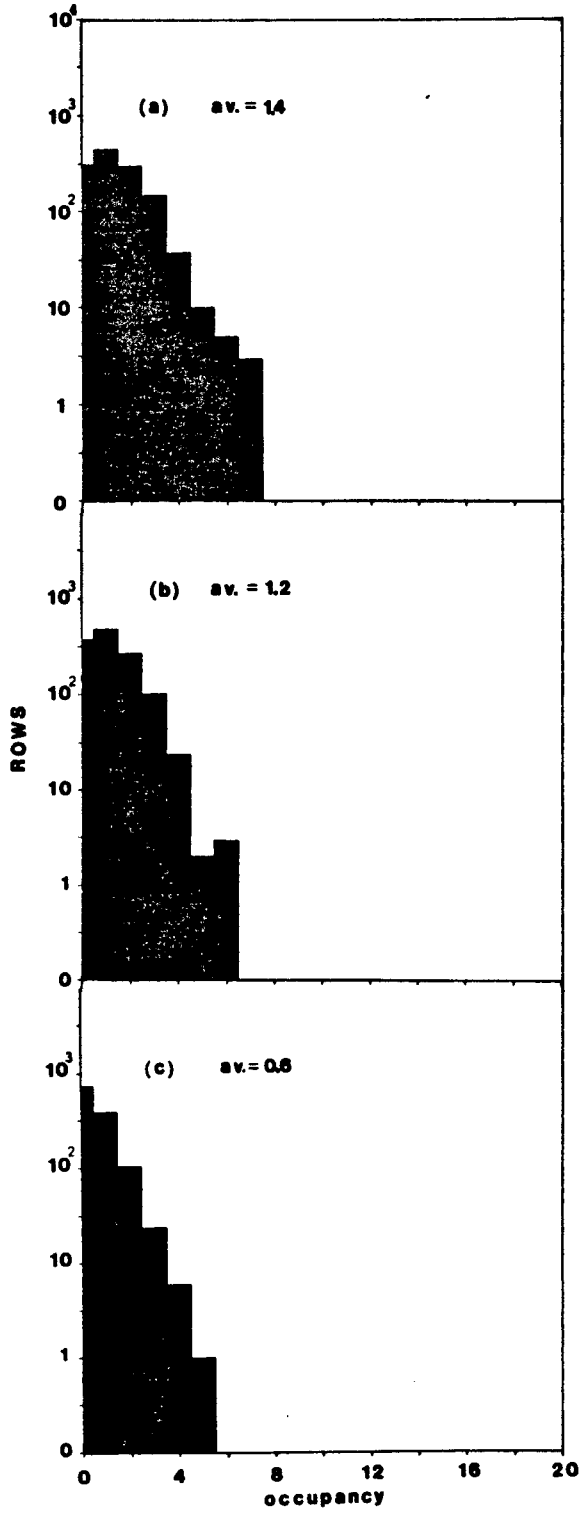


Figure 4.15 σ_f =1.6 stack occupancy: (a) external border stack; (b) internal border stack; (c) transmission stack

