"AUTOMATIC MAIN ROAD EXTRACTION FROM HIGH RESOLUTION SATELLTE IMAGERY"

A project report submitted in partial fulfillments of the requirements

for the degree of

Bachelor Of Technology in Electrical Engineering

BY

Mr. Pratik Panchabhaiyye

Roll No. : 10502010

Mr. Ravi Shekhar Chaudhary

Roll No. : 10502051

Guided by

Prof. K.R. Subhashini

Deptt. Of Electrical Engineering

NIT Rourkela

National Institute Of Technology Rourkela,

Rourkela, Orissa - 769008

2008-2009

Certificate

This is to certify that **Mr. Ravi Shekhar Chaudhary** of 8th semester Electrical Engineering has completed the B.Tech Project on "**Automatic Main Road Extraction From High Resolution Satellite Imagery**" in a satisfactory manner.

This project is submitted in partial fulfillment towards the Bachelor's degree in Electrical Engineering as prescribed by the institute during the year 2008-2009.

PROF. K.R. Subhashini
Project Guide
N.I.T ROURKELA

PROF.B.D.Subudhi
H.O.D
Deptt Of Electrical Engg

Certificate

This is to certify that **Mr. Pratik Panchabhaiyye** of 8th semester Electrical Engineering has completed the B.Tech Project on "**Automatic Main Road Extraction From High Resolution Satellite Imagery**" in a satisfactory manner.

This project is submitted in partial fulfillment towards the Bachelor's degree in Electrical Engineering as prescribed by the institute during the year 2008-2009.

PROF. K.R. Subhashini
Project Guide
N.I.T ROURKELA

PROF.B.D.Subudhi
H.O.D
Deptt Of Electrical Engg

ABSTRACT

Road information is essential for automatic GIS (geographical information system) data acquisition, transportation and urban planning. Automatic road (network) detection from high resolution satellite imagery will hold great potential for significant reduction of database development/updating cost and turnaround time. From so called low level feature detection to high level context supported grouping, so many algorithms and methodologies have been presented for this purpose. There is not any practical system that can fully automatically extract road network from space imagery for the purpose of automatic mapping. This paper presents the methodology of automatic main road detection from high resolution satellite IKONOS imagery. The strategies include multiresolution or image pyramid method, Gaussian blurring and the line finder using 1-dimemsional template correlation filter, line segment grouping and multi-layer result integration. Multi-layer or multi-resolution method for road extraction is a very effective strategy to save processing time and improve robustness. To realize the strategy, the original IKONOS image is compressed into different corresponding image resolution so that an image pyramid is generated; after that the line finder of 1-dimemsional template correlation filter after Gaussian blurring filtering is applied to detect the road centerline. Extracted centerline segments belong to or do not belong to roads. There are two ways to identify the attributes of the segments, the one is using segment grouping to form longer line segments and assign a possibility to the segment depending on the length and other geometric and photometric attribute of the segment, for example the longer segment means bigger possibility of being road. Perceptual-grouping based method is used for road segment linking by a possibility model that takes multi-information into account; here the clues existing in the gaps

are considered. Another way to identify the segments is feature detection back-to-higher resolution layer from the image pyramid.

CONTENTS

S.NO.	TOPIC	PAGE NO.
1.	ABSTRACT	1
2.	INTRODUCTION	2
3.	LITERATURE REVIEW	4
4.	STRATEGY	9
5.	IMPLEMENTATION - CODING	15
6.	CONCLUSION & FUTURE WORK	27
7.	REFERENCES	28

1. Introduction

Modern digital technology has made it possible to manipulate multi-dimensional signals with systems that range from simple digital circuits to advanced parallel computers. The goal of this manipulation can be divided into three categories:

- Image Processing *image in* → *image out*
- Image Analysis *image in* → *measurements out*
- Image Understanding image in ⟶ high-level description out

Here we will focus on the fundamental concepts of *image processing*.
Image understanding requires an approach that differs fundamentally from the theme of our discussion. Further, we will restrict ourselves to two–dimensional (2D) image processing although most of the concepts and techniques that are to be described can be extended easily to three or more dimensions.
We begin with certain basic definitions. An ***image*** defined in the "real world" is considered to be a function of two real variables, for example, $a(x,y)$ with a as the amplitude (e.g. brightness) of the image at the *real* coordinate position (x,y). An image may be considered to contain sub-images sometimes referred to as
regions–of–interest, *ROIs*, or simply *regions*. This concept reflects the fact that images frequently contain collections of objects each of which can be the basis for a region. In a sophisticated image processing system it should be possible to apply specific image processing operations to selected regions. Thus one part of an image (region) might be processed to suppress motion blur while another part might be processed to improve color rendition. The amplitudes of a given image will almost always be either real numbers or integer numbers. The latter is usually a result of a quantization process that converts a continuous range (say, between 0 and 100%) to a discrete number of levels. In certain image-forming processes, however, the signal may involve photon counting which implies that the amplitude would be inherently quantized. In other image forming procedures, such as magnetic resonance imaging, the direct physical measurement yields a complex number in the form of a real magnitude and a real phase.

2. Digital Image Definitions

A digital image $a[m,n]$ described in a 2D discrete space is derived from an analog image $a(x,y)$ in a 2D continuous space through a *sampling* process that is frequently referred to as digitization. The mathematics of that sampling process will be described in Section 5. For now we will look at some basic definitions
associated with the digital image. The effect of digitization is shown in Figure 1.
The 2D continuous image $a(x,y)$ is divided into *N rows* and *M columns*. The intersection of a row and a column is termed a *pixel*. The value assigned to the
integer coordinates $[m,n]$ with $\{m=0,1,2,\ldots,M-1\}$ and $\{n=0,1,2,\ldots,N-1\}$ is $a[m,n]$.
In fact, in most cases $a(x,y)$—which we might consider to be the physical signal that impinges on the face of a 2D sensor—is actually a function of many variables including depth (z), color (l), and time (t). Unless otherwise stated, we will consider the case of 2D, monochromatic, static images in this chapter.

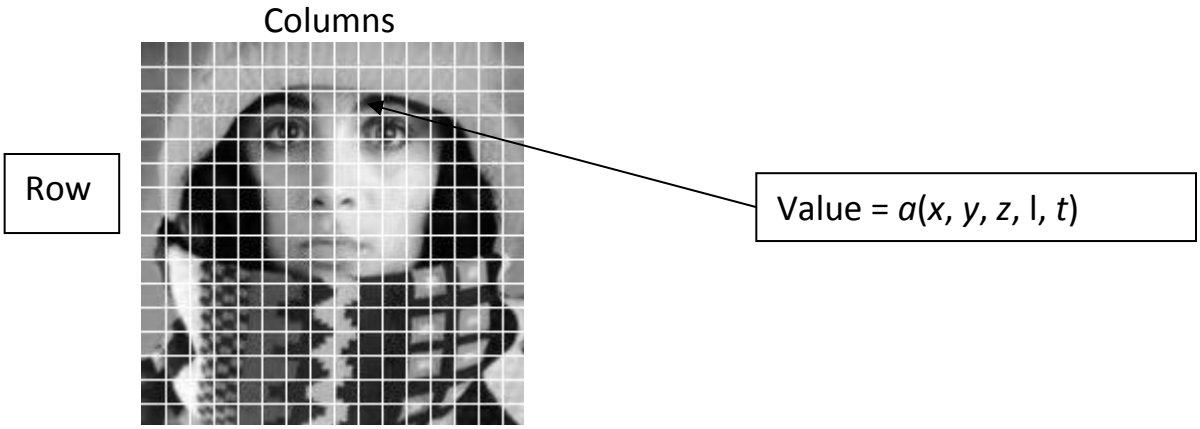


Figure 2.1: Digitization of a continuous image. The pixel at coordinates $[m=10, n=3]$ has the integer brightness value 110.

The image shown in Figure 2.1 has been divided into $N = 16$ rows and $M = 16$ columns. The value assigned to every pixel is the average brightness in the pixel rounded to the nearest integer value. The process of representing the amplitude of the 2D signal at a given coordinate as an integer value with L different gray levels is usually referred to as amplitude quantization or simply *quantization*.

2.1 COMMON VALUES

There are standard values for the various parameters encountered in digital image processing. These values can be caused by video standards, by algorithmic requirements, or by the desire to keep digital circuitry simple. Table 1 gives some commonly encountered problems.

Parameters	Symbol	Typical Values					
Rows	N	256	512	525	625	1024	1035
Columns	M	256	512	768	1024	1320	
Grey Levels	L	2	64	256	1024	4096	16384

Table 1: Common values of digital image parameters

Quite frequently we see cases of $M=N=2K$ where $\{K = 8,9,10\}$. This can be motivated by digital circuitry or by the use of certain algorithms such as the (fast)
Fourier transforms. The number of distinct gray levels is usually a power of 2, that is, $L=2B$ where B is the number of bits in the binary representation of the brightness levels. When $B>1$ we speak of a *gray-level image*; when $B=1$ we speak of a *binary image*. In a binary image there are just two gray levels which can be referred to, for example, as “black” and “white” or “0” and “1”.

2.2 CHARACTERISTICS OF IMAGE OPERATIONS

There is a variety of ways to classify and characterize image operations. The reason for doing so is to understand what type of results we might expect to achieve with a given type of operation or what might be the computational burden associated with a given operation.

2.2.1 Types of operations

The types of operations that can be applied to digital images to transform an input image $a[m,n]$ into an output image $b[m,n]$ (or another representation) can be classified into three categories as shown in Table 2.

Operation	Characterization	Generic Complexity/Pixel
• *Point*	– the output value at a specific coordinate is dependent only on the input value at that same coordinate.	*constant*
• *Local*	– the output value at a specific coordinate is dependent on the input values in the *neighborhood* of that same coordinate.	P^2
• *Global*	– the output value at a specific coordinate is dependent on all the values in the input image.	N^2

Table 2: Types of image operations. Image size = $N \times N$; neighborhood size = $P \times P$. Note that the complexity is specified in operations *per pixel*.

2.2.2 Types of neighborhoods

Neighborhood operations play a key role in modern digital image processing. It is therefore important to understand how images can be sampled and how that relates to the various neighborhoods that can be used to process an image.

• Rectangular sampling – In most cases, images are sampled by laying a rectangular grid over an image as illustrated in Figure 1. This results in the type of sampling shown in Figure 3ab.

• Hexagonal sampling – An alternative sampling scheme is shown in Figure 3c and is termed hexagonal sampling.

Both sampling schemes have been studied extensively [1] and both represent a possible periodic tiling of the continuous image space. We will restrict our attention, however, to only rectangular sampling as it remains, due to hardware and software considerations, the method of choice.

Local operations produce an output pixel value *b*[*m*=*mo*,*n*=*no*] based upon the pixel values in the *neighborhood* of *a*[*m*=*mo*,*n*=*no*]. Some of the most common neighborhoods are the 4-connected neighborhood and the 8-connected neighborhood in the case of rectangular sampling and the 6-connected neighborhood in the case of hexagonal sampling illustrated in Figure 3.

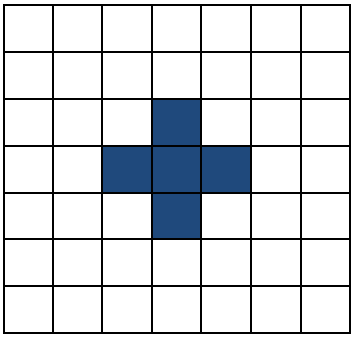


Figure 2.2a
Rectangular sampling
4-connected

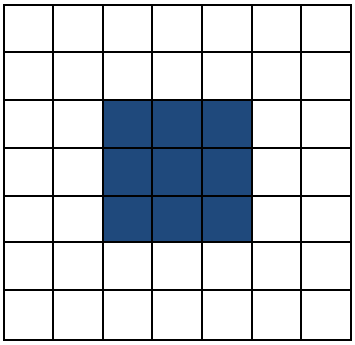


Figure 2.2b
Rectangular sampling
8-connected

3. Tools

Certain tools are central to the processing of digital images. These include
Mathematical tools such as ***convolution***, ***Fourier analysis***, and ***statistical descriptions,*** and manipulative tools such as *chain codes* and *run codes*. We will present these tools without any specific motivation. The motivation will follow in later sections.

3.1 CONVOLUTION

There are several possible notations to indicate the convolution of two (multidimensional) signals to produce an output signal. The most common are:

$$c = a \otimes b = a * b \tag{1}$$

We shall use the first form, $c = a * b$, with the following formal definitions.

In 2D continuous space:

$$c(x, y) = a(x, y) * b(x, y) = \iint a(\chi, \xi) b(x - \chi, y - \xi) d\chi \, d\xi \tag{2}$$

In 2D discrete space:

$$c[m,n] = a[m,n] \otimes b[m, n] = \sum\sum a[j,k] b[m- j,n - k] \tag{3}$$

3.2 PROPERTIES OF CONVOLUTION

There are a number of important mathematical properties associated with convolution.

• Convolution is *commutative*.

$c = a\ b \otimes = b \otimes a$ (4)

• Convolution is *associative*.

$c = a \otimes (b \otimes d) = (a \otimes b) \otimes d = a \otimes b \otimes d$ (5)

• Convolution is *distributive*.

$c = a \otimes (b + d) = (a \otimes b) + (a \otimes d)$ (6)

where a, b, c, and d are all images, either continuous or discrete.

3.3 FOURIER TRANSFORMS

The Fourier transform produces another representation of a signal, specifically a representation as a weighted sum of complex exponentials. Because of Euler's formula:

$e^{jq} = \cos(q) + j\sin(q)$ (7)

where $j^2 = -1$, we can say that the Fourier transform produces a representation of a (2D) signal as a weighted sum of sines and cosines. The defining formulas for the forward Fourier and the inverse Fourier transforms are as follows. Given an image a and its Fourier transform A, then the forward transform goes from the spatial domain (either continuous or discrete) to the frequency domain which is always continuous.

Forward – $A = F\{a\}$ (8)

The inverse Fourier transform goes from the frequency domain back to the spatial domain.

Inverse – $a = F^{-1}\{A\}$ (9)

The Fourier transform is a unique and invertible operation so that:

$a = F^{-1}\{F\{a\}\}$ *and* $A = F\{F^{-1}\{A\}\}$ (10)

The specific formulas for transforming back and forth between the spatial domain and the frequency domain are given below.

In 2D continuous space:

Forward – $A(u,v) = \iint a(x,y)e^{-j(ux+vy)}dxdy$ (11)

Inverse – $a(x,y) = 1/4\pi^2 \iint_{-\infty}^{+\infty} A(u,v)e^{-j(ux+vy)}\,dudv$ (12)

In 2D discrete space:

Forward – $A(\Omega, \Psi) = \sum_{m=-\infty}^{+\infty} \sum_{n=-\infty}^{+\infty} a[m, n] e^{-j(\Omega m + \Psi n)}$ (13)

Inverse – $a[m, n] = 1/4\pi^2 \int_{-\pi}^{+\pi} \int_{-\pi}^{+\pi} A(\Omega, \Psi) e^{+j(\Omega m + \Psi n)} d\Omega d\Psi$ (14)

3.4 PROPERTIES OF FOURIER TRANSFORMS

There are a variety of properties associated with the Fourier transform and the inverse Fourier transform. The following are some of the most relevant for digital image processing.

• The Fourier transform is, in general, a complex function of the real frequency variables. As such the transform can be written in terms of its magnitude and phase.

$A(u,v) = |A(u,v)|\, e^{j\varphi(u,v)}$ $A(W,Y) = |A(\Omega,\Psi)| e^{j\varphi(\Omega,\Psi)}$ (15)

• A 2D signal can also be complex and thus written in terms of its magnitude and phase.

$a(x,y) = |a(x,y)| e^{j\vartheta(x,y)}$ $a[m,n] = |a[m,n]| e^{j\vartheta[m,n]}$ (16)

• If a 2D signal is real, then the Fourier transform has certain symmetries.

$A(u,y) = A^*(-u,-v)$ $A(\Omega,\Psi) = A^*(-\Omega,-\Psi)$ (17)

The symbol (*) indicates complex conjugation. For real signals eq. (17) leads directly to:

$|A(u,v)| = |A(-u,-v)|$ (18)

• If a 2D signal is real and even, then the Fourier transform is real and even.

$A(u,v) = A(-u,-v)$ $A(\Omega,\Psi) = A(-\Omega,-\Psi)$ (19)

• The Fourier and the inverse Fourier transforms are linear operations.

$F\{w_1 a + w_2 b\} = F\{w_1 a\} + F\{w_2 b\} = w_1 A + w_2 B$

$F^{-1}\{w_1A + w_2B\} = F^{-1}\{w_1A\} + F^{-1}\{w_2B\} = w_1a + w_2b$ (20)

where *a* and *b* are 2D signals (images) and *w1* and *w2* are arbitrary, complex constants.

• The Fourier transform in discrete space, $A(\Omega, \Psi)$, is periodic in both W and Y. Both periods are 2π.

$A(\Omega + 2\pi j, \Psi + 2\pi k) = A(\Omega, \Psi)$ *j*, *k* integers (21)

• The energy, *E*, in a signal can be measured either in the spatial domain or the frequency domain. For a signal with finite energy:

Parseval's theorem (2D continuous space):

$$E = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}|a(x,y)|^2\,dxdy = 1/4\pi^2\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty}|A(u,v)|^2\,dudv \qquad (22)$$

This "signal energy" is not to be confused with the physical energy in the phenomenon that produced the signal. If, for example, the value *a*[*m*,*n*] represents a photon count, then the *physical* energy is proportional to the amplitude, *a*, and not the square of the amplitude. This is generally the case in video imaging.

3.4.1 Importance of phase and magnitude

Equation (15) indicates that the Fourier transform of an image can be complex.
This is illustrated below in Figures 4a-c. Figure 4a shows the original image *a*[*m*,*n*], Figure 4b the magnitude in a scaled form as log(|*A*(Ω,Ψ)|), and Figure 4c the phase j(Ω,Ψ).

Figure 4a
Original

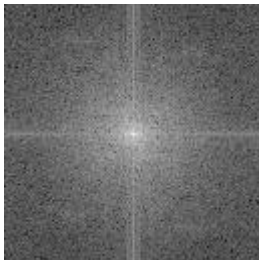


Figure 4b
log(|A(,Ω,Ψ)|)

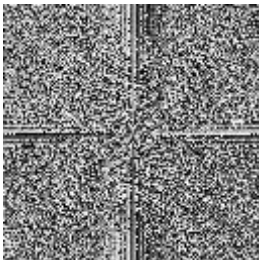


Figure 4c
Φ(Ω,Ψ)

Both the magnitude and the phase functions are necessary for the complete reconstruction of an image from its Fourier transform. Figure 5a shows what happens when Figure 4a is restored solely on the basis of the magnitude information and Figure 5b shows what happens when Figure 4a is restored solely on the basis of the phase information.

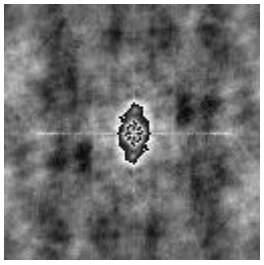


Figure 5a

$\Phi(\Omega,\Psi) = 0$

Figure 5b

$|A(\Omega,\Psi)| = constant$

Neither the magnitude information nor the phase information is sufficient to restore the image. The magnitude–only image (Figure 5a) is unrecognizable and has severe dynamic range problems. The phase-only image (Figure 5b) is barely recognizable, that is, severely degraded in quality.

3.4.2 Circularly symmetric signals

An arbitrary 2D signal $a(x,y)$ can always be written in a polar coordinate system as $a(r,\theta)$. When the 2D signal exhibits a circular symmetry this means that:

$$a(x, y) = a(r,\theta) = a(r) \tag{23}$$

where $r^2 = x^2 + y^2$ and $\tan\theta = y/x$. As a number of physical systems such as lenses exhibit circular symmetry, it is useful to be able to compute an appropriate Fourier representation.
The Fourier transform $A(u, v)$ can be written in polar coordinates $A(\omega r,\xi)$ and then, for a circularly symmetric signal, rewritten as a *Hankel transform*:

$$A(u,v) = \{a(x,y)\} = 2\pi \int_0^\infty a(r)\, J_o(\omega_r r) r dr = A(\omega_r) \tag{24}$$

where $\omega_r^2 = u^2 + v^2$ and $\tan\xi = v\, u$ and $J_o(\bullet)$ is a Bessel function of the first kind of order zero.