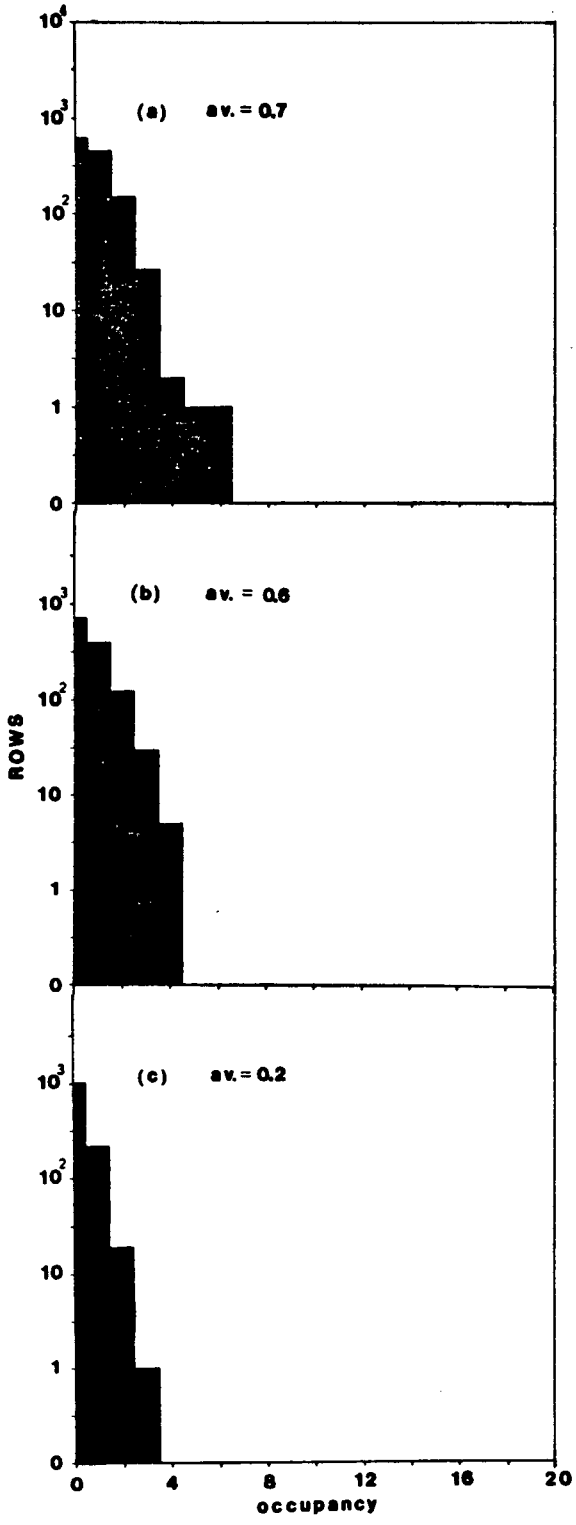


Figure 4.16 σ_r =2.4 stack occupancy: (a) external border stack; (b) internal border stack; (c) transmission stack

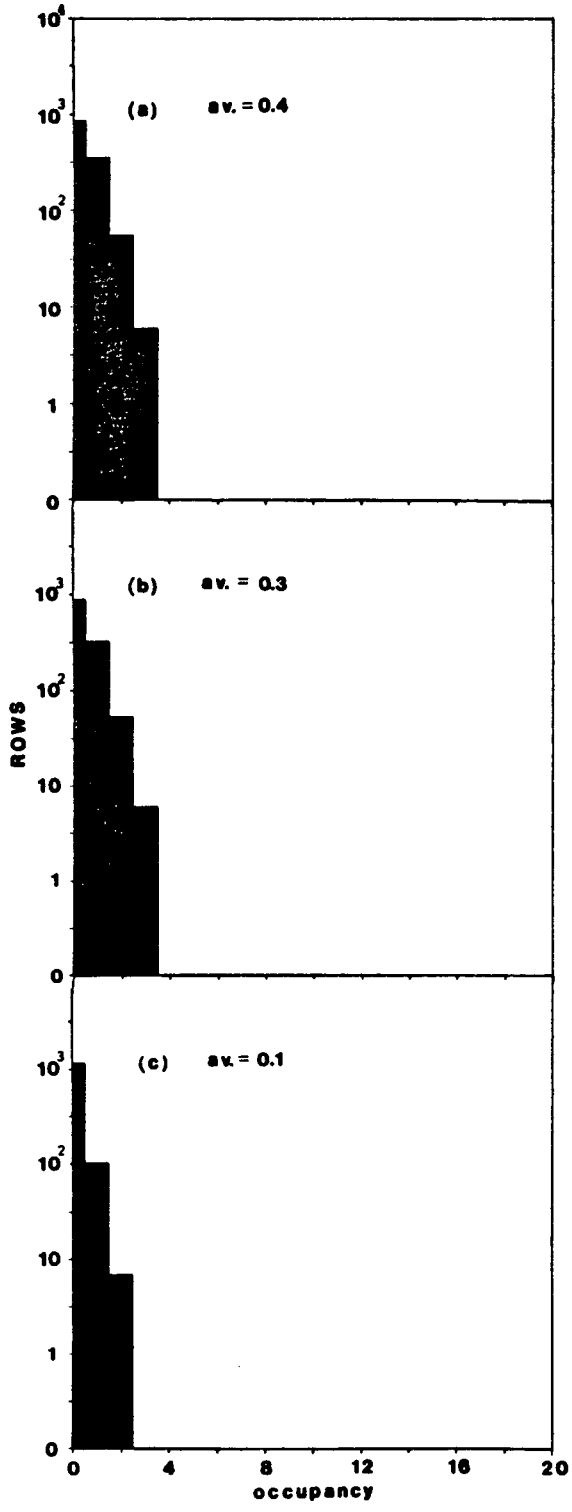


Figure 4.17 σ_r =3.2 stack occupancy: (a) external border stack; (b) internal border stack; (c) transmission stack