The inverse *Hankel transform* is given by:

$$a(r) = 1/2\pi \int_0^{\infty} A\,(\omega_r\,) J_o(\omega_r r)\omega_r d\omega_r \qquad (25)$$

The Fourier transform of a circularly symmetric 2D signal is a function of only the radial frequency, ω_r. The dependence on the angular frequency, ξ, has vanished.
Further, if $a(x,y) = a(r)$ is real, then it is automatically even due to the circular symmetry. According to equation (19), $A(\omega_r)$ will then be real and even.

3.4.3 Examples of 2D signals and transforms

Table 4 shows some basic and useful signals and their 2D Fourier transforms. In using the table entries in the remainder of this chapter we will refer to a spatial domain term as the *point spread function* (*PSF*) or the *2D impulse response* and its Fourier transforms as the *optical transfer function* (*OTF*) or simply *transfer function*. Two standard signals used in this table are $u(\bullet)$, the unit step function, and $J1(\bullet)$, the Bessel function of the first kind. Circularly symmetric signals are treated as functions of r as in eq. (23).

4. GEOMETRICAL IMAGE RESAMPLING

As noted in the preceding sections of this chapter, the reverse address computation process usually results in an address result lying between known pixel values of an input image. Thus it is necessary to estimate the unknown pixel amplitude from its known neighbors. This process is related to the image reconstruction task, as described in Chapter 4, in which a space-continuous display is generated from an array of image samples. However, the geometrical resampling process is usually not spatially regular. Furthermore, the process is discrete to discrete; only one output pixel is produced for each input address.
In this section, consideration is given to the general geometrical resampling process in which output pixels are estimated by interpolation of input pixels. The special, but common case, of image magnification by an integer zooming factor is also discussed. In this case, it is possible to perform pixel estimation by convolution.

4.1. Interpolation Methods

The simplest form of resampling interpolation is to choose the amplitude of an output image pixel to be the amplitude of the input pixel nearest to the reverse address. This process, called *nearest-neighbor interpolation*, can result in a spatial offset error by as much as pixel units. The resampling interpolation error can be significantly reduced by utilizing all four nearest neighbors in the interpolation. A common approach, called *bilinear interpolation*, is to interpolate linearly along each row of an image and then interpolate that result linearly in the columnar direction. The estimated pixel is easily found to be

$$F(p2, q2)=(1-a)[(1-b)F(p, q)+ bF(p, q+1)] + a[(1-b)F(p+1, q)+ bF(p+1, q+1)] \quad (4.1.1)$$

Although the horizontal and vertical interpolation operations are each linear, in general, their sequential application results in a nonlinear surface fit between the four neighboring pixels.

The expression for bilinear interpolation of Eq. 4.5-1 can be generalized for any interpolation function that is zero-valued outside the range of sample spacing. With this generalization, interpolation can be considered as the summing of four weighted interpolation functions as given by

$$F(p2, q2) = F(p, q)R\{-a\}R\{b\} + F(p, q+1)R\{-a\}R\{-(1-b)\}$$
$$+ F(p+1, q)R\{1-a\}R\{b\} + F(p+1, q+1)R\{1-a\}R\{-(1-b)\} \quad (4.1\text{–}2)$$

In the special case of linear interpolation, , where is defined in Eq. 4.3-2. Typically, for reasons of computational complexity, resampling interpolation is limited to a pixel neighborhood. Figure 13.5-2 defines a generalized bicubic interpolation neighborhood in which the pixel is the nearest neighbor to the pixel to be interpolated. The interpolated pixel may be expressed in the compact form

$$F(p2, q2) = \sum \sum F(\square p + m\square\square, q + n)\square Rc\ \square\{m-a\}\square Rc\{-(\square n-b)\}\square$$

where denotes a bicubic interpolation function such as a cubic B-spline or cubic interpolation function, as defined in Section 4.3-2.

4.2. Convolution Methods

When an image is to be magnified by an integer zoom factor, pixel estimation can be implemented efficiently by convolution (12). As an example, consider image magnification by a factor of 2:1. This operation can be accomplished in two stages. First, the input image is transferred to an array in which rows and columns of zeros are interleaved with the input image data as follows:

Peg

$$\begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}$$

Pyramid

$$\frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

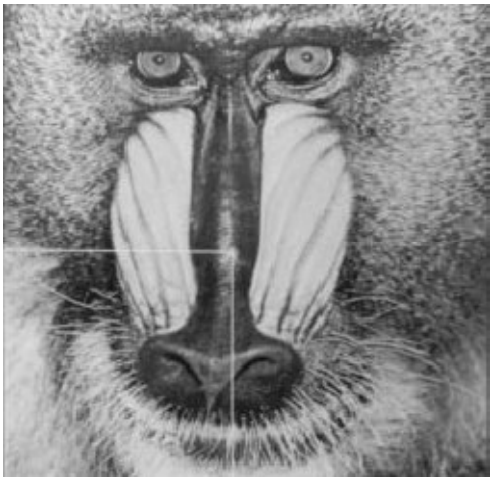
Bell

$$\frac{1}{16}\begin{bmatrix} 1 & 3 & 3 & 1 \\ 3 & 9 & 9 & 3 \\ 3 & 9 & 9 & 3 \\ 1 & 3 & 3 & 1 \end{bmatrix}$$

Cubic B-spline

$$\frac{1}{64}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

FIGURE 4-3. Interpolation kernels for 2:1 magnification.



(a) Original

(b) Zero interleaved quadrant

(c) Peg

(d) Pyramid

(e) Bell

(f)Cubic B-spline

FIGURE 4.5-4. Image interpolation on the `mandrill_mon` image for 2:1 magnification

Next, the zero-interleaved neighborhood image is convolved with one of the discrete interpolation kernels listed in Figure 4-3. Figure 4-4 presents the magnification results for several interpolation kernels. The inevitable visual trade-off between the interpolation error (the jaggy line artifacts) and the loss of high spatial frequency detail in the image is apparent from the examples.

This discrete convolution operation can easily be extended to higher-order magnification factors. For *N*:1 magnification, the core kernel is a peg array. For large kernels it may be more computationally efficient in many cases, to perform the interpolation indirectly by Fourier domain filtering rather than by convolution (6).

IMAGE IMPROVEMENT

The use of digital processing techniques for image improvement has received much interest with the publicity given to applications in space imagery and medical research. Other applications include image improvement for photographic surveys and industrial radiographic analysis.

Image improvement is a term coined to denote three types of image manipulation processes: image enhancement, image restoration, and geometrical image modification.

Image enhancement entails operations that improve the appearance to a human viewer, or operations to convert an image to a format better suited to machine processing. Image restoration has commonly been defined as the modification of an observed image in order to compensate for defects in the imaging system that produced the observed image. Geometrical image modification includes image magnification, minification, rotation, and nonlinear spatial warping.

Chapter 4 describes several techniques of monochrome and color image enhancement. The chapters that follow develop models for image formation and restoration, and present methods of point and spatial image restoration. The final chapter of this part considers geometrical image modification.

5.IMAGE ENHANCEMENT

Image enhancement processes consist of a collection of techniques that seek to improve the visual appearance of an image or to convert the image to a form better suited for analysis by a human or a machine. In an image enhancement system, there is no conscious effort to improve the fidelity of a reproduced image with regard to some ideal form of the image, as is done in image restoration. Actually, there is some evidence to indicate that often a distorted image, for example, an image with amplitude overshoot and undershoot about its object edges, is more subjectively pleasing than a perfectly reproduced original.

For image analysis purposes, the definition of image enhancement stops short of information extraction. As an example, an image enhancement system might emphasize the edge outline of objects in an image by high-frequency filtering. This edge-enhanced image would then serve as an input to a machine that would trace the outline of the edges, and perhaps make measurements of the shape and size of the outline. In this application, the image enhancement processor would emphasize salient features of the original image and simplify the processing task of a data extraction machine.

There is no general unifying theory of image enhancement at present because there is no general standard of image quality that can serve as a design criterion for an image enhancement processor. Consideration is given here to a variety of techniques that have proved useful for human observation improvement and image analysis.

5.1. CONTRAST MANIPULATION

One of the most common defects of photographic or electronic images is poor contrast resulting from a reduced, and perhaps nonlinear, image amplitude range. Image contrast can often be improved by amplitude rescaling of each pixel (1,2).

Figure 5.1-1*a* illustrates a transfer function for contrast enhancement of a typical continuous amplitude low-contrast image. For continuous amplitude images, the transfer function operator can be implemented by photographic techniques, but it is often difficult to realize an arbitrary transfer function accurately. For quantized amplitude images, implementation of the transfer function is a

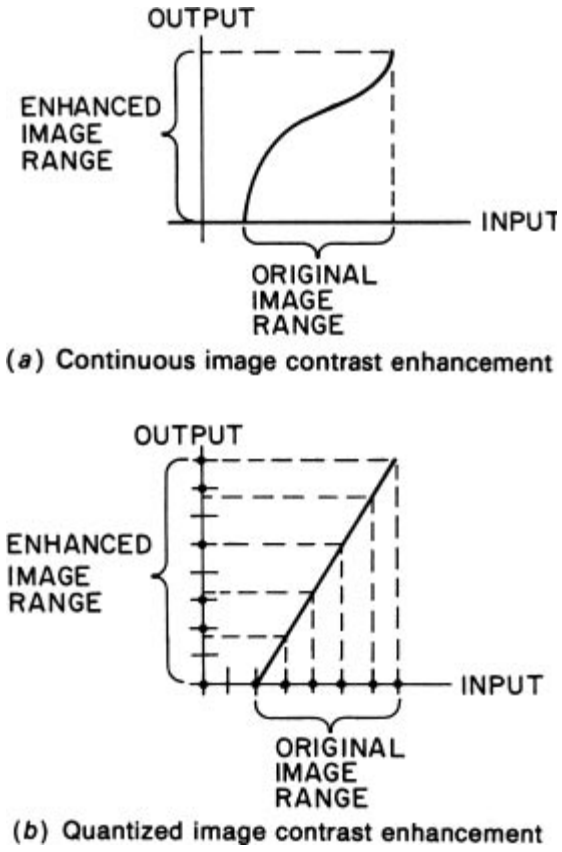
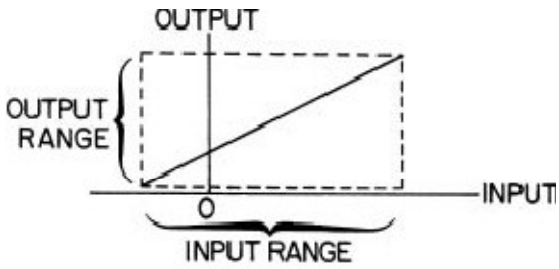
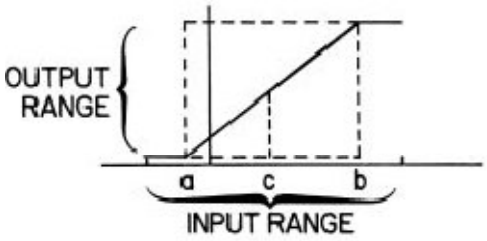
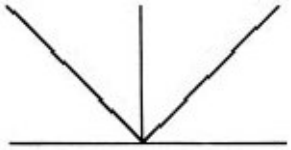


FIGURE 5.1-1. Continuous and quantized image contrast enhancement.

relatively simple task. However, in the design of the transfer function operator, consideration must be given to the effects of amplitude quantization. With reference to Figure 5.l-l*b*, suppose that an original image is quantized to *J* levels, but it occupies a smaller range. The output image is also assumed to be restricted to *J* levels, and the mapping is linear. In the mapping strategy indicated in Figure 5.1-1*b*, the output level chosen is that level closest to the exact mapping of an input level. It is obvious from the diagram that the output image will have unoccupied levels within its range, and some of the gray scale transitions will be larger than in the original image. The latter effect may result in noticeable gray scale contouring. If the output image is quantized to more levels than the input image, it is possible to approach a linear placement of output levels, and hence, decrease the gray scale contouring effect.





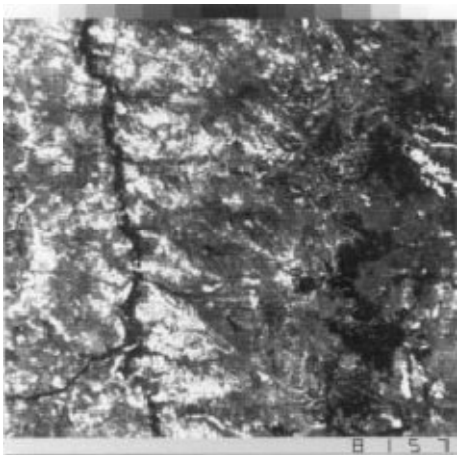


c) Absolute value scaling

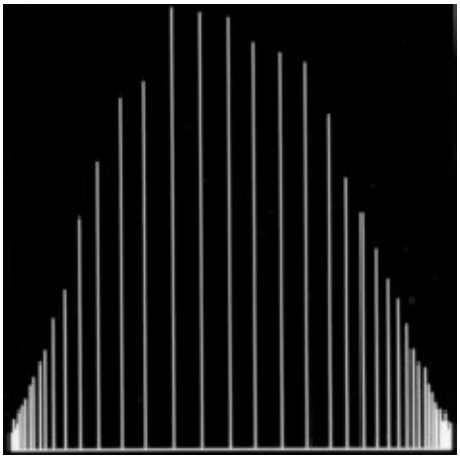
FIGURE 5.1-2. Image scaling methods.

5.1.1. Amplitude Scaling

A digitally processed image may occupy a range different from the range of the original image. In fact, the numerical range of the processed image may encompass negative values, which cannot be mapped directly into a light intensity range. Figure 5.1-2 illustrates several possibilities of scaling an output image back into the domain of values occupied by the original image. By the first technique, the processed image is linearly mapped over its entire range, while by the second technique, the extreme amplitude values of the processed image are clipped to maximum and minimum limits. The second technique is often subjectively preferable, especially for images in which a relatively small number of pixels exceed the limits. Contrast enhancement algorithms often possess an option to clip a fixed percentage of the amplitude values on each end of the amplitude scale. In medical image enhancement applications, the contrast modification operation shown in Figure 5.2-2*b*, for , a ≥ b is called a *window-level transformation*. The window value is the width of the linear slope,b-a ; the level is located at the midpoint *c* of the slope line. The third technique of amplitude scaling, shown in Figure 5.1-2*c*, utilizes an absolute value transformation for visualizing an image with negatively valued pixels. This is a useful transformation for systems that utilize the two's complement



(e) Min. clip = 0.24, max. clip = 0.35



(f) Enhancement histogram

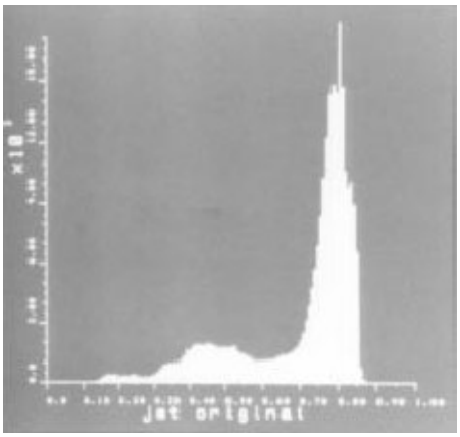
FIGURE 5.1-4. Window-level contrast stretching of an earth satellite image.

5.1.2. Contrast Modification

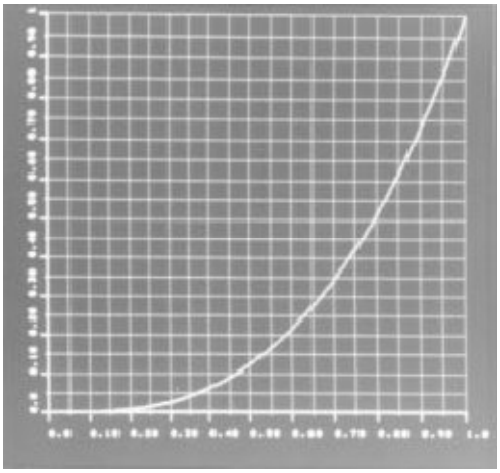
Section 10.1.1 dealt with amplitude scaling of images that do not properly utilize the dynamic range of a display; they may lie partly outside the dynamic range or occupy only a portion of the dynamic range. In this section, attention is directed to point transformations that modify the contrast of an image within a display's dynamic range.

Figure 10.1-5*a* contains an original image of a jet aircraft that has been digitized to 256 gray levels and numerically scaled over the range of 0.0 (black) to 1.0 (white).

(a) Original



(b) Original histogram



(c) Cube function

(d) Cube output

FIGURE 5.1-6. Square and cube contrast modification of the jet_mon image.

$$G(j, k) = [F(j,k)]^p \qquad (5.1\text{-}1)$$

Where $0.0 \leq F(j, k) \leq 1.0$ represents the original image and p is the power law variable. It is important that the amplitude limits of Eq. 5.1-1 be observed; processing of the integer code (e.g., 0 to 255) by Eq. 5.1-1 will give erroneous results. The square function provides the best visual result. The rubber band transfer function shown in Figure 5.1-8*a* provides a simple piecewise linear approximation to the power law curves. It is often useful in interactive enhancement machines in which the inflection point is interactively placed.

The Gaussian error function behaves like a square function for low-amplitude pixels and like a square root function for high- amplitude pixels. It is defined as

$$G(j,k) = \frac{erf\{(F(j,k) - 0.5)/a\sqrt{2}\} + 0.5/a\sqrt{2}}{2\ \mathrm{erf}\{0.5/a\sqrt{2}\}} \qquad (5.1\text{-}2a)$$

IMAGE ANALYSIS

Image analysis is concerned with the extraction of measurements, data or information from an image by automatic or semiautomatic methods. In the literature, this field has been called image data extraction, scene analysis, image description, automatic photo interpretation, image understanding, and a variety of other names.

Image analysis is distinguished from other types of image processing, such as coding, restoration, and enhancement, in that the ultimate product of an image analysis system is usually numerical output rather than a picture.

6.LINEAR PROCESSING TECHNIQUES

Most discrete image processing computational algorithms are linear in nature; an output image array is produced by a weighted linear combination of elements of an input array. The popularity of linear operations stems from the relative simplicity of spatial linear processing as opposed to spatial nonlinear processing. However, for image processing operations, conventional linear processing is often computationally infeasible without efficient computational algorithms because of the large image arrays. This chapter considers indirect computational techniques that permit more efficient linear processing than by conventional methods.

6.1. TRANSFORM DOMAIN PROCESSING

Two-dimensional linear transformations have been defined in Section 5.4 in series form as

$$P(m1, m2) = \sum \sum F(n1, n2)T(n1, n2; m1, m2) \quad (6.1\text{-}1)$$

and defined in vector form as

$$\mathbf{p=Tf} \quad (6.1\text{-}2)$$

It will now be demonstrated that such linear transformations can often be computed more efficiently by an indirect computational procedure utilizing two-dimensional unitary transforms than by the direct computation indicated by Eq. 6.1-1 or 6.1-2.

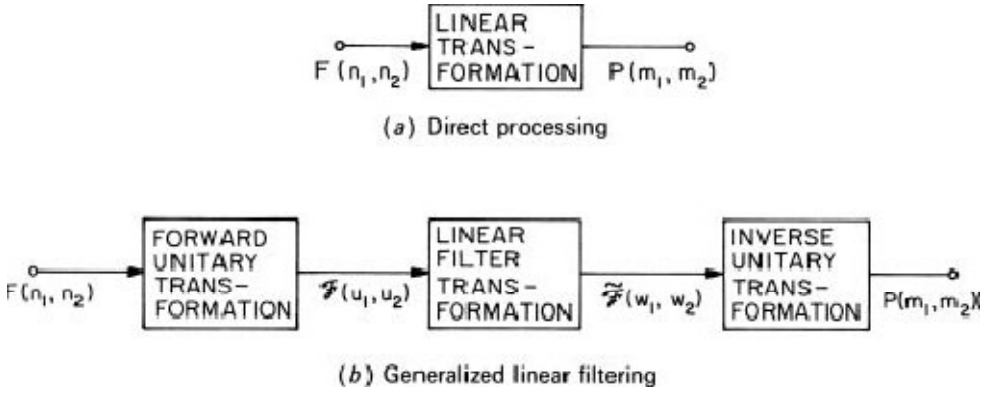


FIGURE 6.1-1. Direct processing and generalized linear filtering; series formulation.

Figure 6.1-1 is a block diagram of the indirect computation technique called *generalized linear filtering* (1). In the process, the input array undergoes a two-dimensional unitary transformation, resulting in an array (n1,n2) of transform coefficients $F(u1, u2)$. Next, a linear combination of these coefficients is taken according to the general relation

$$\tilde{F}(w1,w2) = \sum\sum F(u1,u2)T(u1,u2;w1,w2) \quad (6.1\text{-}3)$$

where $T(u1, u2, ; w1, w2)$ represents the linear filtering transformation function. Finally, an inverse unitary transformation is performed to reconstruct the processed array $P(m1, m2)$. If this computational procedure is to be more efficient than direct computation by Eq. 6.1-1, it is necessary that fast computational algorithms exist for the unitary transformation, and also the kernel must be reasonably sparse; that is, it must contain many zero elements.

The generalized linear filtering process can also be defined in terms of vectorspace computations as shown in Figure 9.1-2. For notational simplicity, let $N1 = N2 = N$ and $M1 = M2 = M$. Then the generalized linear filtering process can be described by the equations

$$f = [A_{N^2}]\, f \quad (6.1\text{-}4a)$$

$$\tilde{f} = Tf \quad (6.1\text{-}4b)$$

$$p=[A_{M^2}]^{-1}\tilde{f} \quad (6.1\text{-}4c)$$

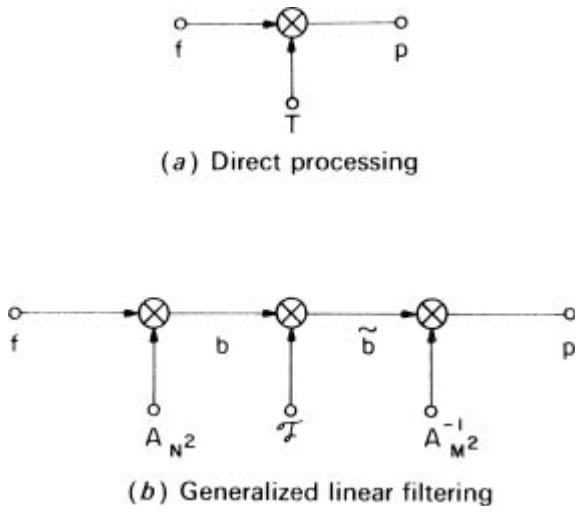


FIGURE 9.1-2. Direct processing and generalized linear filtering; vector formulation.

where A_N^2 is a N^2 x N^2 unitary transform matrix, T is a M^2 x N^2 linear filtering transform operation, and A_{M^2} is a unitary transform matrix. From Eq. 9.1-4, the input and output vectors are related by

$$p=[A_{M^2}]^{-1}T[A_{N^2}]f \quad (6.1\text{-}5)$$

6.2. TRANSFORM DOMAIN SUPERPOSITION

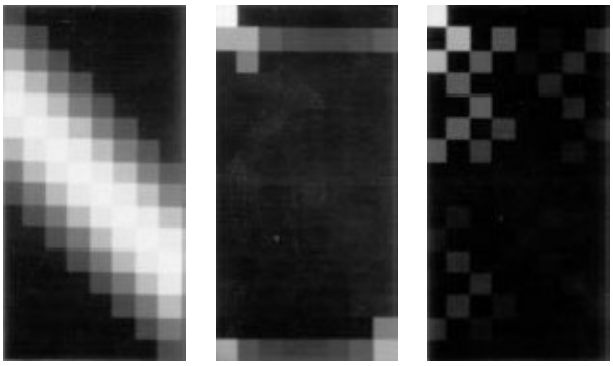
The superposition operations can often be performed more efficiently by transform domain processing rather than by direct processing. Figure 6.2-1*a* and *b* illustrate block diagrams of the computational steps involved in direct finite area or sampled image superposition. In Figure 6.2-1*d* and *e*, an alternative form of processing is illustrated in which a unitary transformation operation is performed on the data vector **f** before multiplication by a finite area filter matrix D or sampled image filter matrix B . An inverse transform reconstructs the output vector. From Figure 6.2-1, for finite-area superposition, because

$$q=Df \quad (6.2\text{-}1a)$$

and

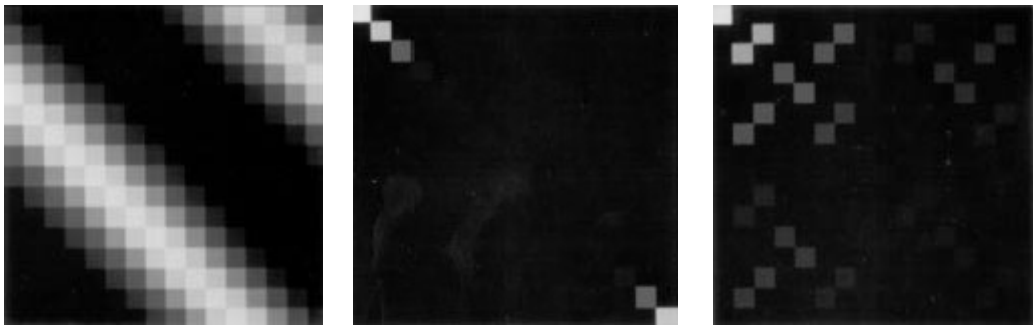
$$\mathbf{q}=[\mathbf{A}_{M}2]-1\mathsf{D}[\mathbf{A}_{N}2]\mathbf{f} \quad (6.2\text{-}1b)$$

then clearly the finite-area filter matrix may be expressed as



(a) Finite length convolution

(b) Sampled data convolution



(c) Circulant convolution

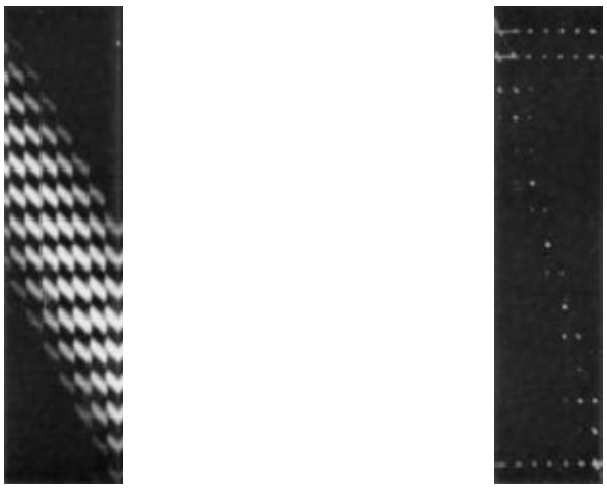
FIGURE 6.2-2. One-dimensional Fourier and Hadamard domain convolution matrices.

Figure 6.2-2 shows the Fourier and Hadamard domain filter matrices for the three forms of convolution for a one-dimensional input vector and a Gaussian-shaped impulse response (6). As expected, the transform domain representations are much more sparse than the data domain representations. Also, the Fourier domain circulant convolution filter is seen to be of diagonal form. Figure 6.2-3 illustrates the structure of the three convolution matrices for two-dimensional convolution (4).

6.3. FAST FOURIER TRANSFORM CONVOLUTION

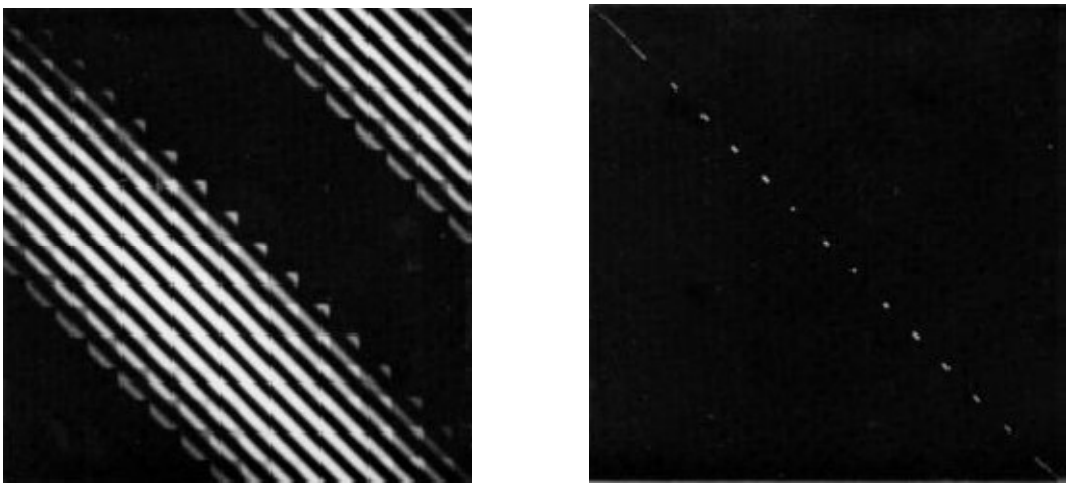
As noted previously, the equivalent output vector for either finite-area or sampled image convolution can be obtained by an element selection operation on the extended output vector **k***E* for circulant convolution or its matrix counterpart **K***E*.

Spatial domain Fourier domain



(a) Finite-area convolution

(b) Sampled image convolution



(c) Circulant convolution

FIGURE 9.2-3. Two-dimensional Fourier domain convolution matrices.

This result, combined with Eq. 6.2-13, leads to a particularly efficient means of convolution computation indicated by the following steps:

1. Embed the impulse response matrix in the upper left corner of an all-zero *JxJ* matrix, for J≥Nfinite-area convolution or for J ≥ M sampled infinite-area convolution, and take the two-dimensional Fourier transform of the extended impulse response matrix, giving

$$H_E=A_J H_E A_J \tag{6.3-1}$$

2. Embed the input data array in the upper left corner of an all-zero matrix, and take the two-dimensional Fourier transform of the extended input data matrix to obtain

$$F_E=A_J H_E A_J \tag{6.3-2}$$

3. Perform the scalar multiplication

$$KE(m,n) = JHE(m,n)FE(m,n) \tag{6.3-3}$$

4. Take the inverse Fourier transform

$$K_E=[A_J^2]^{-1}H_E[A_J^2] \tag{6.3-4}$$

5. Extract the desired output matrix

$$Q=[S1_J^{(M)}]K_E[S1_J^{(M)}]^T \tag{6.3-5}$$

It is important that the size of the extended arrays in steps 1 and 2 be chosen large enough to satisfy the inequalities indicated. If the computational steps are performed with *J* = *N*, the resulting output array, shown in Figure 9.3-1, will contain erroneous terms in a boundary region of width *L* – 1 elements, on the top and left-hand side of the output field. This is the *wraparound error* associated with incorrect use of the Fourier domain convolution method. In addition, for finite area (*D*-type) convolution, the bottom and right-hand-side strip of output elements will be missing. If the computation is performed with *J* = *M*, the output array will be completely filled with the correct terms for *D*-type convolution. To force *J* = *M* for *B*-type convolution, it is necessary to truncate the bottom and right-hand side of the input array. As a consequence, the top and left-hand-side elements of the output array are erroneous.

Fourier domain processing is more computationally efficient than direct processing for image convolution if the impulse response is sufficiently large. However, if the image to be processed is large, the relative computational advantage of Fourier domain processing diminishes. Also, there are attendant problems of computational

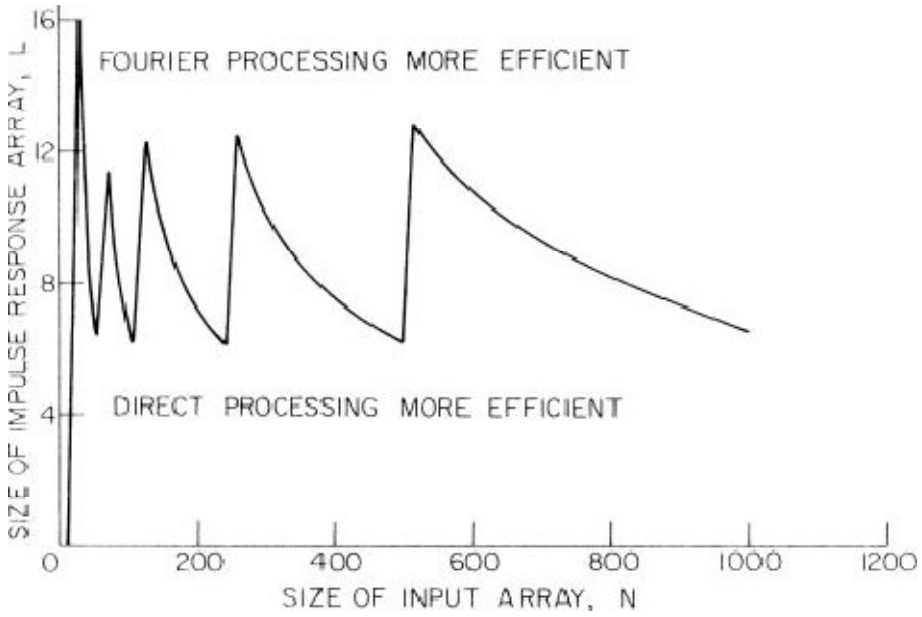


FIGURE 6.3-5. Comparison of direct and Fourier domain processing for finite-area convolution.

accuracy with large Fourier transforms. Both difficulties can be alleviated by a block-mode filtering technique in which a large image is separately processed in adjacent overlapped blocks (2, 7–9).

Figure 9.3-6*a* illustrates the extraction of a pixel block from the upper left corner of a large image array. After convolution with a impulse response, the resulting pixel block is placed in the upper left corner of an output

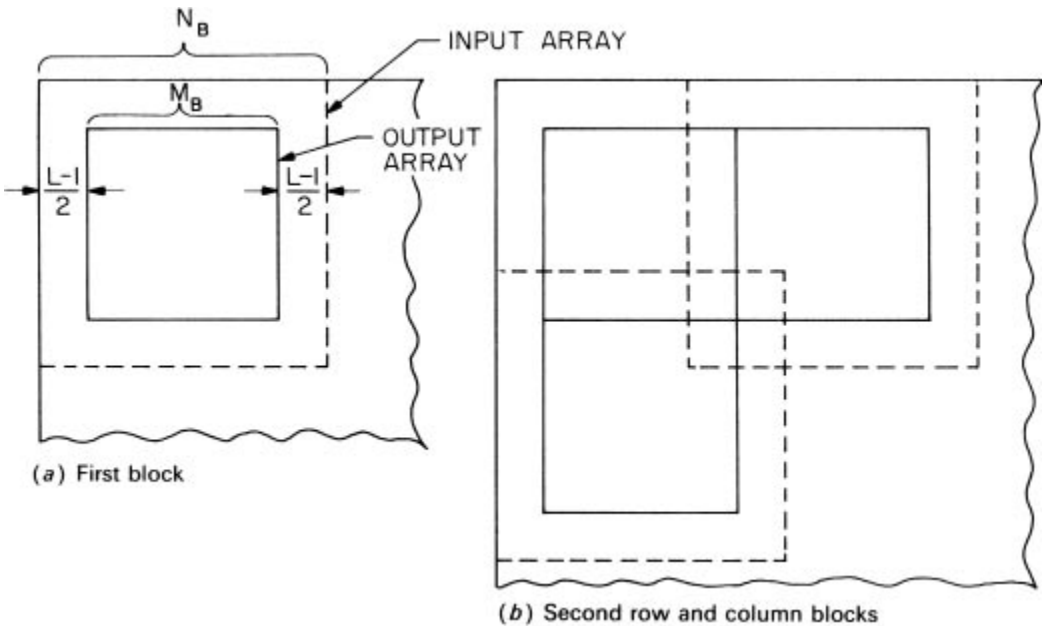


FIGURE 6.3-5. Comparison of direct and Fourier domain processing for finite-area convolution.

data array as indicated in Figure 6.3-6*a*. Next, a second block of pixels is extracted from the input array to produce a second block of output pixels that will lie adjacent to the first block. As indicated in Figure 6.3-6*b*, this second input block must be overlapped by (L – 1) pixels in order to generate an adjacent output block. The computational process then proceeds until all input blocks are filled along the first row. If a partial input block remains along the row, zero-value elements can be added to complete the block. Next, an input block, overlapped by (L –1) pixels with the first row blocks, is extracted to produce the first block of the second output row. The algorithm continues in this fashion until all output points are computed.

7.IMAGE RESTORATION MODELS

Image restoration may be viewed as an estimation process in which operations are performed on an observed or measured image field to estimate the ideal image field that would be observed if no image degradation were present in an imaging system. Mathematical models are described in this chapter for image degradation in general classes of imaging systems. These models are then utilized in subsequent chapters as a basis for the development of image restoration techniques.

7.1. GENERAL IMAGE RESTORATION MODELS

In order effectively to design a digital image restoration system, it is necessary quantitatively to characterize the image degradation effects of the physical imaging system, the image digitizer, and the image display. Basically, the procedure is to model the image degradation effects and then perform operations to undo the model to obtain a restored image. It should be emphasized that accurate image modeling is often the key to effective image restoration. There are two basic approaches to the modeling of image degradation effects: a priori modeling and a posteriori modeling.

In the former case, measurements are made on the physical imaging system, digitizer, and display to determine their response for an arbitrary image field. In some instances it will be possible to model the system response deterministically, while in other situations it will only be possible to determine the system response in a stochastic sense. The a posteriori modeling approach is to develop the model for the image degradations based on measurements of a particular image to be restored.

Basically, these two approaches differ only in the manner in which information is gathered to describe the character of the image degradation.

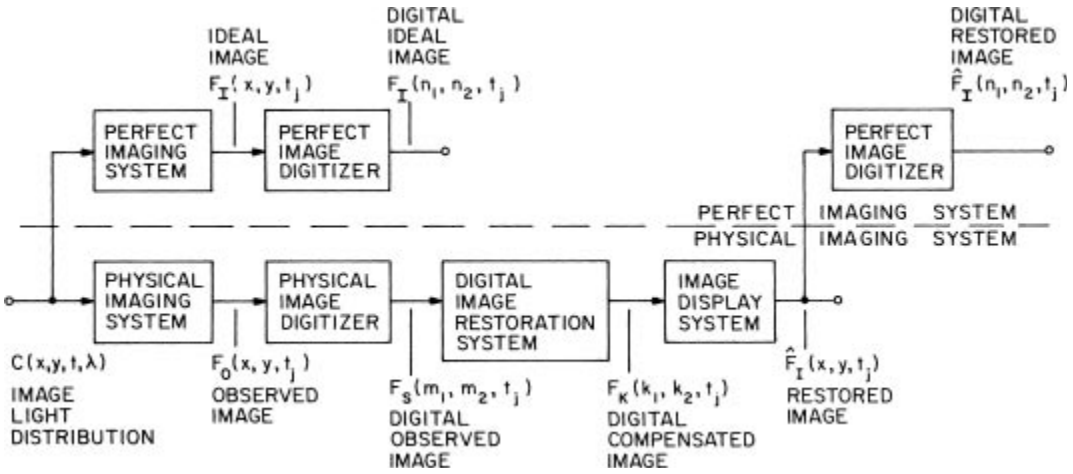


FIGURE 7.1-1. Digital image restoration model.

Figure 7.1-1 shows a general model of a digital imaging system and restoration process. In the model, a continuous image light distribution dependent on spatial coordinates (*x*, *y*), time (*t*), and spectral wavelength is assumed to exist as the driving force of a physical imaging system subject to point and spatial degradation effects and corrupted by deterministic and stochastic disturbances. Potential degradations include diffraction in the optical system, sensor nonlinearities, optical system aberrations, film nonlinearities, atmospheric turbulence effects, image motion blur, and geometric distortion. Noise disturbances may be caused by electronic imaging sensors or film granularity. In this model, the physical imaging system produces a set of output image fields at time instant described by the general relation

$$FO^{(i)}(x,y,tj) = OP\{C(x, y, t, \lambda)\} \qquad (7.1\text{–}1)$$

where represents a general operator that is dependent on the space coordinates (*x*, *y*), the time history (*t*), the wavelength , and the amplitude of the light distribution (*C*). For a monochrome imaging system, there will only be a single output field, while for a natural color imaging system, may denote the red, green, and blue tristimulus bands for *i* = 1, 2, 3, respectively. Multispectral imagery may also involve several output bands of data.