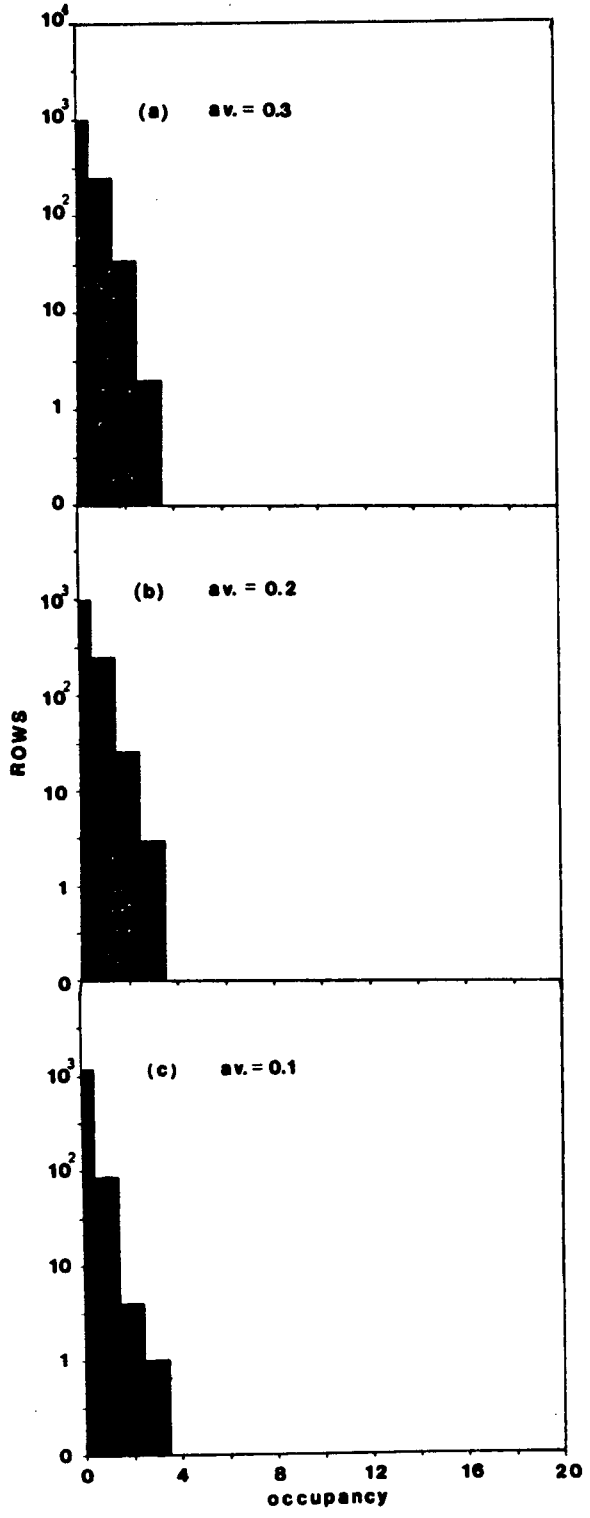


Figure 4.18 σ_f =4.0 stack occupancy: (a) external border stack; (b) internal border stack; (c) transmission stack