In the general model of Figure 7.1-1, each observed image field is digitized, following the techniques outlined in Part 3, to produce an array of image samples at each time instant . The output samples of the digitizer are related to the input observed field by

$$FS^{(i)}(m1,m2,tj) = OG\{FO^{(i)}(x,y,tj)\} \qquad (7.1\text{–}2)$$

where is an operator modeling the image digitization process. A digital image restoration system that follows produces an output array by the transformation

$$\hat{F}^{(i)}(x,y,tj) = OD\{FK^{(i)}(k1,k2,tj)\} \qquad (7.1\text{–}3)$$

where $O_D\{.\}$models the display transformation.

7.2. OPTICAL SYSTEMS MODELS

One of the major advances in the field of optics during the past 40 years has been the application of system concepts to optical imaging. Imaging devices consisting of lenses, mirrors, prisms, and so on, can be considered to provide a deterministic transformation of an input spatial light distribution to some output spatial light distribution.
Also, the system concept can be extended to encompass the spatial propagation of light through free space or some dielectric medium.
In the study of geometric optics, it is assumed that light rays always travel in a straight-line path in a homogeneous medium. By this assumption, a bundle of rays passing through a clear aperture onto a screen produces a geometric light projection of the aperture. However, if the light distribution at the region between the light and

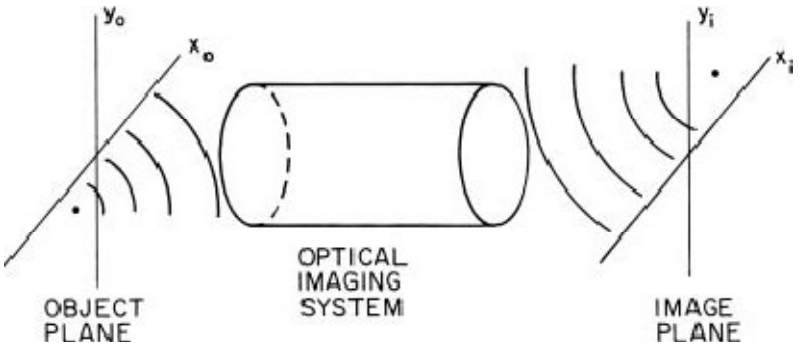


FIGURE 11.2-1. Generalized optical imaging system.

dark areas on the screen is examined in detail, it is found that the boundary is not sharp. This effect is more pronounced as the aperture size is decreased. For a pinhole aperture, the entire screen appears diffusely illuminated. From a simplistic viewpoint, the aperture causes a bending of rays called *diffraction*. Diffraction of light can be quantitatively characterized by considering light as electromagnetic radiation that satisfies Maxwell's equations. The formulation of a complete theory of optical imaging from the basic electromagnetic principles of diffraction theory is a complex and lengthy task. In the following, only the key points of the formulation are presented; details may be found in References 1 to 3.

Figure 7.2-1 is a diagram of a generalized optical imaging system. A point in the object plane at coordinate □*xo*□□*yo*□□of intensity *Io (xo, yo* □radiates energy toward an imaging system characterized by an entrance pupil, exit pupil, and intervening system transformation. Electromagnetic waves emanating from the optical system are focused to a point on the image

plane producing an intensity *Ii (xi, yi* . The imaging system is said to be *diffraction limited* if the light distribution at the image plane produced by a point-source object consists of a converging spherical wave whose extent is limited only by the exit pupil. If the wavefront of the electromagnetic radiation emanating from the exit pupil is not spherical, the optical system is said to possess *aberrations*.

7.3. DISCRETE IMAGE RESTORATION MODELS

This chapter began with an introduction to a general model of an imaging system and a digital restoration process. Next, typical components of the imaging system were described and modeled within the context of the general model. Now, the discussion turns to the development of several discrete image restoration models. In the development of these models, it is assumed that the spectral wavelength response and temporal response characteristics of the physical imaging system can be separated from the spatial and point characteristics. The following discussion considers only spatial and point characteristics.
After each element of the digital image restoration system of Figure 7.1-1 is modeled, following the techniques described previously, the restoration system may be conceptually distilled to three equations:

Observed image:

$$FS(m1, m2) = OM\{FI(n1,n2), N1(m1,m2).....,NN(m1m2)\} \qquad (7.3\text{–}1\alpha)$$

Compensated image:

$$FK(k1,k2) = OR\{FS(m1,m2)\} \qquad (7.3\text{–}1\beta)$$

Restored image:

$$\hat{F}I(n1,n2) = OD\{(FK(k1,k2)\} \qquad (7.3\text{–}1c)$$

where *FS* represents an array of observed image samples, *FI* and are arrays of ideal image points and estimates, respectively, *FK* is an array of compensated image points from the digital restoration system, *Ni* denotes arrays of noise samples from various system elements, and , , represent general transfer functions of the imaging system, restoration processor, and display system, respectively.
Vector-space equivalents of Eq. 7.4-1 can be formed for purposes of analysis by column scanning of the arrays of Eq. 7.3-1.
Several estimation approaches to the solution of 7.3-1 or 7.3-2 are described in the following chapters. Unfortunately, general solutions have not been found; recourse must be made to specific solutions for less general models.

The most common digital restoration model is that of Figure 7.3-1*a,* in which a continuous image field is subjected to a linear blur, the electrical sensor responds nonlinearly to its input intensity, and the sensor amplifier introduces additive Gaussian noise independent of the image

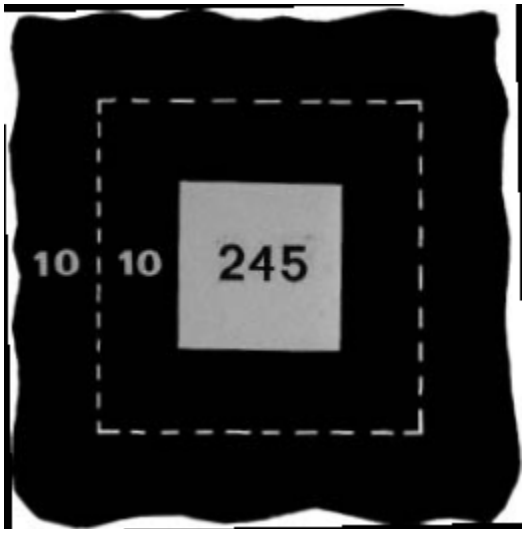
$$fP(i) = OP\{fB(i)\} \qquad (7.3–2)$$

Equation for the observed physical image samples in terms of points on the ideal image:

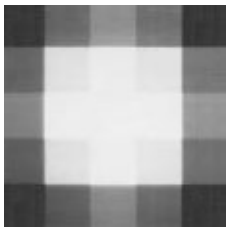
$$\mathbf{f}S=\mathbf{B}\mathrm{POP}\{\mathbf{B}\mathrm{Bf}\mathrm{I}\}\mathbf{B}P\mathbf{n} \qquad (7.3\text{-}3)$$

Several special cases of Eq. 11.4-5 will now be defined. First, if the point nonlinearity is absent,

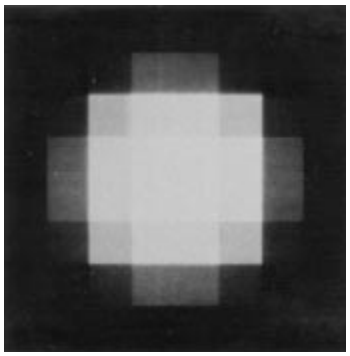
$$\mathbf{f}S=\mathbf{B}\mathbf{f}+\mathbf{n}B \qquad (7.3\text{-}4)$$



(a) Original



(b) Impulse response



(c) Observation

where **B** = **B***P***B***B* and **n***B* = **B***P***n**. This is the classical discrete model consisting of a set of linear equations with measurement uncertainty. Another case that will be defined for later discussion occurs when the spatial blur of the physical image digitizer is negligible.

Chapter 8 contains results for several image restoration experiments based on the restoration model defined by Eq. 7.3-6. An artificial image has been generated for these computer simulation experiments (9). The original image used for the analysis of underdetermined restoration techniques, shown in Figure 7.3-3*a*, consists of a pixel square of intensity 245 placed against an extended background of intensity 10 referenced to an intensity scale of 0 to 255. All images are zoomed for display purposes.

In the computer simulation restoration experiments, the observed blurred image model has been obtained by multiplying the column-scanned original image of Figure 7.3-3*a* by the blur matrix **B**. Next, additive white Gaussian observation noise has been simulated by adding output variables from an appropriate random number generator to the blurred images. For display, all image points restored are clipped to the intensity range 0 to 255.

8.IMAGE DETECTION AND REGISTRATION

Image detection is concerned with the determination of the presence or absence of objects suspected of being in an image.

TEMPLATE MATCHING

One of the most fundamental means of object detection within an image field is by *template matching*, in which a replica of an object of interest is compared to all unknown objects in the image field (1–4). If the template match between an unknown object and the template is sufficiently close, the unknown object is labeled as the template object.

As a simple example of the template-matching process, consider the set of binary black line figures against a white background as shown in Figure 8.1-1*a*. In this example, the objective is to detect the presence and location of right triangles in the image field. Figure 8.1-1*b* contains a simple template for localization of right triangles that possesses unit value in the triangular region and zero elsewhere. The width of the legs of the triangle template is chosen as a compromise between localization accuracy and size invariance of the template. In operation, the template is sequentially scanned over the image field and the common region between the template and image field is compared for similarity.

A template match is rarely ever exact because of image noise, spatial and amplitude quantization effects, and a priori uncertainty as to the exact shape and structure of an object to be detected. Consequently, a common procedure is to produce a difference measure between the template $D\square m\square\square n\square\square$and the image field at all points of the image field where and denote the trial offset. An object is deemed to be matched wherever the difference is smaller than some established level . Normally, the threshold level is constant over the image field. The usual difference measure is the mean-square difference or error as defined by

$$D(m, n) = \sum \sum [F(j,k) - T(j-m, k-n)]^2 \qquad (8.1)$$

where $F(j,k)$ denotes the image field to be searched and $T(j,k)$ is the template. The search, of course, is restricted to the overlap region between the translated template and the image field. A template match is then said to exist at coordinate (m,n) if

$$D(m, n) < L_D(m,n) \tag{8.2}$$

Now, let Eq. 12.1 be expanded to yield

$$D(m, n) = D_1(m,n) - 2D_2(m,n) + D_3(m,n) \tag{8.3}$$

$$D_1(m,n) = \sum \sum [F(j,k)]^2 \tag{8.4a}$$

$$D_2(m,n) = \sum \sum [F(j,k)T(j-m, k-n)] \tag{8.4β}$$

$$D_3(m,n) = \sum \sum [T(j-m,k-n)]^2 \tag{8.4c}$$

The term $D3\ (m,n)$ represents a summation of the template energy. It is constant valued and independent of the coordinate (m,n) . The image energy over the window area represented by the first term $D1\ (m,n)$ generally varies rather slowly over the image field. The second term should be recognized as the cross correlation $RFT\ (m,n)$ between the image field and the template. At the coordinate location of a template match, the cross correlation should become large to yield a small difference.

However, the magnitude of the cross correlation is not always an adequate measure of the template difference because the image energy term $D1\ (m,n)$ is position variant. For example, the cross correlation can become large, even under a condition of template mismatch, if the image amplitude over the template region is high about a particular coordinate (m,n) . This difficulty can be avoided by comparison of the normalized cross correlation

$$\widetilde{R_{FT}}(m,n) = \frac{D1(m,n)}{D2(m,n)} = \frac{\sum \sum F(j,k)T(j-m,k-n)}{\sum \sum [F(j,k)]^2}$$

to a threshold level . A template match is said to exist if

$$\tilde{R}_{FT}(m,n) > L_R(m,n) \tag{8.5}$$

The normalized cross correlation has a maximum value of unity that occurs if and only if the image function under the template exactly matches the template.

One of the major limitations of template matching is that an enormous number of templates must often be test matched against an image field to account for changes in rotation and magnification of template objects. For this reason, template matching is usually limited to smaller local features, which are more invariant to size and shape variations of an object. Such features, for example, include edges joined in a Y or T arrangement.

9.MORPHOLOGICAL IMAGE PROCESSING

Morphological image processing is a type of processing in which the spatial form or structure of objects within an image are modified. Dilation, erosion, and skeletonization are three fundamental morphological operations. With dilation, an object grows uniformly in spatial extent, whereas with erosion an object shrinks uniformly. Skeletonization results in a stick figure representation of an object. The basic concepts of morphological image processing trace back to the research on spatial set algebra by Minkowski (1) and the studies of Matheron (2) on topology. Serra (3–5) developed much of the early foundation of the subject. Steinberg (6,7) was a pioneer in applying morphological methods to medical and industrial vision applications. This research work led to the development of the cytocomputer for high-speed morphological image processing (8,9). In the following sections, morphological techniques are first described for binary images. Then these morphological concepts are extended to gray scale images.

9.1. BINARY IMAGE CONNECTIVITY

Binary image morphological operations are based on the geometrical relationship or *connectivity* of pixels that are deemed to be of the same class (10,11). In the binary image of Figure 9.1-1*a*, the ring of black pixels, by all reasonable definitions of connectivity, divides the image into three segments: the white pixels exterior to the ring, the white pixels interior to the ring, and the black pixels of the ring itself. The pixels within each segment are said to be connected to one another. This concept of connectivity is easily understood for Figure 14.1-1*a*, but ambiguity arises when considering Figure 9.1-1*b*. Do the black pixels still define a ring, or do they instead form four disconnected lines? The answers to these questions depend on the definition of connectivity.

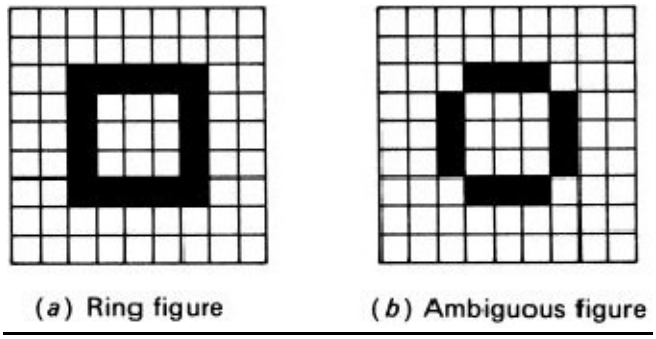


FIGURE 9.1-1. Connectivity.

Consider the following neighborhood pixel pattern:

$$\begin{matrix} X_3 & X_2 & X_1 \\ X_4 & X & X_0 \\ X_5 & X_6 & X_7 \end{matrix}$$

in which a binary-valued pixel , where $X = 0$ (white) or $X = 1$ (black) is surrounded by its eight nearest neighbors . An alternative nomenclature is to label the neighbors by compass directions: north, northeast, and so on:

NW N NE
W X E
SW S SE

Pixel X is said to be *four-connected* to a neighbor if it is a logical 1 and if its east, north, west, or south neighbor is a logical 1. Pixel X is said to be *eight-connected* if it is a logical 1 and if its north, northeast, etc. neighbor is a logical 1.

The connectivity relationship between a center pixel and its eight neighbors canbe quantified by the concept of a *pixel bond*, the sum of the bond weights betweenthe center pixel and each of its neighbors. Each four-connected neighbor has a bondof two, and each eight-connected neighbor has a bond of one. In the followingexample, the pixel bond is seven.

$$\begin{matrix} 1 & 1 & 1 \\ 0 & X & 0 \\ 1 & 1 & 0 \end{matrix}$$

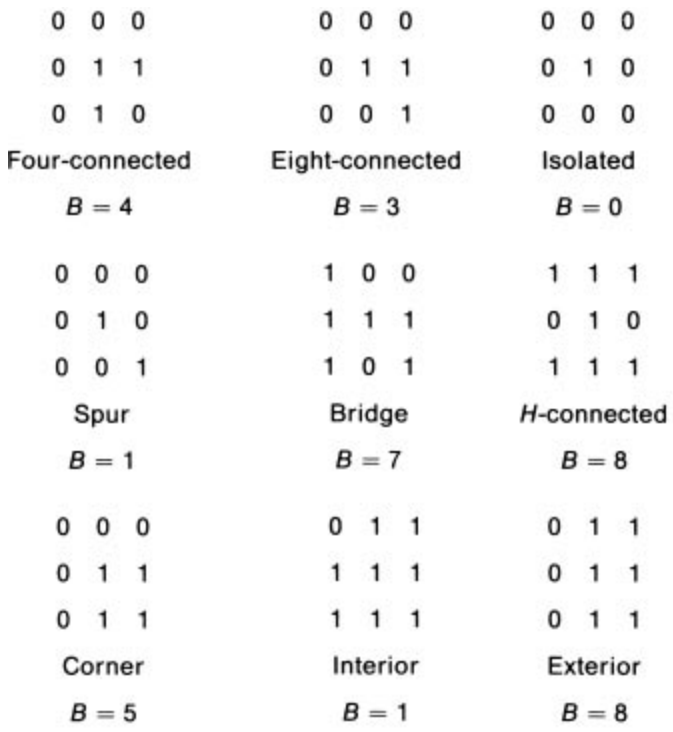


FIGURE 9.1-2. Pixel neighborhood connectivity definitions.

Under the definition of four-connectivity, Figure 9.1-1*b* has four disconnected black line segments, but with the eight-connectivity definition, Figure 9.1-1*b* has a ring of connected black pixels. Note, however, that under eight-connectivity, all white pixels are connected together. Thus a paradox exists. If the black pixels are to be eight-connected together in a ring, one would expect a division of the white pixels into pixels that are interior and exterior to the ring. To eliminate this dilemma, eight-connectivity can be defined for the black pixels of the object, and four-connectivity can be established for the white pixels of the background. Under this definition, a string of black pixels is said to be *minimally connected* if elimination of any black pixel results in a loss of connectivity of the remaining black pixels. Figure 9.1-2 provides definitions of several other neighborhood connectivity relationships between a center black pixel and its neighboring black and white pixels.

The preceding definitions concerning connectivity have been based on a discrete image model in which a continuous image field is sampled over a rectangular array of points. Golay (12) has utilized a hexagonal grid structure. With such a structure, many of the connectivity problems associated with a rectangular grid are eliminated.

In a hexagonal grid, neighboring pixels are said to be *six-connected* if they are in the same set and share a common edge boundary. Algorithms have been developed for the linking of boundary points for many feature extraction tasks (13). However, two major drawbacks have hindered wide acceptance of the hexagonal grid. First, most image scanners are inherently limited to rectangular scanning. The second problem is that the hexagonal grid is not well suited to many spatial processing operations, such as convolution and Fourier transformation.

9.2. BINARY IMAGE HIT OR MISS TRANSFORMATIONS

The two basic morphological operations, dilation and erosion, plus many variants can be defined and implemented by *hit-or-miss transformations* (3). The concept is quite simple. Conceptually, a small odd-sized mask, typically 3 x 3, is scanned over a binary image. If the binary-valued pattern of the mask matches the state of the pixels under the mask (hit), an output pixel in spatial correspondence to the center pixel of the mask is set to some desired binary state. For a pattern mismatch (miss), the output pixel is set to the opposite binary state. For example, to perform simple binary noise cleaning, if the isolated 3 x3 pixel pattern

$$\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \end{matrix}$$

is encountered, the output pixel is set to zero; otherwise, the output pixel is set to the state of the input center pixel. In more complicated morphological algorithms, a large number of the 2^9 = 512 possible mask patterns may cause hits.

It is often possible to establish simple neighborhood logical relationships that define the conditions for a hit. In the isolated pixel removal example, the defining equation for the output pixel G(j,k)becomes

$$G(j,k) = X(X_0 \cap X_1 \cap \ldots\ldots\ldots\cap X_7) \qquad (9.2\text{-}1)$$

Where $\cap$ denotes the intersection operation (logical AND) and $\cup$ denotes the union operation (logical OR). For complicated algorithms, the logical equation method of definition can be cumbersome. It is often simpler to regard the hit masks as a collection of binary patterns.

Hit-or-miss morphological algorithms are often implemented in digital image processing hardware by a pixel stacker followed by a look-up table (LUT), as shown in Figure 9.2-1 (14). Each pixel of the input image is a positive integer, represented by a conventional binary code, whose most significant bit is a 1 (black) or a 0 (white). The pixel stacker extracts the bits of the center pixel *X* and its eight neighbors and puts them in a neighborhood pixel stack. Pixel stacking can be performed by convolution with the 3x3 pixel kernel

$$\begin{matrix} 2^{-4} & 2^{-3} & 2^{-2} \\ 2^{-5} & 2^{0} & 2^{-1} \\ 2^{-6} & 2^{-7} & 2^{-8} \end{matrix}$$

The binary number state of the neighborhood pixel stack becomes the numeric input address of the LUT whose entry is *Y* For isolated pixel removal, integer entry 256, corresponding to the neighborhood pixel stack state 100000000, contains *Y* = 0; all other entries contain *Y* = *X*.

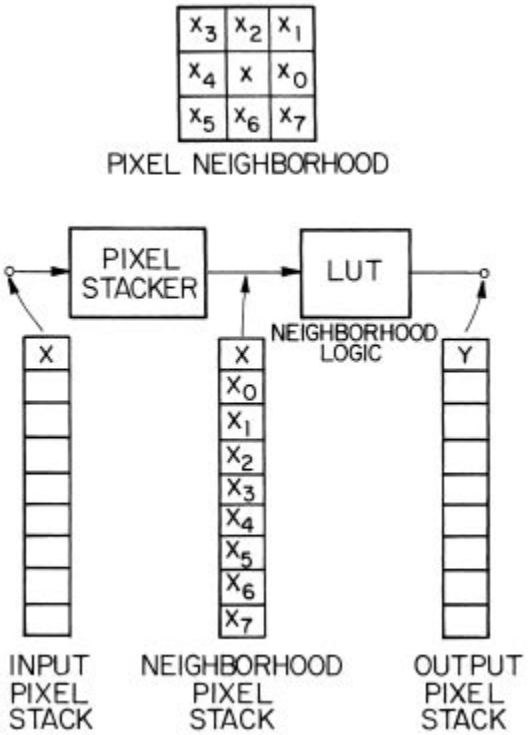


FIGURE 14.2-1. Look-up table flowchart for binary unconditional operations

Several 3x3 other hit-or-miss operators are described in the following subsections.

9.2.1. Additive Operators

Additive hit-or-miss morphological operators cause the center pixel of a pixelwindow to be converted from a logical 0 state to a logical 1 state if the neighboringpixels meet certain predetermined conditions. The basic operators are now defined.

Interior Fill. Create a black pixel if all four-connected neighbor pixels are black.

$$G(j, k) = X \cup [X_0 \cap X_2 \cap X_4 \cap X_6] \quad (9.2\text{-}2)$$

Diagonal Fill. Create a black pixel if creation eliminates the eight-connectivity of the background.

$$G(j, k) = X \cup [\cup P_1 \cup P_2 \cup P_3 \cup P_4] \quad (9.2\text{-}3a)$$

FIGURE 9.2-1. Look-up table flowchart for binary unconditional operations.where

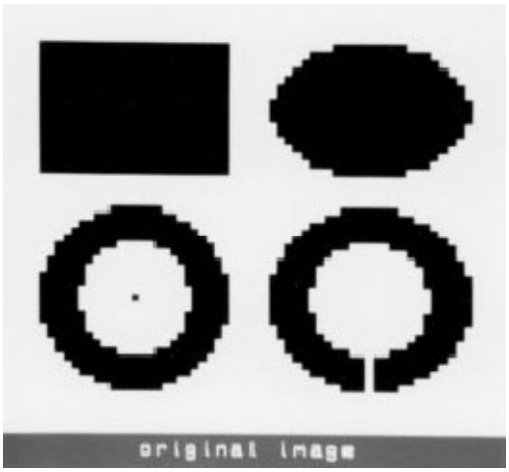
$$P_1 = X \cap X_0 \cap X_1 \cap X_2 \quad (9.2\text{-}3b)$$

$$P_2 = X \cap X_2 \cap X_3 \cap X_4 \quad (9.2\text{-}3b)$$

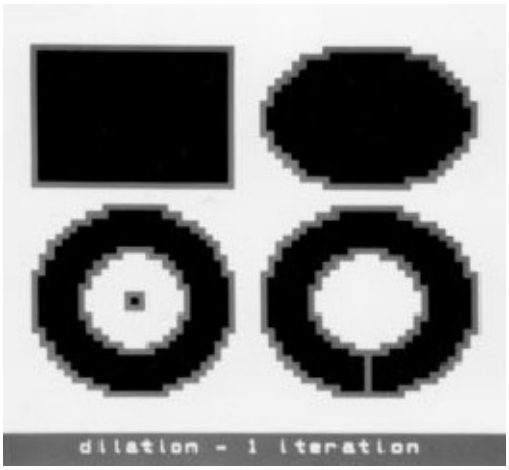
$$P_3 = X \cap X_4 \cap X_5 \cap X_6 \quad (9.2\text{-}3d)$$

$$P_4 = X \cap X_6 \cap X_7 \cap X_0 \quad (9.2\text{-}3e)$$

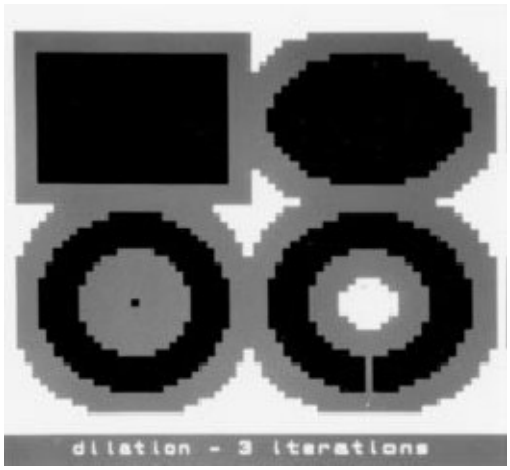
There are 132 such qualifying patterns. This strategem will not prevent connection of two objects separated by two rows or columns of white pixels. A solution to this problem is considered in Section 9.3. Figure 9.2-3 provides an example of fattening.



(a) Original



(b) One iteration



(c) Three iterations

FIGURE 9.2-2. Dilation of a binary image

9.2.2. Subtractive Operators

Subtractive hit-or-miss morphological operators cause the center pixel of a 3 x 3 window to be converted from black to white if its neighboring pixels meet predetermined conditions. The basic subtractive operators are defined below.

Isolated Pixel Remove: Erase a black pixel with eight white neighbors.

$$G(j,k) = X \cup [X_0 \cap X_1 \cap X_2 \cap \ldots \cap X_7] \quad (9.2\text{-}6)$$

Spur Remove : Erase a black pixel with a single eight-connected neighbor.

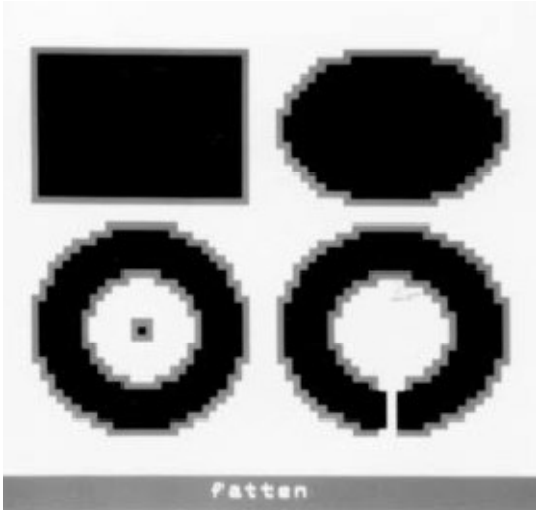


FIGURE 9.2-3. Fattening of a binary image

The following is one of four qualifying patterns:

$$\begin{matrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{matrix}$$

Interior Pixel Remove: Erase a black pixel if all four-connected neighbors are black.

$$G(j,k) = X \cup [X_0 \cap X_2 \cap X_4 \cap X_6] \quad (9.2\text{-}7)$$

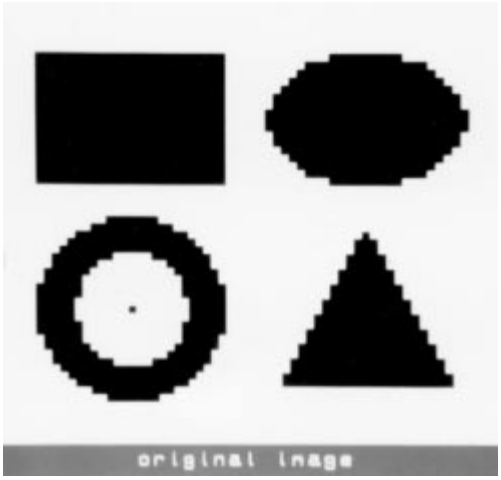
There are 16 qualifying patterns.

H-Break: Erase a black pixel that is H-connected. There are two qualifying patterns.

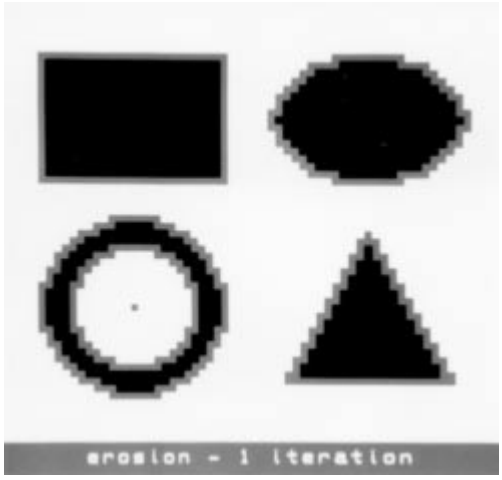
$$\begin{matrix} 1 & 1 & 1 \\ 0 & 1 & 0 \\ 1 & 1 & 1 \end{matrix} \qquad \begin{matrix} 1 & 0 & 1 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{matrix}$$

Eight-Neighbor Erode: Erase a black pixel if at least one eight-connected neighbor pixel is white.

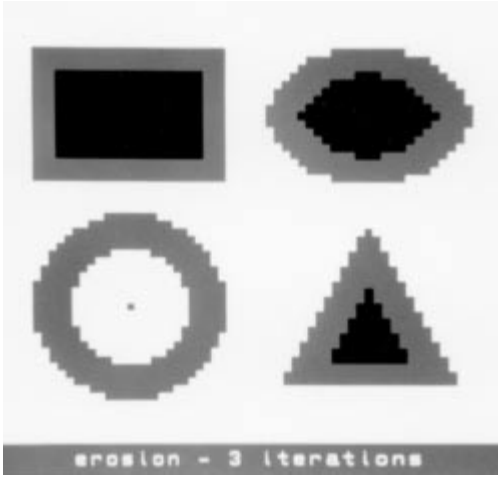
$$G(j,k) = X \cap X0 \cap \ldots \cap X7 \qquad (9.2\text{-}8)$$



(a) Original image



(b) One iteration



(c) Three iterations

FIGURE 9.2-4. Erosion of a binary image.

A generalized erosion operator is defined in Section 9.4. Recursive application of the erosion operator will eventually erase all black pixels. Figure 9.2-4 shows results for one and three iterations of the erode operator. The eroded pixels are midgray. It should be noted that after three iterations, the ring is totally eroded.

9.2.3. Majority Black Operator

The following is the definition of the *majority black operator:*

Majority Black. Create a black pixel if five or more pixels in a 3x 3 window are black; otherwise, set the output pixel to white. The majority black operator is useful for filling small holes in objects and closing short gaps in strokes.

9.3. BINARY IMAGE SHRINKING, THINNING, SKELETONIZING, AND THICKENING

Shrinking, thinning, skeletonizing, and thickening are forms of conditional erosion in which the erosion process is controlled to prevent total erasure and to ensure connectivity.

9.3.1. Binary Image Shrinking

The following is a definition of *shrinking:*

Shrink. Erase black pixels such that an object without holes erodes to a single pixel at or near its center of mass, and an object with holes erodes to a connected ring lying midway between each hole and its nearest outer boundary.

A 3 x 3 pixel object will be shrunk to a single pixel at its center. A 2 x 2 pixel object will be arbitrarily shrunk, by definition, to a single pixel at its lower right corner.

It is not possible to perform shrinking using single-stage 3 x 3 pixel hit-or-miss transforms of the type described in the previous section. The 3 x 3 window does not provide enough information to prevent total erasure and to ensure connectivity. A 5 x 5 hit-or-miss transform could provide sufficient information to perform proper shrinking. But such an approach would result in excessive computational complexity (i.e., 225 possible patterns to be examined!). References 15 and 16 describe twostage shrinking and thinning algorithms that perform a conditional marking of pixels for erasure in a first stage, and then examine neighboring marked pixels in a second stage to determine which ones can be unconditionally erased without total erasure or loss of connectivity. The following algorithm developed by Pratt and Kabir (17) is a pipeline processor version of the conditional marking scheme.

In the algorithm, two concatenated 3 x 3 hit-or-miss transformations are performed to obtain indirect information about pixel patterns within a 5 x 5window. Figure 9.3-1 is a flowchart for the look-up table implementation of this algorithm.

In the first stage, the states of nine neighboring pixels are gathered together by a pixel stacker, and a following look-up table generates a conditional mark *M* for possible erasures. Table 9.3-1 lists all patterns, as indicated by the letter *S* in the table column, which will be conditionally

marked for erasure. In the second stage of the algorithm, the center pixel *X* and the conditional marks in a 3x3 neighborhood centered about *X* are examined to create an output pixel. The shrinking operation can be expressed logically as

$$G(j,k) = X \cap [M \cup P(M, M_0 \ldots M_7)] \qquad (9.3\text{-}1)$$

where $P(M, M_0 \ldots M_7)$ is an erasure inhibiting logical variable, as defined in Table 9.3-2. The first four patterns of the table prevent strokes of single pixel width from being totally erased. The remaining patterns inhibit erasure that would break object connectivity. There are a total of 157 inhibiting patterns. This two-stage process must be performed iteratively until there are no further erasures.

As an example, the square 2x 2 pixel object

1 1
1 1

results in the following intermediate array of conditional marks

M M
M M

The corner cluster pattern of Table 9.3-2 gives a hit only for the lower right corner mark. The resulting output is

0 0
0 1

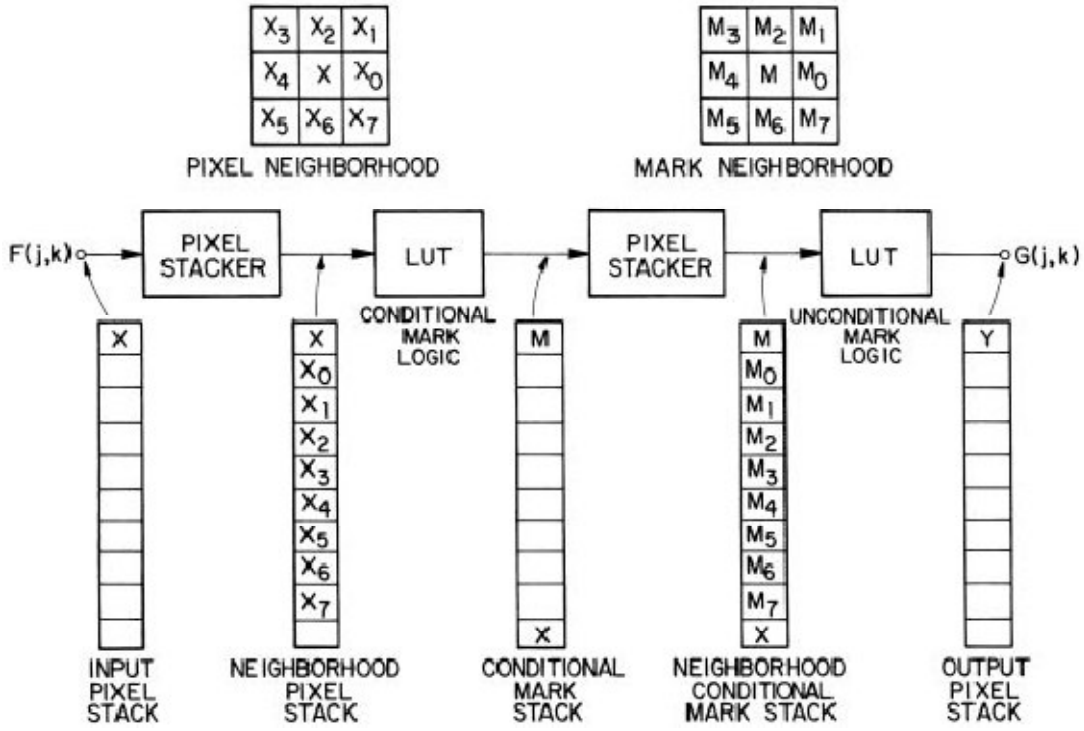


FIGURE 9.3-1. Look-up table flowchart for binary conditional mark operations.

9.3.2. Binary Image Thinning

The following is a definition of *thinning:*

Thin. Erase black pixels such that an object without holes erodes to a minimally connected stroke located equidistant from its nearest outer boundaries, and an object with holes erodes to a minimally connected ring midway between each hole and its nearest outer boundary.

TABLE 9.3-2. Shrink and Thin Unconditional Mark Patterns
[*P*(*M*, *M*0, *M*1, *M*2, *M*3, *M*4, *M*5, *M*6, *M*7) = 1 if hit] *a*

Pattern

Spur

0 0*M* *M*0 0
0 *M*0 0*M*0
0 0 0 0 0 0

Single 4-connection

0 0 0 0 0 0
0*M* 0 0*MM*
0 *M* 0 0 0 0

L Cluster (thin only)

0 0 *M* 0 *M M* *M M* 0 *M* 0 0 0 0 0 0 0 0 0 0 0 0
0 *MM* 0 *M* 0 0 *M* 0 *M M* 0 *M M*0 0*M*0 0*M*0 0*MM*
0 0 0 0 0 0 0 0 0 *M*0 0*M* *M*0 0 *MM* 0 0 *M*

4-Connected offset

0 *MM* *MM*0 0*M*0 00*M*
*MM*0 0*MM* 0*MM* 0*MM*
0 0 0 0 0 0 0 0 *M* 0 *M*0

Spur corner cluster

0 *A M* *MB*0 0 0*M* *M*0 0
0 *MB* *A M*0 *A M*0 0*MB*
*M*0 0 0 0*M* *MB*0 0*A M*

Corner cluster

MMD
MMD
DDD

Tee branch

*DM*0 0*MD* 0 0*D* *D*0 0 *DMD* 0 *M*0 0*M*0 *DMD*
MMM MMM *MMM* *MMM* *MM*0 *MM*0 0*MM* 0 *MM*
D 0 0 0 0 *D* 0 *MD* *DM*0 0*M*0 *DMD* *DMD* 0 *M*0

Vee branch

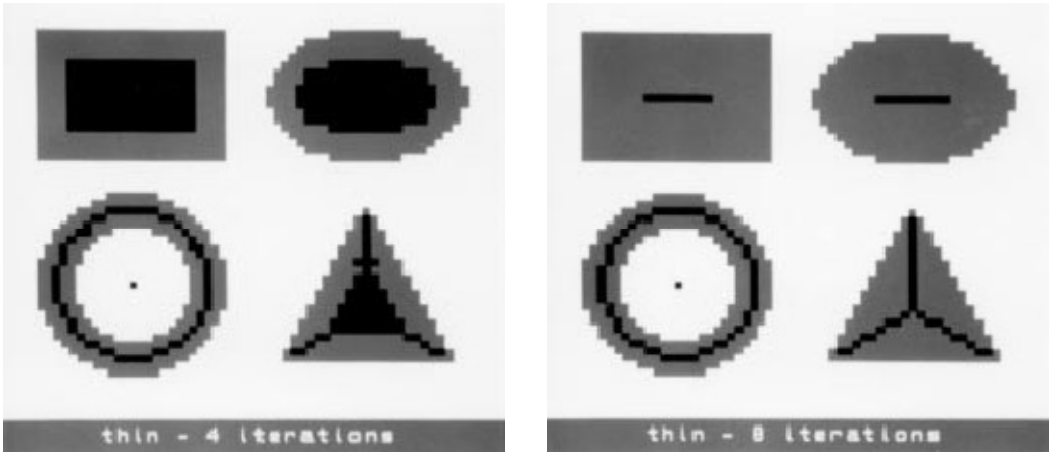
MDM *MD C* *C B A* *A DM*
DMD *DMB* *DMD* *B MD*
A B C *MDA* *MDM* *C DM*

Diagonal branch

*DM*0 0*MD* *D* 0*M* *M*0 *D*
0 *MM* *MM*0 *MM*0 0 *MM*
*M*0*D* *D*0 *M* 0 *MD* *DM*0

before after

Table 9.3-1 lists the conditional mark patterns, as indicated by the letter *T* in the table column, for thinning by the conditional mark algorithm of Figure 14.3-1. The shrink and thin unconditional patterns are identical, as shown in Table 9.3-2.
Figure 9.3-3 contains an example of the thinning of a binary image for four and eight iterations. Figure 9.3-4 provides an example of the thinning of an image of a printed circuit board in order to locate solder pads that have been deposited improperly and that do not have holes for component leads. The pads with holes erode to a minimally connected ring, while the pads without holes erode to a point.
Thinning can be applied to the background of an image containing several objects as a means of separating the objects. Figure 9.3-5 provides an example of the process. The original image appears in Figure 9.3-5*a*, and the backgroundreversed image is Figure 9.3-5*b*. Figure 9.3-5*c* shows the effect of thinning the background. The thinned strokes that separate the original objects are minimally connected, and therefore the background of the separating strokes is eight-connected throughout the image. This is an example of the connectivity ambiguity discussed in
Section 9.1. To resolve this ambiguity, a diagonal fill operation can be applied to the thinned strokes. The result, shown in Figure 9.3-5*d*, is called the *exothin* of the original image. The name derives from the exoskeleton, discussed in the following section.