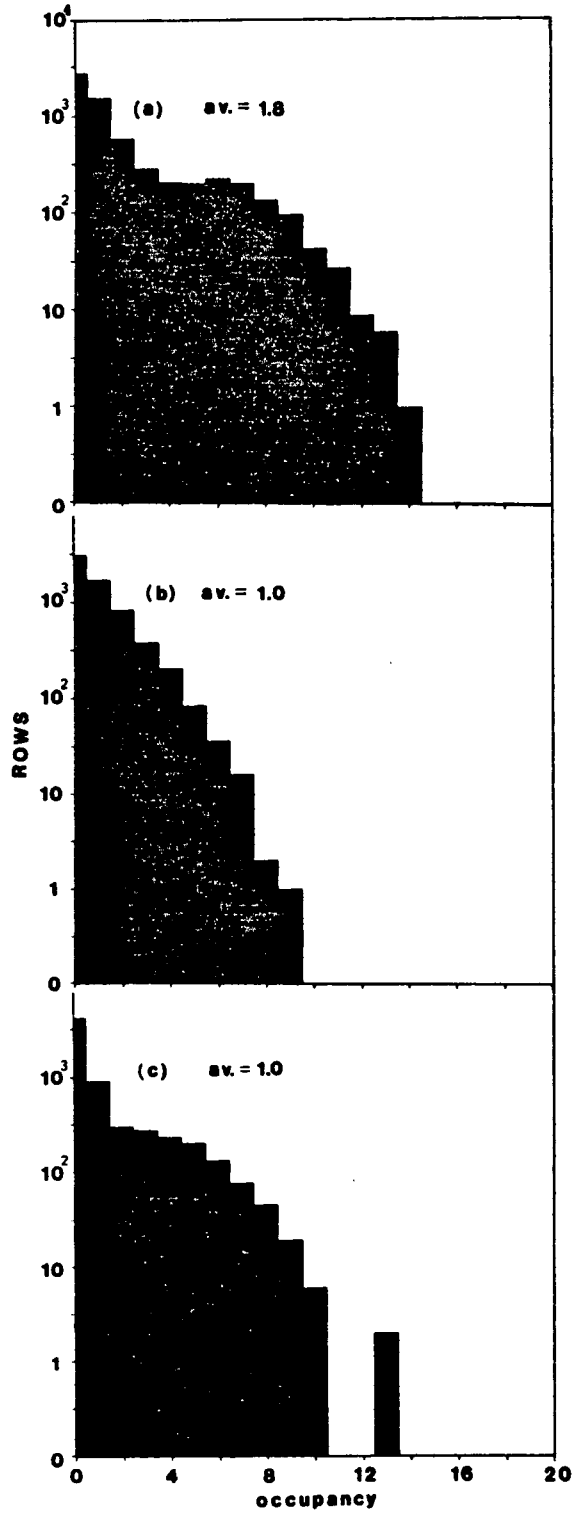


Figure 4.19 Total combined stack occupancy: (a) external border stack; (b) internal border stack; (c) transmission stack

difference may be a statistical anomaly. With a 20 entry safety margin, the previous calculation will require 12.4 X 6.1 + 20 = 95.64 or about 100 locations for the internal and external circular lists. Since no transmission stack is used, the total memory requirement is only 200 locations. Therefore, this system produces an almost 4:1 memory advantage over the multiple stack system. However, regardless of the implementation chosen, it is clear that the list-1-list-2 data structure will dominate the memory requirements of the system.

The simulations of the segmentation system have achieved their primary goal of providing empirical insight into the operation and hardware requirements of the system. The observation that the segmentation system can accomodate further parallelism was unexpected, but welcome. The net result may be a more reliable system in terms of reduced loss of segmentable borders which at the same time is less demanding of hardware technology. Since all the test images were derived from noise, it is not guaranteed that the system will operate in a similar manner on application to other images. However, the complexity of the test images within each resolution class suggests that these images were more demanding of the system than most real applications will be.

4.12 Conclusions

The objective of this chapter was to develop a method for the detection and isolation of closed borders within a dynamic image in real time. The proposed system processed binary-image data in parallel with the input raster scan in order to meet the real-time performance objective. It also took advantage of the dynamic nature of the image in the linkage of detected border points, and in the detection of possible closed borders by monitoring the local change in Euler number.

The overall design recommended for implementation uses a modified Zahn curvature point detector coupled to a multilist data structure to link the curvature points into a closed border. Closure of a border is detected (but not guaranteed) by an Euler number closure detection system which files the pointers to the expected borders on two stacks, one each for internal and external borders. Closure verification occurs at the end of the row scan. Verification of border closure employs the global method of following pointers within the list-1-list-2 curvature point storage in search of cyclical linked lists. On discovery of a closed border, the curvature points, including their coordinate position and edgeout direction, are transmitted to a subsequent processor. Though the nature of the subsequent processor is not of concern to the segmentation system, a reconstruction procedure was investigated to show how the segmented objects detected could be reconstructed from the border information alone. In addition a preliminary procedure was outlined for the separation of touching characters.

Statistics gathered from a software simulation of the

complete system provided the necessary data to facilitate the prediction of memory requirements and estimate the time required to complete each processing stage. The final configuration of the system will be governed largely by the incoming video data rate and the image width in pixels. However, the simulations underscored the fact that much flexibility is possible in the system architecture. It is envisioned that the final implementation will include a rich blend of parallel and serial processes acting in concert.

V. DISCUSSION AND CONCLUSIONS

This thesis has presented the design details of the edge filtering, binarization, and segmentation components of a proposed digital preprocessor for optical character recognition.

After the camera device has acquired the image data, it is to be filtered with the two-dimensional $\nabla^2 g$ optimal edge detection filter. This edge filter was shown to be optimal in the sense of maximizing output signal energy about the character edges. This resulted in a large degree of noise rejection within two filter standard deviations of the edges. The smoothing aspect of the filter also removed any roughness from the character boundaries. Both of these features eliminated the need for an additional noise cleaning stage after binarization.

The resolution response of the $\nabla^2 g$ filter was determined for two edge models. The square wave model is most suitable for predicting when the edge spacings within characters like "m" or "s", or horizontally between multiple characters like "un", can be resolved. The staircase model is not as useful, but it does predict that words locally highlighted by a different color (a practice common in print advertizing) are resolvable. However, it also predicts formation of a psuedo-object associated with the highlight background. If segmentable, this psuedo-object may confuse the recognizer.

The necessary filter width for optimal noise rejection, but matched to the character detail, has a σ_f equal to the observed

edge spacing in pixels divided by the appropriate β. For Dr. Beddoes' CCD scanner, the edge spacing was generally observed to be two pixels. Allowing for moderate blurr, the square wave model predicts a minimum β of 1.25. Therefore, the best σ_f is 2/1.25=1.6. It was further found that the sample spacing should be less than $1.25\sigma_f$. For σ_f=1.6, the maximum spacing is two pixels; clearly, there is no danger of undersampling. The filter was also found to be largely insensitive to coefficient quantization down to at least a low of six bits per word excluding sign. However, it was seen to be important that any resultant dc bias be removed. Since the implementation envisioned will incorporate at least an eight bit total word size, underquantization is unlikely to be a problem. Bias removal proceeds by adding or subtracting a unit from the coefficients near the maximum filter slope at a radius of $0.8\sigma_f$ until all coefficients sum to zero.