(a) Four iterations (b) Eight iterations

FIGURE 9.3-3. Thinning of a binary image.

10.EDGE DETECTION

Changes or discontinuities in an image amplitude attribute such as luminance or tristimulus value are fundamentally important primitive characteristics of an image because they often provide an indication of the physical extent of objects within the image. Local discontinuities in image luminance from one level to another are called *luminance edges*. Global luminance discontinuities, called *luminance boundary segments*, are considered in Section 12.4. In this chapter the definition of a luminance edge is limited to image amplitude discontinuities between reasonably smooth regions. Discontinuity detection between textured regions is considered in Section 12.5. This chapter also considers edge detection in color images, as well as the detection of lines and spots within an image.

10.1. EDGE, LINE, AND SPOT MODELS

Figure 10.1-1*a* is a sketch of a continuous domain, one-dimensional *ramp edge* modeled as a ramp increase in image amplitude from a low to a high level, or vice versa. The edge is characterized by its height, slope angle, and horizontal coordinate of the slope midpoint. An edge exists if the edge height is greater than a specified value. An ideal edge detector should produce an edge indication localized to a single pixel located at the midpoint of the slope. If the slope angle of Figure 10.1-1*a* is 90°, the resultant edge is called a *step edge*, as shown in Figure 10.1-1*b*. In a digital imaging system, step edges usually exist only for artificially generated images such as test patterns and bilevel graphics data. Digital images, resulting from digitization of optical images of real scenes, generally do not possess step edges because the anti aliasing low-pass filtering prior to digitization reduces the edge slope in the digital image caused by any sudden luminance change in the scene. The one-dimensional profile of a *line* is shown in Figure 10.1-1*c*. In the limit, as the line width *w* approaches zero, the resultant amplitude discontinuity is called a *roof edge*.

Continuous domain, two-dimensional models of edges and lines assume that the amplitude discontinuity remains constant in a small neighborhood orthogonal to the edge or line profile. Figure 10.1-2*a* is a sketch of a two-dimensional edge. In addition to the edge parameters of a one-dimensional edge, the orientation of the edge slope with respect to a reference axis is also important. Figure 10.1-2*b* defines the edge orientation nomenclature for edges of an octagonally shaped object whose amplitude is higher than its background.

Figure 10.1-3 contains step and unit width ramp edge models in the discrete domain. The vertical ramp edge model in the figure contains a single transition pixel whose amplitude is at the midvalue of its neighbors. This edge model can be obtained by performing a pixel moving window average on the vertical step edge model. The figure also contains two versions of a diagonal ramp edge. The singlepixel transition model contains a single midvalue transition pixel between the regions of high and low amplitude; the smoothed transition model is generated by a 2x2 pixel moving window average of the diagonal step edge model. Figure 10.1-3 also presents models for a discrete step and ramp corner edge. The edge location for discrete step edges is usually marked at the higher-amplitude side of an edge transition.

For the single-pixel transition model and the smoothed transition vertical and corner edge models, the proper edge location is at the transition pixel. The smoothed transition diagonal ramp edge model has a pair of adjacent pixels in its transition zone. The edge is usually marked at the higher-amplitude pixel of the pair. In Figure

10.1-3 the edge pixels are italicized.

Discrete two-dimensional single-pixel line models are presented in Figure 10.1-4 for step lines and unit width ramp lines. The single-pixel transition model has a mid value transition pixel inserted between the high value of the line plateau and the low-value background. The smoothed transition model is obtained by performing a 2x2 pixel moving window average on the step line model.

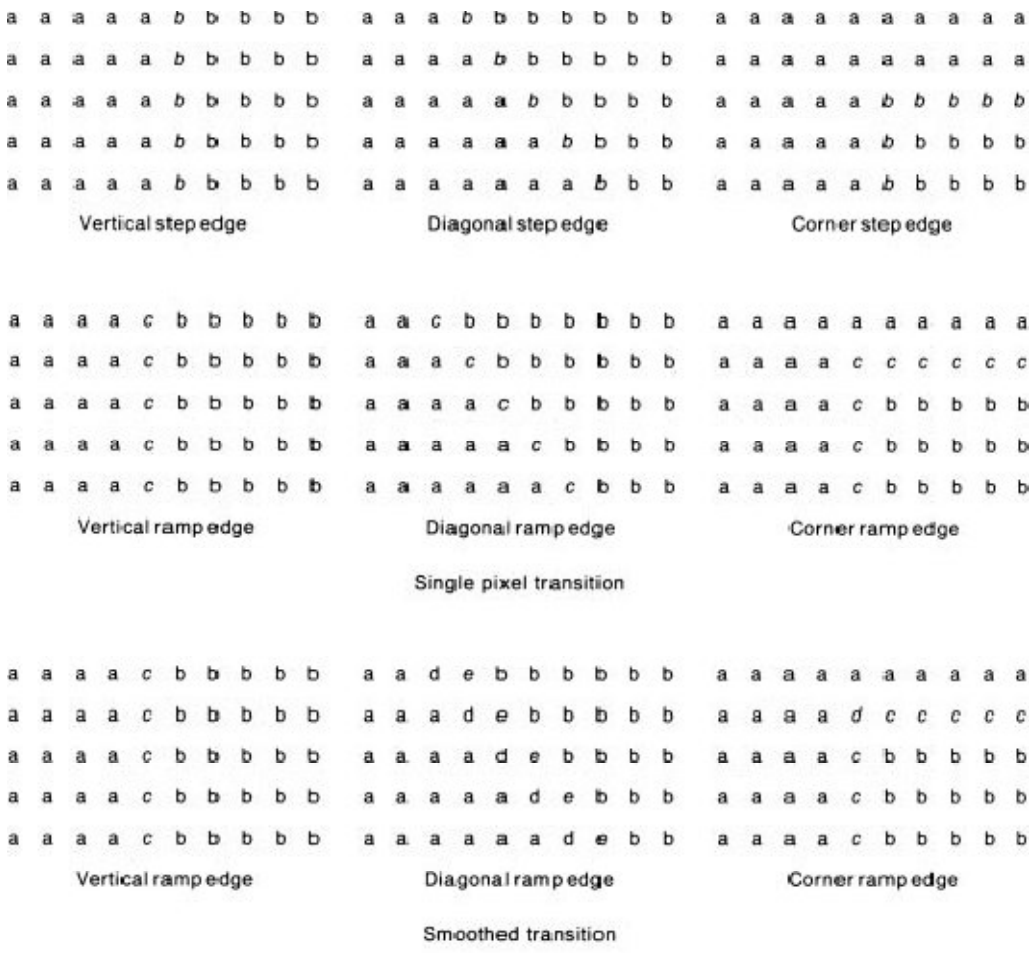


FIGURE 10.1-3. Two-dimensional, discrete domain edge models.

A *spot,* which can only be defined in two dimensions, consists of a plateau of high amplitude against a lower amplitude background, or vice versa. Figure 10.1-5 presents single-pixel spot models in the discrete domain.

There are two generic approaches to the detection of edges, lines, and spots in a luminance image: differential detection and model fitting. With the differential detection approach, as illustrated in Figure 10.1-6, spatial processing is performed on an original image F(j,k) to produce a differential image G(j,k) with accentuated spatial amplitude changes. Next, a differential detection operation is executed to determine the pixel locations of significant differentials. The second general approach to edge, line, or spot detection involves fitting of a local region of pixel values to a model of the edge, line, or spot, as represented in Figures 10.1-1 to 10.1-5. If the fit is sufficiently close, an edge, line, or spot is said to exist, and its assigned parameters are those of the appropriate model. A binary indicator map E(j,k) is often generated to indicate the position of edges, lines, or spots within an image. Typically, edge, line, and spot locations are specified by black pixels against a white background.

10.2. FIRST-ORDER DERIVATIVE EDGE DETECTION

There are two fundamental methods for generating first-order derivative edge gradients. One method involves generation of gradients in two orthogonal directions in an image; the second utilizes a set of directional derivatives.

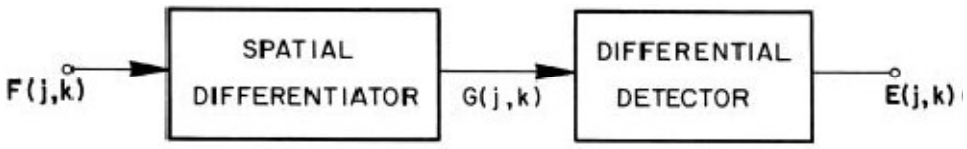


FIGURE 10.1-6. Differential edge, line, and spot detection.

10.2.1. Orthogonal Gradient Generation

An edge in a continuous domain edge segment F(x,y) such as the one depicted in Figure 10.1-2*a* can be detected by forming the continuous one-dimensional gradient G(j,k) along a line normal to the edge slope, which is at an angle θ with respect to the horizontal axis. If the gradient is sufficiently large (i.e., above some threshold value), an edge is deemed present. The gradient along the line normal to the edge slope can be computed in terms of the derivatives along orthogonal axes according to the following (1, p. 106)

$$G(x,y) = \frac{\delta F(x,y)}{\delta x}\cos\theta + \frac{\delta F(x,y)}{\delta y}\sin\theta \qquad (10.2\text{-}1)$$

Figure 10.2-1 describes the generation of an *edge gradient* in the discrete domain in terms of a *row gradient* and a *column gradient* . The spatial gradient amplitude is given by

$$G(j,k) = G_R[(j,k)]^2 + [G_C[(j,k)]^2]^{1/2} \qquad (10.2\text{-}2)$$

For computational efficiency, the gradient amplitude is sometimes approximated by the magnitude combination

$$G(j,k) = G_R(j,k) + G_C(j,k) \qquad (10.2\text{-}3)$$

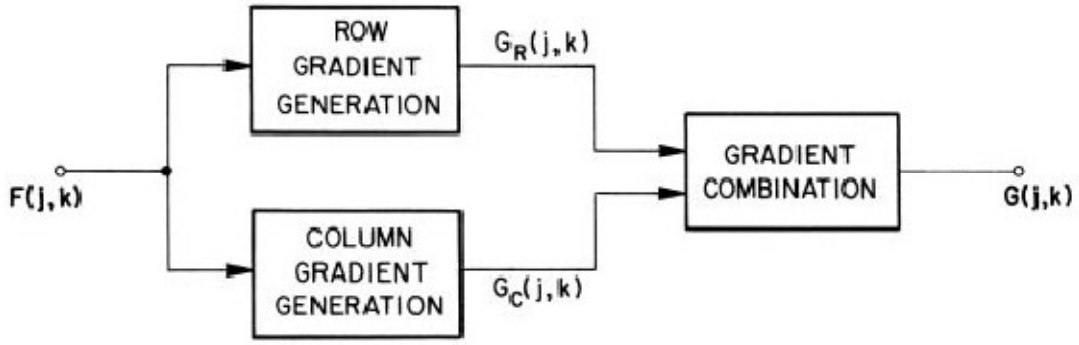
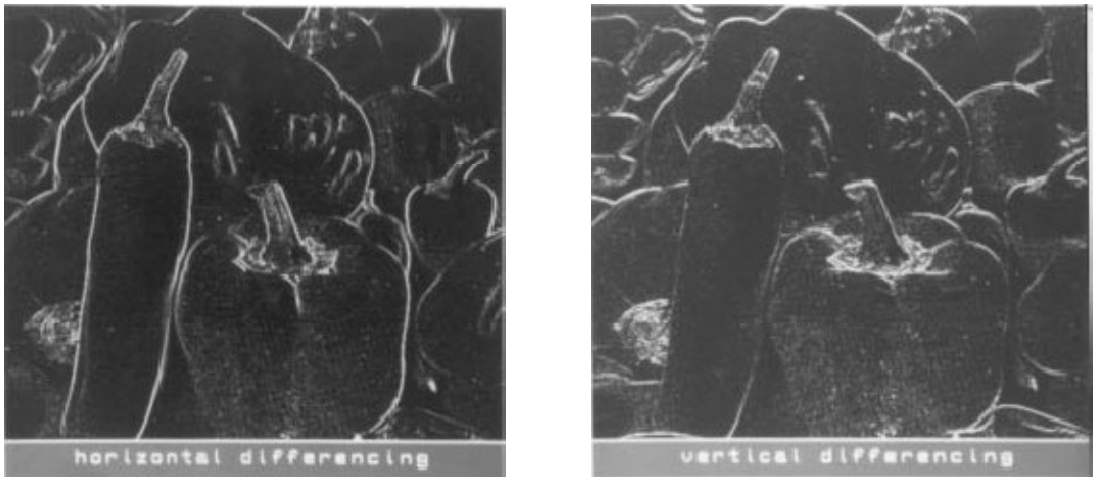


FIGURE 15.2-1. Orthogonal gradient generation.

(a) Original



(b) Horizontal magnitude (c) Vertical magnitude

FIGURE 10.2-2. Horizontal and vertical differencing gradients of the peppers_mon image

Diagonal edge gradients can be obtained by forming running differences of diagonal pairs of pixels. This is the basis of the Roberts (2) cross-difference operator, which is defined in magnitude form as

$$G(j, k) = G_1(j, k) + G_2(j, k) \quad (10.2\text{–}6\alpha)$$

and in square-root form as

$$G(j,k) = [(G1(j,k))^2 + (G2(j,k))^2]^{1/2} \quad (10.2\text{-}6b)$$

Where

$$G_1(j, k) = F(j, k) - F(j + 1, k + 1) \quad (10.2\text{–}6c)$$

$$G_2(j, k) = F(j, k + 1) - F(j + 1, k) \quad (10.2\text{-}6d)$$

Figure 10.2-3 presents the edge gradients of the peppers image for the *Roberts operators*. Visually, the objects in the image appear to be slightly better distinguished with the Roberts square-root gradient than with the magnitude gradient. In Section 10.5, a quantitative evaluation of edge detectors confirms the superiority of the square-root combination technique.

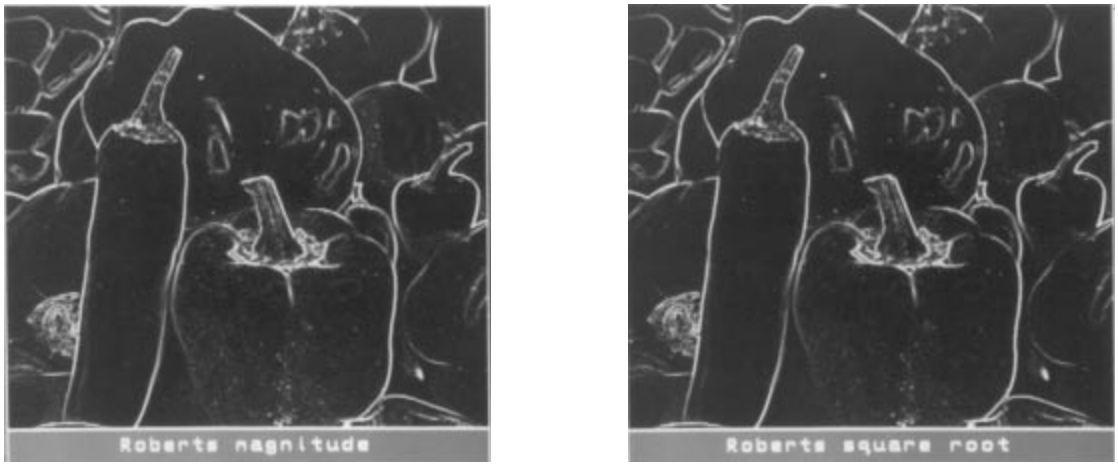
The pixel difference method of gradient generation can be modified to localize the edge center of the ramp edge model of Figure 15.1-3 by forming the pixel difference separated by a null value. The row and column gradients then become

$$G_R(j, k) = F(j,k+ 1) - F(j,k- 1) \quad (10.2\text{–}7\alpha)$$

$$G_C(j, k) = F(j- 1, k) - F(j + 1, k) \quad (10.2\text{–}7\beta)$$

The row gradient response for a vertical ramp edge model is then

0 0 h/2 h h/2 0 0



(a) Magnitude (b) Square root

FIGURE 10.2-3. Roberts gradients of the `peppers_mon` image.

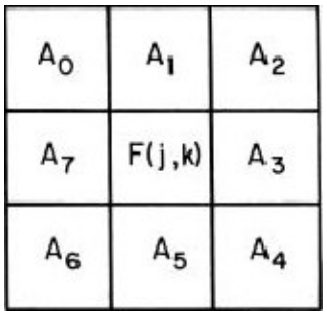


FIGURE 10.2-4. Numbering convention for 3 · 3 edge detection operators.

Although the ramp edge is properly localized, the separated pixel difference gradient generation method remains highly sensitive to small luminance fluctuations in the image. This problem can be alleviated by using two-dimensional gradient formation operators that perform differentiation in one coordinate direction and spatial averaging in the orthogonal direction simultaneously.