The implementation of this filter is a subject which merits further study. However, some suggestions can be made at this point. The most obvious implementation is a simple direct-form discrete convolution. Special purpose hardware is certainly the only means to achieve this at video acquisition rates. To supply a growing demand for this type of operation, special purpose CCD convolution integrated circuits have been manufactured able to support up to a 26X26 operator [76]. With the current image data held in a similar CCD line such a system could readily accomodate a $\nabla^2 g$ filter with σ_f as large as 3.2 at eight bit resolution. An interesting alternative approach was developed by Orbach [51] incorporating hybrid digital-analog circuitry. The video delay

line was implemented with digital RAM memory to achieve a faster data transfer rate than was possible with a CCD line. However, the convolution is performed with multiplying digital to analog converters (MDACs) whose output is then summed by an operational amplifier. The particular attraction of this system is that the output signal is carried on a single line whose sign can be determined with a comparator for binarization. This contrasts with the eight line digital output of an eight bit system where all but the sign bit line are subsequently discarded.

Once the image has been filtered, binarization is a simple matter of labling the positive pixels (dark regions) "1", and the negative pixels (light background) "0". The result is then passed to the segmentation stage.

The segmentor's task was to separate characters from the background provided they do not touch the image window sides, and are surrounded by "0" background pixels. Border pixels were detected through boolean operations performed on a 3X3 window convolved over the binary image at the video rate. To achieve the real-time performance requirement, the border pixels found were logged into linked lists immediately on detection. Simultaneously, an Euler number based technique recorded those pixels suspected of pointing to closed borders. Verification of closure involved following the pointers linking the stored border pixels at the end of each row scan. On closure confirmation, all external and internal borders were subsequently transmitted to the reconstruction/recognition circuitry.

The simulation trials performed indicated that, even for the

most complex images that could reasonably be expected, the processing load is light enough to allow new-found points to be processed in parallel with closure verification and border transmission. Any resultant memory reference conflicts were seen as unlikely to result in serious delays.

After transmission, a method was outlined whereby the binarized image of the segmented characters could be reconstructed from the border data. However, it should be noted that this is not the only avenue to recognition that could be taken. Since the border data fully describes the character it contains, it could be analyzed directly without the intervening reconstruction step. Some specific algorithms to perform such shape analysis from chain coded borders were presented by Pavlidis [77, pp. 185-215] and Freeman [78]. Both these authors have also published general surveys detailing the progress in feature extraction and shape recognition from border descriptions [79], [80]. The major attraction of this approach is that the border data, processed to remove redundant points and normalize the pixel positions, could prove a fully scale- and rotation-invarient representation. That it can also be fast and efficient is demonstrated by the 100 character per second recognition rate reported by D'Amato et al. [66] where shape matching against features extracted from a chain code facilitates recognition.

The design of the OCR preprocessing system is by no means complete; considerable opportunities remain for further work. The need was already outlined for a system component to accurately parse touching characters. However, at the other end

of the resolution scale, an accurate, autonomous method is needed for the tracing of print lines. This will guide the correct causal acquisition of words and characters from a current print line, and, subsequently, the correct acquisition of gramatically connected text lines. A method of signaling and/or ignoring non-text material such as illustrations may also prove necessary. Both these methods would require filtering the image at lower resolutions, choosing σ_f so that print lines would be resolved as a unit, blurring the component words. Illustrations would represent non-segmentable objects at this low resolution level. The character tracking and acquisition stages could then be signalled to ignore the high resolution data generated by non-text input. Such an expanded system would seem to favor new imaging devices capable of acquiring more than a single text line per scan. All of these extensions however would still include the same system components outlined in this report: a $\nabla^2 g$ filter to highlight the desired level of detail; and a non-sequential object-border segmentation method to acquire the necessary image information. Since these components have, in this thesis, been shown capable of real-time, autonomous operation, it is likely that such an expanded preprocessor will also operate satisfactorily.

REFERENCES

[1] The British Computer Society, "Character recognition," Gresham press, Unwin Brothers Ltd., London, 1967.

[2] Auerbach Info. Inc., "Auerbach on optical character recognition," Auerbach Publishers, Princeton N.J., 1971.

[3] L.D. Harmon, "Automatic recognition of print and script," Proceedings of IEEE, Vol. 60, No. 10, pp. 1165-1176, Oct. 1972.