Prewitt has introduced a 3x3 pixel edge gradient operator described by the pixel numbering convention of Figure 10.2-4. The *Prewitt operator* square root edge gradient is defined as

$$G(j,k) = [G_R[(j,k)]^2 + G_C[(j,k)]^2]^{1/2} \quad (10.2\text{-}8a)$$

With

$$G_R(j,k) = 1/K + 2\,[(A_2 + KA_3 + A_4) - (A_0 + KA_7 + A_6)] \quad (10.2\text{–}8\beta)$$

$$G_C(j,k) = 1/K + 2\,[(A_0 + KA_1 + A_2) - (A_6 + KA_5 + A_4)] \quad (10.2\text{–}8\chi)$$

where $K = 1$. In this formulation, the row and column gradients are normalized to provide unit-gain positive and negative weighted averages about a separated edge position. The *Sobel*

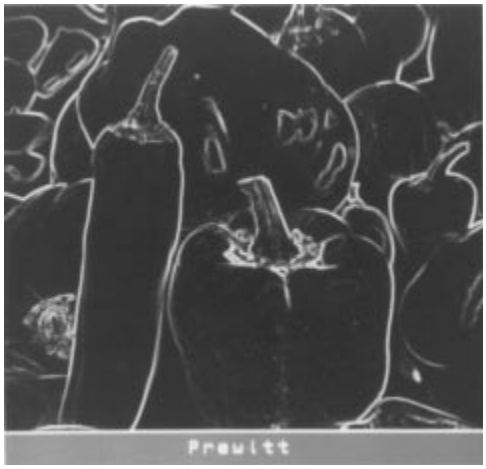
operator edge detector (3, p. 271) differs from the Prewitt edge detector in that the values of the north, south, east, and west pixels are doubled (i. e.,
K = 2). The motivation for this weighting is to give equal importance to each pixel in terms of its contribution to the spatial gradient. Frei and Chen (4) have proposed north, south, east, and west weightings by $k = \sqrt{2}$ so that the gradient is the same for horizontal, vertical, and diagonal edges. The edge gradient G(j,k) for these three operators along a row through the single pixel transition vertical ramp edge model of Figure 1.1-3 is

0 0 h/2 h h/2 0 0

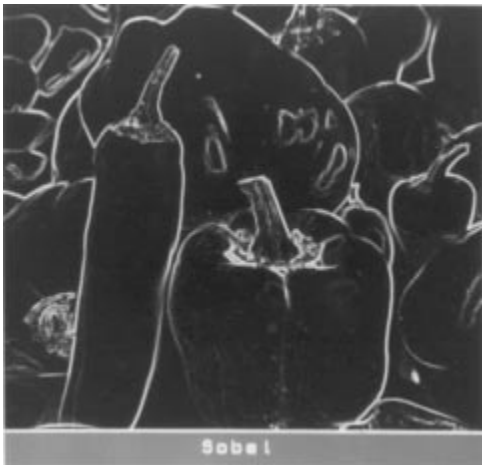
Along a row through the single transition pixel diagonal ramp edge model, the gradient is

0 $h2/(2+K)$ $h/2$ $2(1+K)h/2 + K$ $h/2$ $h/2(2+K)$ 0

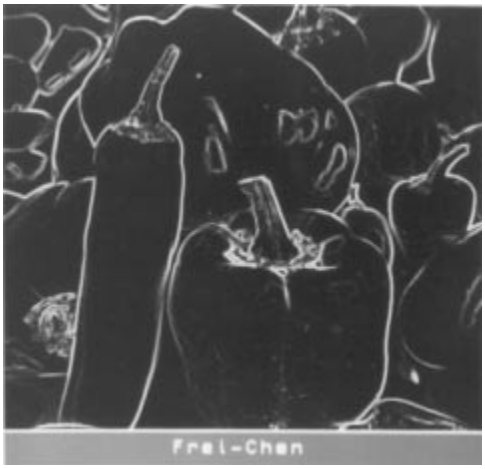
In the *Frei–Chen operator* with $k = \sqrt{2}$, the edge gradient is the same at the edge center for the single-pixel transition vertical and diagonal ramp edge models. The Prewitt gradient for a diagonal edge is 0.94 times that of a vertical edge. The corresponding factor for a Sobel edge detector is 1.06. Consequently, the Prewitt operator is more sensitive to horizontal and vertical edges than to diagonal edges; the reverse is true for the Sobel operator. The gradients along a row through the smoothed transition diagonal ramp edge model are different for vertical and diagonal edges for all three of the edge detectors. None of them are able to localize the edge to a single pixel.



(a) Prewitt



(b) Sobel



(b) Frei–Chen

FIGURE 10.2-5. Prewitt, Sobel , and Frei–Chen gradients of the `peppers_mon` image.

Figure 10.2-5 shows examples of the Prewitt, Sobel, and Frei–Chen gradients of the peppers image. The reason that these operators visually appear to better delineate object edges than the Roberts operator is attributable to their larger size, which provides averaging of small luminance fluctuations.

The row and column gradients for all the edge detectors mentioned previously in this subsection involve a linear combination of pixels within a small neighborhood. Consequently, the row and column gradients can be computed by the convolution relationships

$$G_R j\,k = F(j,k) \oplus H_R(j,k)$$

$$G_C(j,k) = F(j,k) \oplus H_C(j,k)$$

Where $H_R(j,k)$ and $H_C(j,k)$ are 3x3 row and column impulse response arrays, respectively, as defined in Figure 15.2-6. It should be noted that this specification of the gradient impulse response arrays takes into account the 180° rotation of an impulse response array inherent to the definition of convolution in Eq. 7.1-14.

A limitation common to the edge gradient generation operators previously defined is their inability to detect accurately edges in high-noise environments. This problem can be alleviated by properly extending the size of the neighborhood operators over which the differential gradients are computed. As an example, a Prewitt type operator has a row gradient impulse response of the form

$$H_R = 1/21 \begin{bmatrix} 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \end{bmatrix} \tag{15.2-11}$$

An operator of this type is called a *boxcar operator*. Figure 15.2-7 presents the boxcar gradient of a 7x7 array.

Operator	Row gradient	Column gradient
Pixel difference	$\begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Separated pixel difference	$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} 0 & -1 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$
Roberts	$\begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$	$\begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$
Prewitt	$\frac{1}{3}\begin{bmatrix} 1 & 0 & -1 \\ 1 & 0 & -1 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{3}\begin{bmatrix} -1 & -1 & -1 \\ 0 & 0 & 0 \\ 1 & 1 & 1 \end{bmatrix}$
Sobel	$\frac{1}{4}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{4}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$
Frei–Chen	$\frac{1}{2+\sqrt{2}}\begin{bmatrix} 1 & 0 & -1 \\ \sqrt{2} & 0 & -\sqrt{2} \\ 1 & 0 & -1 \end{bmatrix}$	$\frac{1}{2+\sqrt{2}}\begin{bmatrix} -1 & -\sqrt{2} & -1 \\ 0 & 0 & 0 \\ 1 & \sqrt{2} & 1 \end{bmatrix}$

FIGURE 15.2-6. Impulse response arrays for 3 x 3 orthogonal differential gradient edge operators.

Abdou (5) has suggested a *truncated pyramid operator* that gives a linearly decreasing weighting to pixels away from the center of an edge. The row gradient impulse response array for a 7x7 truncated pyramid operator is given by

$$H_R = 1/34 \begin{bmatrix} 1 & 1 & 1 & 0 & -1 & -1 & -1 \\ 1 & 2 & 2 & 0 & -2 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 3 & 0 & -3 & -2 & -1 \\ 1 & 2 & 2 & 0 & -2 & -2 & -1 \\ 1 & 1 & 1 & 0 & -1 & -1 & -1 \end{bmatrix} \quad (15.2\text{-}12)$$

Argyle (6) and Macleod (7,8) have proposed large neighborhood Gaussian-shaped weighting functions as a means of noise suppression. Let

$$g(x, s) = [2\pi s^2]^{-1/2} \exp\{-1 \Delta 2(x/s)^2\} \quad (10.2\text{–}12)$$

denote a continuous domain Gaussian function with standard deviation *s*. Utilizing this notation, the *Argyle operator* horizontal coordinate impulse response array can be expressed as a sampled version of the continuous domain impulse response

$$HR(j,k) = f(x) = \begin{cases} 2g(x,s)g(y,t), & x < 0 \\ -2g(x,s)g(y,t), & x \geq 0 \end{cases} \quad (10.2\text{-}13)$$

where *s* and *t* are spread parameters. The vertical impulse response function can be expressed similarly. The *Macleod operator* horizontal gradient impulse response function is given by

$$H_R(j,k) = [g(x+s,s) - g(x-s,s)]g(y,t) \quad (10.2\text{–}14)$$

The Argyle and Macleod operators, unlike the boxcar operator, give decreasing importance to pixels far removed from the center of the neighborhood. Figure 15.2-7 provides examples of the Argyle and Macleod gradients.

Extended-size differential gradient operators can be considered to be compound operators in which a smoothing operation is performed on a noisy image followed by a differentiation operation. The compound gradient impulse response can be written as

$$H(j,k) = H_G(j,k) \otimes H_S(j,k) \quad (10.2\text{–}15)$$

where $H_G(j,k)$is one of the gradient impulse response operators of Figure 15.2-6 and $H_G(j,k)$is a low-pass filter impulse response. For example, if $H_S(j,k)$is the 3x3 Prewitt row gradient operator and $H_S(j,k)$, for all , is a 3x3 uniform smoothing operator, the resultant 5x5 row gradient operator, after normalization to unit positive and negative gain, becomes

$$H_R = 1/34 \begin{bmatrix} 1 & 1 & 0 & -1 & -1 \\ 2 & 2 & 0 & -2 & -2 \\ & & & & \\ 3 & 3 & 0 & -3 & -3 \\ 2 & 2 & 0 & -2 & -2 \\ 1 & 1 & 0 & -1 & -1 \end{bmatrix} \quad (10.2\text{-}16)$$

The decomposition of Eq. 15.2-16 applies in both directions. By applying the SVD/ SGK decomposition of Section 9.6, it is possible, for example, to decompose a 5x5 boxcar operator into the sequential convolution of a 3x3 smoothing kernel and a 3x3 differentiating kernel.

A well-known example of a compound gradient operator is the *first derivative of Gaussian* (FDOG) *operator*, in which Gaussian-shaped smoothing is followed by differentiation (9). The FDOG continuous domain horizontal impulse response is

$$H_R(j,k) = -\frac{\partial[g(x,s)g(y,t)]}{\partial x} \quad (10.2\text{-}17)$$

which upon differentiation yields

$$H_R(j,k) = \frac{-xg(x,s)g(y,t)}{s2} \quad (10.2\text{-}18)$$

Figure 10.2-7 presents an example of the FDOG gradient. All of the differential edge enhancement operators presented previously in this subsection have been derived heuristically.

Canny (9) has taken an analytic approach to the design of such operators. Canny's development is based on a onedimensional continuous domain model of a step edge of amplitude *hE* plus additive white Gaussian noise with standard deviation . It is assumed that edge detection is performed by convolving a one-dimensional continuous domain noisy edge signal f(x,y) with an antisymmetric impulse response function , which is of zero amplitude outside the range [-W,W]. An edge is marked at the local maximum of the convolved gradient f(x,y) ⊗ h(x,y) . The *Canny operator* impulse response h(x,y) is chosen to satisfy the following three criteria.

1. *Good detection*. The amplitude signal-to-noise ratio (SNR) of the gradient is maximized to obtain a low probability of failure to mark real edge points and a low probability of falsely marking nonedge points. The SNR for the model is

$$\text{SNR} = \frac{h_E S(h)}{\sigma n} \tag{10.2-19}$$

With

$$\text{S(h)} = \frac{\int_{-W}^{0} h(x)dx}{\int_{-W}^{W} h(x)^2 dx} \tag{10.2-20}$$

2. *Good localization*. Edge points marked by the operator should be as close to the center of the edge as possible. The localization factor is defined as

$$\text{LOC} = \frac{h_E L(h)}{\sigma n} \tag{10.2-21a}$$

$$\text{L(h)} = \frac{h(0)}{\int_{-W}^{W} h(x)^2 dx} \tag{10.2-21b}$$

3. *Single response*. There should be only a single response to a true edge. Thedistance between peaks of the gradient when only noise is present, denoted as *xm*, is set to some fraction *k* of the operator width factor *W*. Thus

$$X_m = kW \tag{10.2-22}$$

Canny has combined these three criteria by maximizing the product subject to the constraint of Eq. 10.2-21. Because of the complexity of the formulation, no analytic solution has been found, but a variational approach has been developed. Figure 10.2-8 contains plots of the Canny impulse response functions in terms of *xm*.

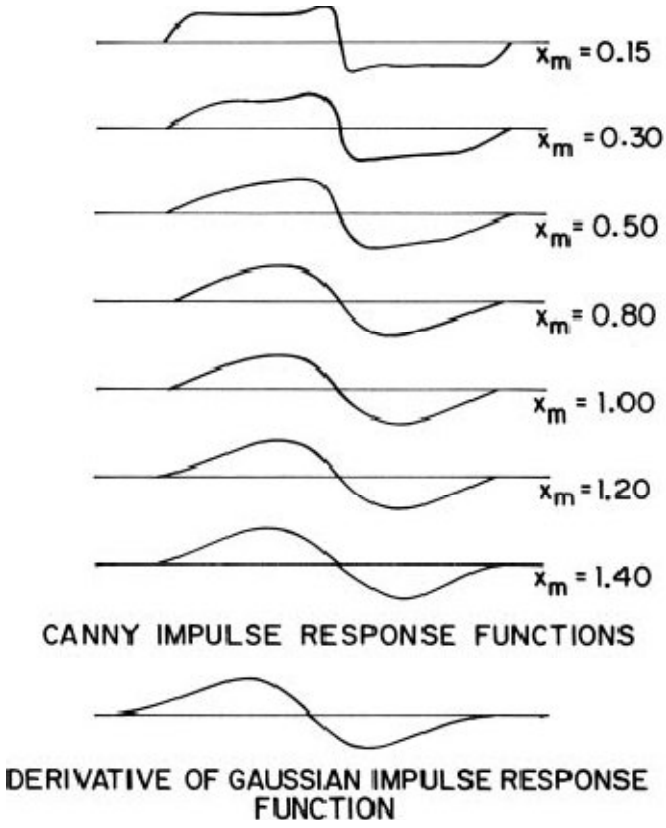


FIGURE 10.2-8. Comparison of Canny and first derivative of Gaussian impulse response functions

As noted from the figure, for low values of *xm*, the Canny function resembles a boxcar function, while for *xm* large, the Canny function is closely approximated by a FDOG impulse response function.

Discrete domain versions of the large operators defined in the continuous domain can be obtained by sampling their continuous impulse response functions over some window. The window size should be chosen sufficiently large that truncation of the impulse response function does not cause high-frequency artifacts. Demigny and Kamie (10) have developed a discrete version of Canny's criteria, which lead to the computation of discrete domain edge detector impulse response arrays.

10.2.2. Edge Template Gradient Generation

With the orthogonal differential edge enhancement techniques discussed previously, edge gradients are computed in two orthogonal directions, usually along rows and columns, and then the edge direction is inferred by computing the vector sum of the gradients. Another approach is to compute gradients in a large number of directions by convolution of an image with a set of template gradient impulse response arrays. The edge template gradient is defined as

$$G(j,k) = \text{MAX}\{G_1(j,k),\ldots\ldots, G_m(j,k),\ldots\ldots\ldots G_M(j,k)\} \qquad (10.2\text{–}22\alpha)$$

Where

11.IMAGE SEGMENTATION

Segmentation of an image entails the division or separation of the image into regions of similar attribute. The most basic attribute for segmentation is image luminance amplitude for a monochrome image and color components for a color image. Image edges and texture are also useful attributes for segmentation.
The definition of segmentation adopted in this chapter is deliberately restrictive; no contextual information is utilized in the segmentation. Furthermore, segmentation does not involve classifying each segment. The segmenter only subdivides an image; it does not attempt to recognize the individual segments or their relationships to one another.
There is no theory of image segmentation. As a consequence, no single standard method of image segmentation has emerged. Rather, there are a collection of ad hoc methods that have received some degree of popularity. Because the methods are ad hoc, it would be useful to have some means of assessing their performance. Haralick and Shapiro (1) have established the following qualitative guideline for a good image segmentation: "Regions of an image segmentation should be uniform and homogeneous with respect to some characteristic such as gray tone or texture.
Region interiors should be simple and without many small holes. Adjacent regions of a segmentation should have significantly different values with respect to the characteristic on which they are uniform. Boundaries of each segment should be simple, not ragged, and must be spatially accurate." Unfortunately, no quantitative image segmentation performance metric has been developed.
Several generic methods of image segmentation are described in the following sections. Because of their complexity, it is not feasible to describe all the details of the various algorithms. Surveys of image segmentation methods are given in References 1 to 6.

11.1. AMPLITUDE SEGMENTATION METHODS

This section considers several image segmentation methods based on the thresholding of luminance or color components of an image. An amplitude projection segmentation technique is also discussed.

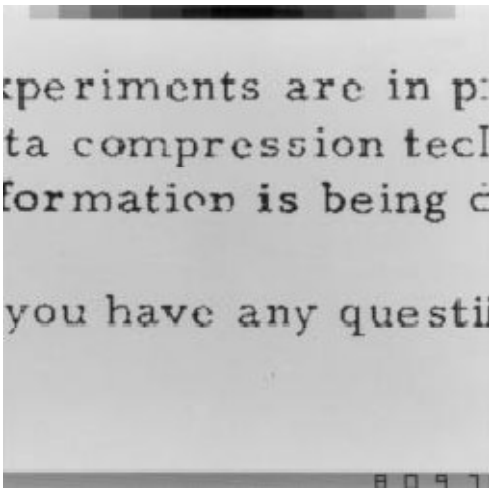
11.1.1. Bilevel Luminance Thresholding

Many images can be characterized as containing some object of interest of reasonably uniform brightness placed against a background of differing brightness. Typical examples include handwritten and typewritten text, microscope biomedical samples, and airplanes on a runway.

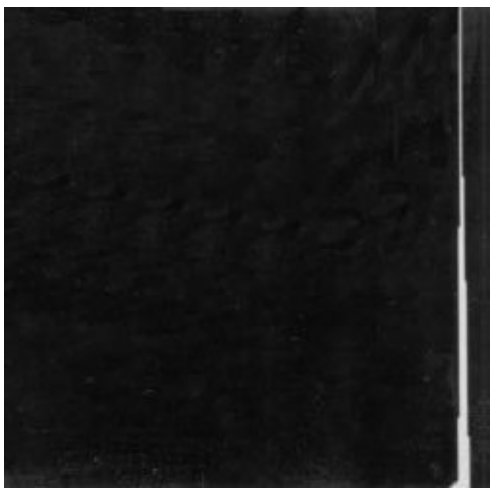
For such images, luminance is a distinguishing feature that can be utilized to segment the object from its background. If an object of interest is white against a black background, or vice versa, it is a trivial task to set a midgray threshold to segment the object from the background. Practical problems occur, however, when the observed image is subject to noise and when both the object and background assume some broad range of gray scales. Another frequent difficulty is that the background may be nonuniform. Figure 17.1-1*a* shows a digitized typewritten text consisting of dark letters against a lighter background. A gray scale histogram of the text is presented in Figure 17.1-1*b*. The expected bimodality of the histogram is masked by the relatively large percentage of background pixels. Figure 17.1-1*c* to *e* are threshold displays in which all pixels brighter than the threshold are mapped to unity display luminance and all the remaining pixels below the threshold are mapped to the zero level of display luminance. The photographs illustrate a common problem associated with image thresholding. If the threshold is set too low, portions of the letters are deleted (the stem of the letter "p" is fragmented). Conversely, if the threshold is set too high, object artifacts result (the loop of the letter "e" is filled in).

Several analytic approaches to the setting of a luminance threshold have been proposed (7,8). One method is to set the gray scale threshold at a level such that the cumulative gray scale count matches an a priori assumption of the gray scale probability distribution (9). For example, it may be known that black characters cover 25% of the area of a typewritten page. Thus, the threshold level on the image might be set such that the quartile of pixels with the lowest luminance are judged to be black. Another approach to luminance threshold selection is to set the threshold at the minimum point of the histogram between its bimodal peaks (10). Determination of the minimum is often difficult because of the jaggedness of the histogram. A solution to this problem is to fit the histogram values between the peaks with some analytic function and then obtain its minimum by differentiation. For example, let *y* and *x* represent the histogram ordinate and abscissa, respectively. Then the quadratic curve

$$y = ax^2 + bx + c \qquad (11.1\text{-}1)$$

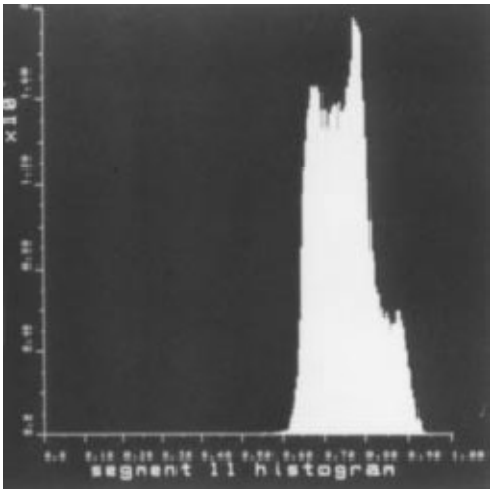


(a) Gray scale text



(b) Histogram

(c) Segment 11



(d) Segment 11 histogram

FIGURE 11.1-4. Multilevel luminance thresholding image segmentation of the `peppers_mon` image; second-level segmentation, 1 branch.

stage of separation under manual surveillance. Ohlander's segmentation technique using multidimensional thresholding aided by texture discrimination has proved quite effective in simulation tests. However, a large part of the segmentation control has been performed by a human operator; human judgment, predicated on trial
threshold setting results, is required for guidance.