[4] J.R. Ullmann, "Pattern recognition techniques," Crane, Russak and Co. Inc., New York, 1973.

[5] J.R. Ullmann, "Picture analysis in character recognition," Digital Picture Analysis (ed. A. Rosenfeld), Springer-Verlag, pp. 295-343, 1976.

[6] J.R. Ullmann, "A relational view of text image processing," Issues in Digital Image Processing (eds. R.M. Haralick and J.C. Simon), Sijthoff and Noordhoff, pp. 139-169, 1980.

[7] J. Schürmann, "Reading Machines," 6th International Conf. on Pattern Recognition, Munich, Germany, pp. 1031-1044, Oct. 19-22, 1982.

[8] K.Y. Wong, R.G. Cassey, F.M. Wahl, "Document analysis system," IBM J. Research and Development, Vol. 26, No. 6, pp. 647-656, Nov. 1982.

[9] M.R. Bartz, "The IBM 1975 optical page reader part II: video thresholding system," IBM J. Research and Development, Vol. 12, No. 5, pp. 354-363, Sept. 1968.

[10] N. Otsu, "A threshold selection method from gray-level histograms," IEEE Trans. Syst. Man. and Cybern., Vol. SMC-9, No. 1, pp. 62-66, Jan. 1979.

[11] J.T. Tou, D. Lu, J. McCarthy, "Automatic detection of document falsification," Proc. PRIP83 - IEEE Computer Soc. Conf. Pattern Recognition and Image Processing, Las Vegas, Nevada, pp. 158-164, June 14-17, 1982.

[12] A. Rosenfeld and C.M. Park, "Noise cleaning in digital pictures," EASCON'69 record - IEEE Electronics and Aerospace Systems Conv., Washington, D.C., pp. 264-273, Oct. 27-29, 1969.

[13] E.C. Greanias, P.F. Meagher, R.J. Norman, P. Essinger, "The recognition of handwritten numerals by contour analysis," IBM J. Research and Development, Vol. 7, pp. 14-21, Jan. 1963.

[14] D.O. Clayden, M.B. Clowes, J.R. Parks, "Letter recognition and the Segmentation of running text," Information and Control, Vol. 9, pp. 246-264, 1966.

[15] International Business Machines, "IBM System/360 Component Description. IBM 1275 Optical Reader Sorter," File No. S 360-03, Form A19-0034-0, Jan. 1969.

[16] D. Marr and E. Hildreth, "Theory of edge detection," Proc. R. Soc. Lord. B, Vol. 207, pp. 187-217, 1980.

[17] F.M. Dickey and K.S. Shanmugam, "Optimum edge detection filter," Applied Optics, Vol. 16, No. 1, pp. 145-148, Jan. 1977.

[18] L.G. Roberts, "Machine perception of three-dimensional solids," Optical and Electro-Optical Information Processing (eds. J.T. Tippett et al.), MIT Press, Cambridge, MA., pp. 159-197, 1965.

[19] A. Rosenfeld and M. Thurston, "Edge and curve detection for visual scene analysis," IEEE Trans. Comput., Vol. C-20, No. 5, pp. 562-569, May 1971.

[20] I.D.G. Macleod, "On finding structure in pictures," Picture Language Machines (ed. S. Kaneff), pp. 231-256, New York: Academic, 1970.

[21] M.H. Hueckel, "An operator which locates edges in digitized pictures," J. Ass. Comput. Mach., Vol. 18, pp. 113-125, Jan. 1971.

[22] M.H. Hueckel, "A local visual operator which recognizes edges and lines," J. Ass. Comput. Mach., Vol. 20, pp. 634-647, Oct. 1973.

[23] J.A.G. Halé, "Detection of elementry features in a picture by non linear local numerical processing," Proc. 3rd Int. Joint Conf. Pattern Recognition, Coronado, California, pp. 764-768, Nov. 8-11, 1976.

[24] F.W. Campbell, "The physics of visual perception," Phil. Trans. R. Soc. Lond. B, Vol. 290, pp. 5-9, 1980.

[25] H.R. Wilson and S.C. Giese, "Threshold visibility of frequency gradient patterns," Vision Res. Vol. 17, pp. 1177-1190, 1977.

[26] D. Marr and T. Poggio, "A computational theory of human sterio vision," Proc. R. Soc. Lond. B, Vol. 204, pp. 301-328, 1979.

[27] R. Leipnik, "The extended entropy uncertainty principle," Information and Control, Vol. 3, pp. 18-25, 1960.

[28] B.F. Logan Jr., "Information in the zero crossings of bandpass signals," Bell Sys. Tech. J., Vol. 56, No. 4, pp. 487-510, April 1977.

[29] D. Marr, S. Ullman and T. Poggio, "Bandpass channels, zero crossings, and early visual information processing," J. Opt. Soc. Am., Vol. 69, No. 6, pp. 914-916, June 1979.

[30] C. Flammer, "Spheroidal wave functions," Stanford University Press, Stanford CA., 1957.

[31] J.C. Dainty and R. Shaw, "Image science: principles, analysis and evaluation of photographic - type imaging processes," London: Academic, 1974.

[32] K.S. Shanmugam, F.M. Dickey and J.A. Green, "An optimal frequency domain filter for edge detection in digital pictures," IEEE Trans. PAMI, Vol. PAMI-1, No. 1, pp. 37-49, Jan. 1979.

[33] W.H.H.J. Lunscher, "The asymptotic optimal frequency domain filter for edge detection," IEEE Trans. PAMI, paper no. 83-2-21, in press.

[34] R. Paley and N. Wiener, "Fourier transforms in the complex domain," Amer. Math. Soc. Colloq. Publ. XIX, New York, 1934.

[35] W.E.L. Grimson, "A computer implementation of a theory of human stereo vision," Phil. Trans. R. Soc. Lond. B, Vol. 292, pp. 217-253, 12 May 1981.

[36] J. Clark, private communication, Univ. of British Columbia, Dept. of Elec. Eng., Vancouver, B.C., Canada.

[37] G.F.D. Duff and D. Naylor, "Differential equations of applied mathematics," J. Wiley and Sons Inc., 1966.

[38] P.H. Smith, B.J. McCartin and D.L. Davies, "Software concepts useful for industrial image processing," Proc. PRIP79-IEEE Computer Soc. Conf. Pattern Recognition and Image Processing, Chicago, Ill., pp. 653-659, 1979.

[39] R.E. Ziemer and W.H. Trantor, "Principles of communications; systems, modulation, and noise," Houghton Mifflin Co., 1976.

[40] A. Rosenfeld and A.C. Kak, "Digital Picture Processing," second editon, Vol. 2, New York: Academic, 1982.

[41] S.W. Zucker, "Local structure, consistency, and continuous relaxation," Issues in Digital Image Processing (eds. R.M. Haralick and J.C. Simon), Sijthoff and Noordhoff Int., The Netherlands, pp. 195-220, 1980.

[42] J.M. Prager, "Extracting and labelling boundary segments in natural scenes," IEEE Trans. PAMI, Vol. PAMI-2, No. 1, pp. 16-27, Jan. 1980.

[43] A.R. Hanson and E.M. Riseman, "Processing cones; a computational structure for image analysis," Structured Computer Vision: machine perception through hierarchical computation structures (eds. S. Tanimoto and A. Klinger), New York: Academic, pp. 101-131, 1980.

[44] M.D. Kelly, "Edge detection in pictures by computer using planning," Machine Intelligence, Vol. 6, pp. 397-409, 1971.

[45] W.B. Green, "Digital image processing: a systems approach" New York: Van Nostrand Reinhold Co., 1983.

[46] D.S.K. Chan and L.R. Rabiner, "Analysis of quantization errors in the direct form for finite impulse response digital filters," IEEE Trans. Audio and Electroacoustics, Vol. AU-21, No. 4, pp. 354-366, August 1973.

[47] W. Feller, "An introduction to probability theory and its applications," second edition, Vol. 2, J. Wiley and Sons Inc., 1971.

[48] L. Kitchen and A. Rosenfeld, "Edge evaluation using local edge coherence," IEEE Trans. Sys. Man, and Cybern. Vol. SMC-11, No. 9, pp. 597-605, Sept. 1981.

[49] H. Freeman, "On the encoding of arbitrary geometric configurations," IRE Trans. Electron. Comput., Vol. EC-10, pp. 260-268, June 1961.

[50] T. Mimaroglu, "A high speed two-dimensional hardware convolver for image processing," Proc. PRIP82-IEEE Computer Soc. Conf. Pattern Recognition and Image Processing, Las Vegas, Nevada, pp. 386-389, June 14-17, 1982.

[51] Z. Orbach, "Real-time video image enhancement using hardware convolver," Proc. PRIP82-IEEE Computer Soc. Conf. Pattern Recognition and Image Processing, Las Vegas, Nevada, pp. 390-392, June 14-17, 1982.

[52] J.R. Fram and E.S. Deutsch, "On the quantitative evaluation of edge detection schemes and their comparison with human performance," IEEE Trans. Comput., Vol C-24, No. 6, pp. 616-627, June 1975.

[53] E.S. Deutsch and J.R. Fram, "A quantitative study of the orientation bias of some edge detector schemes," IEEE Trans. Comput., Vol. C-27, No. 3, pp. 205-213, Mar. 1978.

[54] I.E. Abdou and W.K. Pratt, "Qantitative design and evalutation of enhancement/thresholding edge detectors," Proceedings of IEEE, Vol. 67, No. 5, pp. 753-763, May 1979.

[55] J.M.S. Prewitt, "Object enhancement and extraction," Picture Processing and Psychopictories (eds. B.S. Lipkin and A. Rosenfeld), New York: Academic, 1970.

[56] R.O. Duda and P.E. Hart, "Pattern classification and scene analysis," New York: Wiley, 1973.

[57] R. Kirsch, "Computer determination of the constituent structure of biological images," Comput. Biomed. Res., Vol. 4, No. 3, pp. 315-328, 1971.

[58] T. Peli and D. Malah, "A study of edge detection algorithms," Computer Graphics and Image Processing, Vol. 20, pp. 1-21, Sept. 1982.

[59] G.B. Shaw, "Local and regional edge detectors: some comparisons," Computer Graphics and Image Processing, Vol. 9, pp. 135-149, 1979.

[60] A.Rosenfeld, "Connectivity in Digital Pictures," J. of A.C.M., Vol. 17, No. 1, pp. 146-160, Jan. 1970.

[61] S.J. Mason and J.K. Clemens, "Character Recognition in an experimental reading machine for the blind," Recognizing Patterns; studies in living and automatic systems (eds. P.A. Kolers and M. Eden), pp. 156-167, M.I.T. press, Cambridge, MA., 1968.

[62] R.L. Hoffman and J.W. McCullough, "Segmentation methods for recognition of machine-printed characters," IBM J. of Research and Development, Vol. 15, No. 2, pp. 153-165, Mar. 1971.

[63] J.K. Clemens, "Optical character recognition for reading machine applications," Ph.D. Thesis, M.I.T., September, 1965.

[64] M.P. Beddoes and W. Lunscher, "A microcomputer-based letter recognizer: discussion of some problems and solutions and results obtained with an 8085," Int. Congress on Applied Systems Research and Cybernetics, Acapulco, Mexico, 1980.

[65] I. Sobel, "Neighborhood coding of binary images for fast contour following and general binary array processing, "Computer Graphics and Image Processing, Vol. 8, No. 1, pp. 127-135, Aug. 1978.

[66] D. D'Amato, L. Pintsou, H. Koay, D. Stone, J. Tan, K. Tuttle, D. Buck, "High speed pattern recognition system for alphanumeric handprinted characters," Proc. PRIP82 - IEEE Computer Soc. Conf. Pattern Recognition and Image Processing, Las Vegas, Nevada, pp. 165-170, June 14-17, 1982.

[67] T. Pavlidis, "A minimum storage boundary tracing algorithm and its application to automatic inspection, "IEEE Trans. Syst. Man, and Cybern., Vol. SMC-8, No. 1, pp. 66-69, Jan. 1978.

[68] I. Chakravarty, "A single-pass, chain generating algorithm for region boundaries," Computer Graphics and Image Processing, Vol. 15, pp. 182-193, 1981.

[69] B.G. Batchelor and B.K. Marlow, "Fast generation of chain code," IEEE Proc., Pt. E, Vol. 127, No. 4, pp. 143-147, July 1980.

[70] C.T. Zahn, "A formal description for two-dimension patterns," Proc. Int. Joint Conf. on Artificial Intelligence, pp. 621-628, May 1969.

[71] M. Minsky and S. Papert, "Perceptrons, an introduction to computational geometry," M.I.T. press, Cambridge, MA., 1969.

[72] A. Rosenfeld and J.L. Pfaltz, "Sequential operations in digital picture processing," J. of A.C.M., Vol. 13, No. 4, pp. 471-494, Oct. 1966.

[73] S.B. Gray, "Local properties of binary images in two-dimensions," IEEE Trans. Comput., Vol. C-20, No. 5, pp. 551-561, May 1971.

[74] A. Rosenfeld, "Picture languages, formal models for picture recognition," New York: Academic, 1979.

[75] R.G. Casey and G. Nagy, "Recursive segmentation and classification of composite character patterns," 6th Int. Conf. on Pattern Recognition, Munich, Germany, 1023-1026, Oct. 19-22, 1982.

[76] S. Ullmann, "Interfacing the one-dimensional scanning of an image with the application of two-dimensional operators," Computer Graphics and Image Processing, Vol 16, pp. 150-157, 1981.

[77] T. Pavlidis, "Structural Pattern Recognition," Springer-Verlag, 1977.

[78] H. Freeman, "Use incremental curvature for describing and analyzing two-dimensional shape," Proc. PRIP79 - IEEE Computer Soc. Conf. Pattern Recognition and Image Processing, Chicago, Ill., pp. 437-444, Aug. 6-8, 1979.

[79] T. Pavlidis, "A review of algorithms for shape analysis," Computer Graphics and Image Processing, Vol. 7, pp. 243-258, 1978.

[80] H. Freeman, "Computer processing of line-drawing images," A.C.M. Computing Surveys, Vol. 6, No. 1, Mar. 1974.