In Ohlander's segmentation method, the nine property values are obviously interdependent. The *YIQ* and intensity components are linear combinations of *RGB*; the hue and saturation measurements are nonlinear functions of *RGB*. This observation raises several questions. What types of linear and nonlinear transformations of *RGB* are best for segmentation? Ohta et al. (19) suggest an approximation to the spectral Karhunen–Loeve transform. How many property values should be used? What is the best form of property thresholding? Perhaps answers to these last two questions may be forthcoming from a study of clustering techniques in pattern recognition (20).

Property value histograms are really the marginal histograms of a joint histogram of property values. Clustering methods can be utilized to specify multidimensional decision boundaries for segmentation. This approach permits utilization of all the property values for segmentation and inherently recognizes their respective cross correlation. The following section discusses clustering methods of image segmentation.

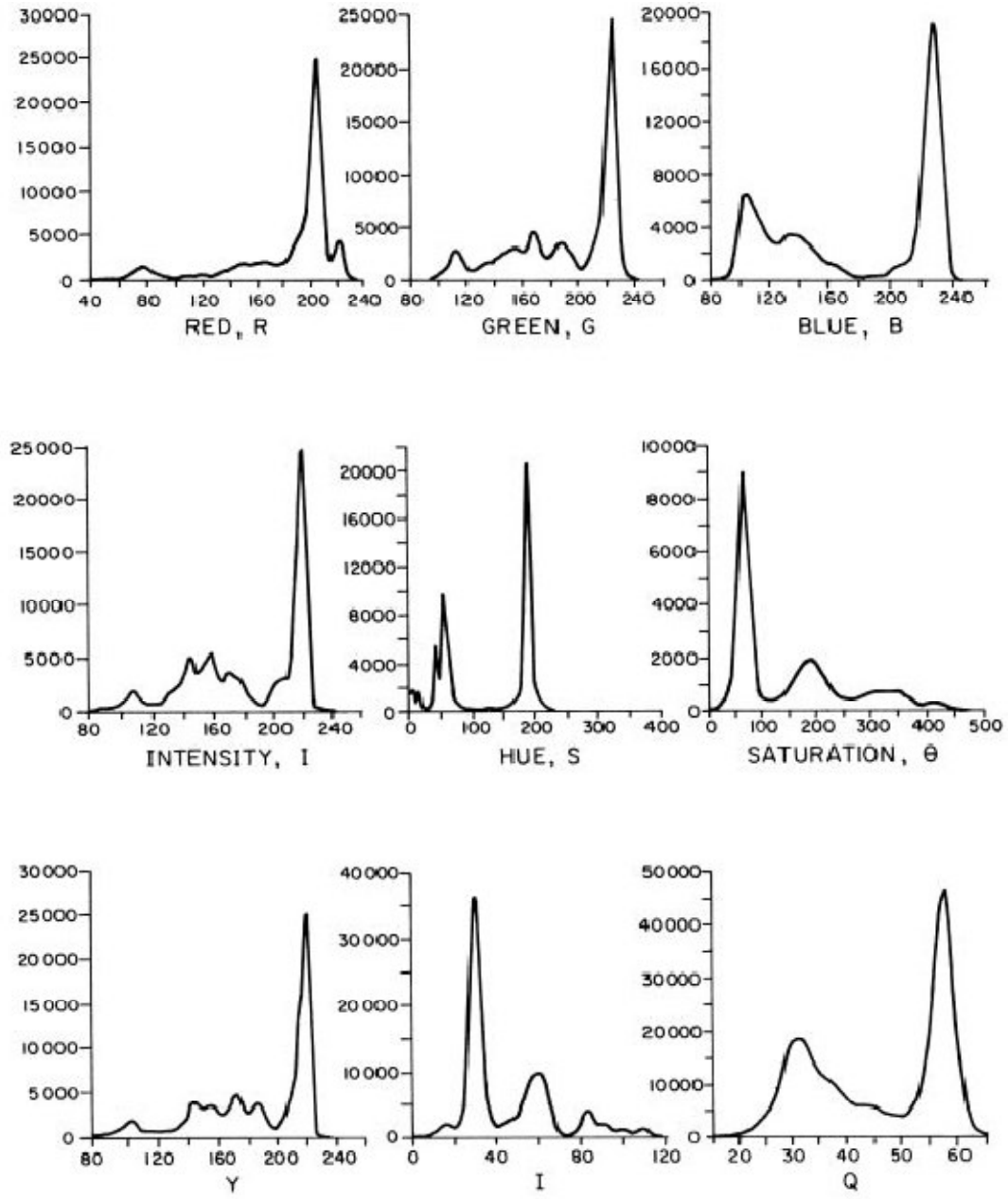
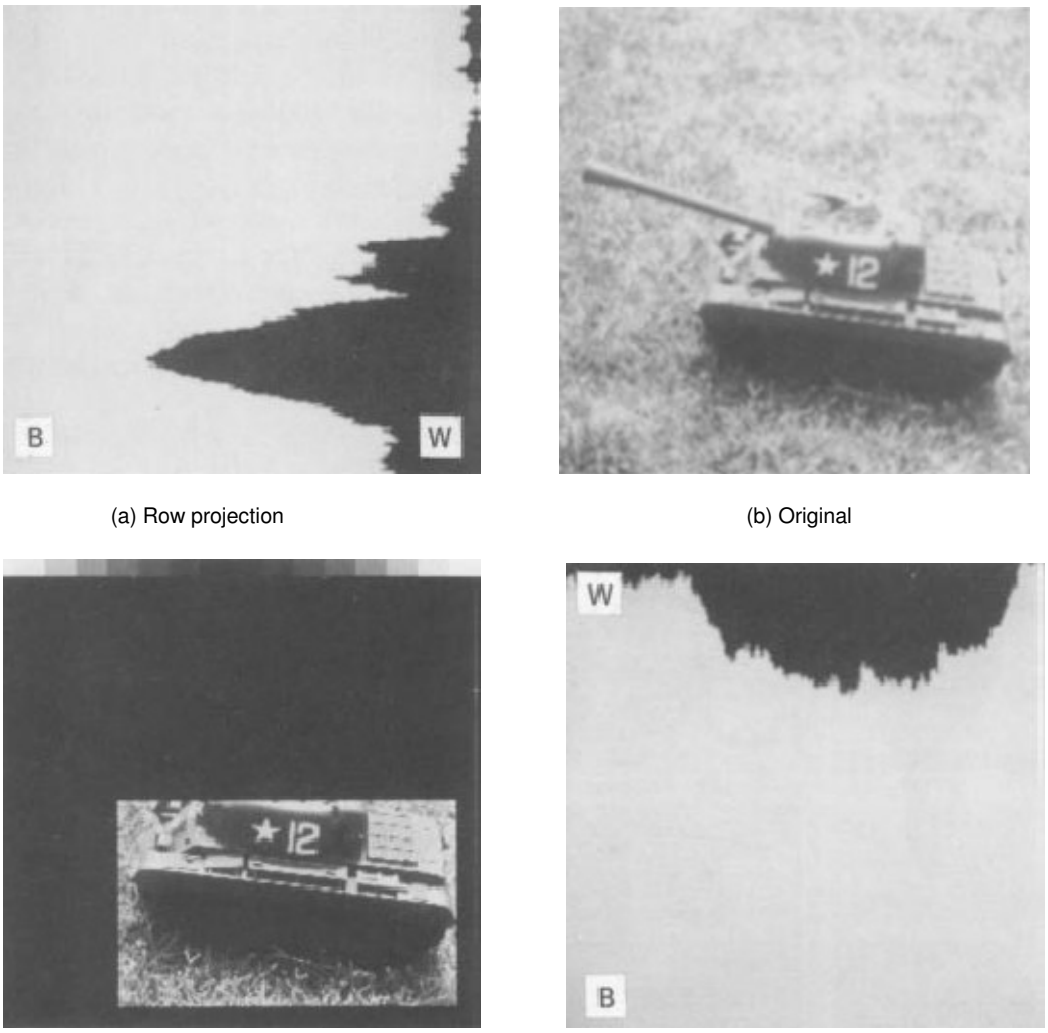


FIGURE 17.1-5. Typical property histograms for color image segmentation.

11.1.2. Amplitude Projection

Image segments can sometimes be effectively isolated by forming the average *amplitude projections* of an image along its rows and columns (21,22).
Figure 17.1-6 illustrates an application of gray scale projection segmentation of an image. The rectangularly shaped segment can be further delimited by taking projections over oblique angles.



(a) Row projection

(b) Original

(c) Segmentation

(d) Column projection

FIGURE 17.1-6. Gray scale projection image segmentation of a toy tank image.

11.2. CLUSTERING SEGMENTATION METHODS

One of the earliest examples of image segmentation, by Haralick and Kelly (23) using data clustering, was the subdivision of multispectral aerial images of agricultural land into regions containing the same type of land cover. The clustering segmentation concept is simple; however, it is usually computationally intensive.

Consider a vector of measurements at each pixel coordinate (*j*, *k*) in an image. The measurements could be point multispectral values, point color components, and derived color components, as in the Ohlander approach described previously, or they could be neighborhood

feature measurements such as the moving window mean, standard deviation, and mode, as discussed in Section 16.2. If the measurement set is to be effective for image segmentation, data collected at various pixels within a segment of common attribute should be similar. That is, the data should be tightly clustered in an *N*-dimensional measurement space. If this condition holds, the segmenter design task becomes one of subdividing the *N*-dimensional measurement space into mutually exclusive compartments, each of which envelopes typical data clusters for each image segment. Figure 17.2-1 illustrates the concept for two features. In the segmentation process, if a measurement vector for a pixel falls within a measurement space compartment, the pixel is assigned the segment name or label of that compartment.

Coleman and Andrews (24) have developed a robust and relatively efficient image segmentation clustering algorithm. Figure 17.2-2 is a flowchart that describes a simplified version of the algorithm for segmentation of monochrome images. The first stage of the algorithm involves feature computation. In one set of experiments,

Coleman and Andrews used 12 mode measurements in square windows of size 1, 3, 7, and 15 pixels. The next step in the algorithm is the clustering stage, in which the optimum number of clusters is determined along with the feature space center of each cluster. In the segmenter, a given feature vector is assigned to its closest cluster center.

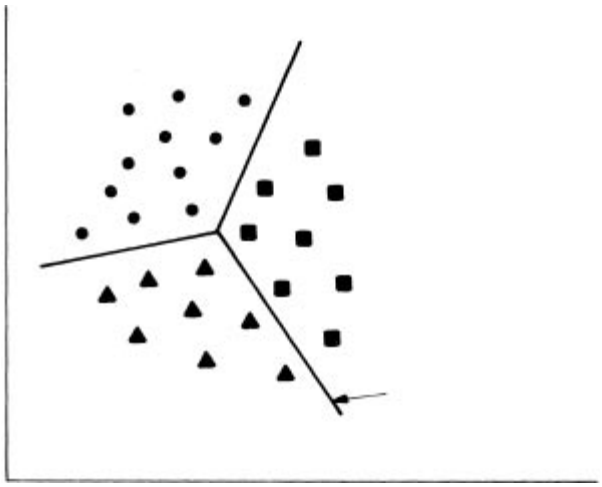


FIGURE 17.2-1. Data clustering for two feature measurements.

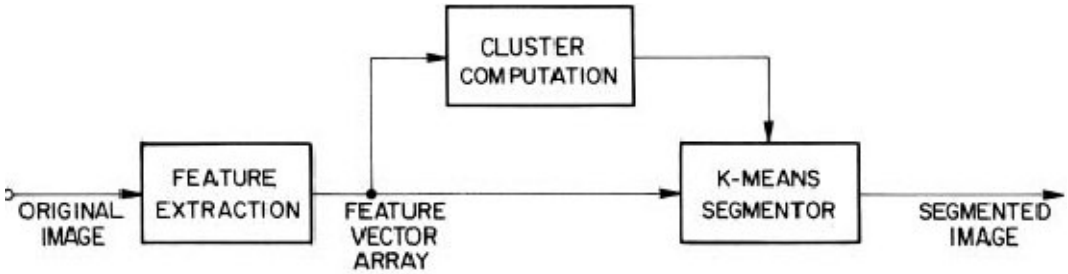


FIGURE 17.2-2. Simplified version of Coleman–Andrews clustering image segmentation method.

11.3. REGION SEGMENTATION METHODS

The amplitude and clustering methods described in the preceding sections are based on point properties of an image. The logical extension, as first suggested by Muerle and Allen (25), is to utilize spatial properties of an image for segmentation.

11.3.1. Region Growing

Region growing is one of the conceptually simplest approaches to image segmentation; neighboring pixels of similar amplitude are grouped together to form a segmented region. However, in practice, constraints, some of which are reasonably complex, must be placed on the growth pattern to achieve acceptable results.

Brice and Fenema (26) have developed a region-growing method based on a set of simple growth rules. In the first stage of the process, pairs of quantized pixels are combined together in groups called *atomic regions* if they are of the same amplitude and are four-connected. Two heuristic rules are next invoked to dissolve weak boundaries between atomic boundaries. Referring to Figure 17.3-1, let $R1$ and $R2$ be two adjacent regions with perimeters $P1$ and $P2$, respectively, which have previously been merged. After the initial stages of region growing, a region may contain previously merged subregions of different amplitude values. Also, let C denote the length of the common boundary and let D represent the length of that portion of C for which the amplitude difference Y across the boundary is smaller than a significance factor . The regions $R1$ and $R2$ are then merged if

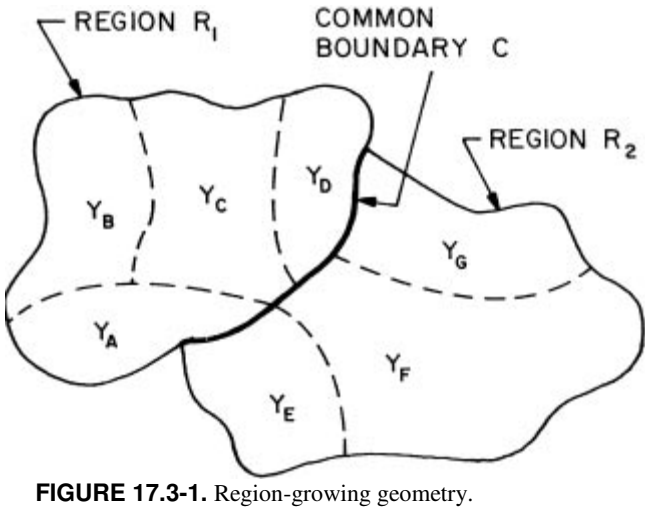


FIGURE 17.3-1. Region-growing geometry.

where is a constant typically set at . This heuristic prevents merger of adjacent regions of the same approximate size, but permits smaller regions to be absorbed into larger regions. The second rule merges weak common boundaries remaining after application of the first rule. Adjacent regions are merged if

$$D/C \geq \varepsilon_3 \qquad (11.3\text{-}2)$$

where is a constant set at about . Application of only the second rule tends to overmerge regions.

The Brice and Fenema region growing method provides reasonably accurate segmentation of simple scenes with few objects and little texture (26, 27) but does not perform well on more complex scenes. Yakimovsky (28) has attempted to improve the region-growing concept by establishing merging constraints based on estimated Bayesian probability densities of feature measurements of each region.

11.3.2. Split and Merge

Split and merge image segmentation techniques (29) are based on a quad tree data representation whereby a square image segment is broken (split) into four quadrants if the original image segment is nonuniform in attribute. If four neighboring squares are found to be uniform, they are replaced (merge) by a single square composed of the four adjacent squares.

In principle, the split and merge process could start at the full image level and initiate split operations. This approach tends to be computationally intensive. Conversely, beginning at the individual pixel level and making initial merges has the drawback that region uniformity measures are limited at the single pixel level. Initializing the split and merge process at an intermediate level enables the use of more powerful uniformity tests without excessive computation.

The simplest uniformity measure is to compute the difference between the largest and smallest pixels of a segment. Fukada (30) has proposed the segment variance as a uniformity measure. Chen and Pavlidis (31) suggest more complex statistical measures of uniformity. The basic split and merge process tends to produce rather blocky segments because of the rule that square blocks are either split or merged. Horowitz and Pavlidis (32) have proposed a modification of the basic process whereby adjacent pairs of regions are merged if they are sufficiently uniform.

11.3.3. Watershed

Topographic and hydrology concepts have proved useful in the development of region segmentation methods (33–36). In this context, a monochrome image is considered to be an altitude surface in which high-amplitude pixels correspond to ridge points, and low-amplitude pixels correspond to valley points. If a drop of water were to fall on any point of the altitude surface, it would move to a lower altitude until it reached a local altitude minimum. The accumulation of water in the vicinity of a local minimum is called a *catchment basin*. All points that drain into a common catchment basin are part of the same *watershed*. A *valley* is a region that is surrounded by a ridge. A *ridge* is the loci of maximum gradient of the altitude surface.

There are two basic algorithmic approaches to the computation of the watershed of an image: rainfall and flooding.

In the *rainfall* approach, local minima are found throughout the image. Each local minima is given a unique tag. Adjacent local minima are combined with a unique tag. Next, a conceptual water drop is placed at each untagged pixel. The drop moves to its lower-amplitude neighbor

until it reaches a tagged pixel, at which time it assumes the tag value. Figure 11.3-2 illustrates a section of a digital image encompassing a watershed in which the local minimum pixel is black and the dashed line indicates the path of a water drop to the local minimum.
In the *flooding* approach, conceptual single pixel holes are pierced at each local minima, and the amplitude surface is lowered into a large body of water. The water enters the holes and proceeds to fill each catchment basin. If a basin is about to overflow, a conceptual dam is built on its surrounding ridge line to a height equal to the highest- altitude ridge point. Figure 11.3-3 shows a profile of the filling process of a catchment basin (37). Figure 11.3-4 is an example of watershed segmentation provided by Moga and Gabbouj (38).

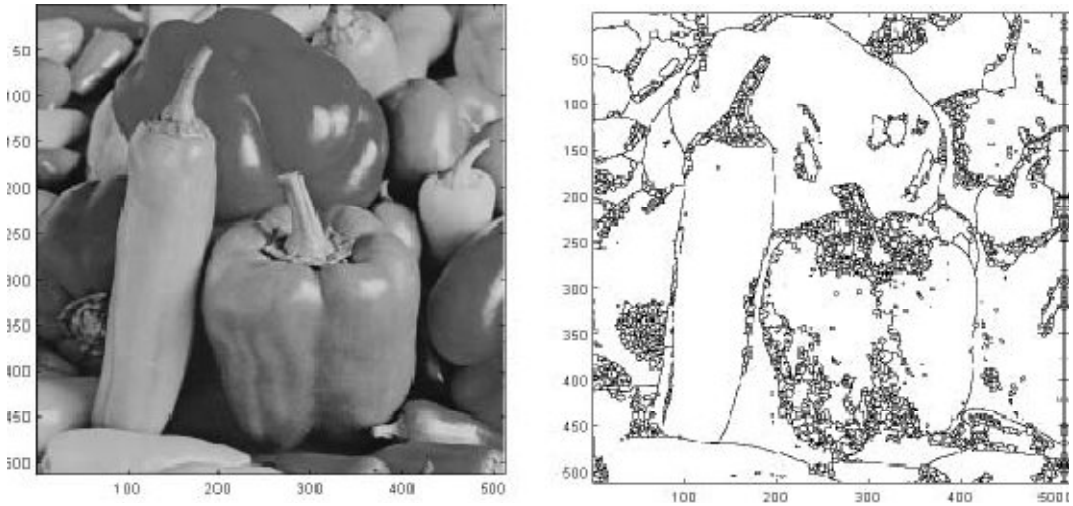


FIGURE 11.3-4. Watershed image segmentation of the peppers_mon image. Courtesy of Alina N. Moga and M. Gabbouj, Tampere University of Technology, Finland.

(a) Original

(b) Edge map (c) Thinned edge map

FIGURE 17.4-1. Boundary detection image segmentation of the projectile image.

Simple watershed algorithms tend to produce results that are oversegmented (39). Najman and Schmitt (37) have applied morphological methods in their watershed algorithm to reduce over segmentation. Wright and Acton (40) have performed watershed segmentation on a pyramid of different spatial resolutions to avoid oversegmentation.

11.4. BOUNDARY DETECTION

It is possible to segment an image into regions of common attribute by detecting the boundary of each region for which there is a significant change in attribute across the boundary. Boundary detection can be accomplished by means of edge detection as described in Chapter 10. Figure 11.4-1 illustrates the segmentation of a projectile from its background. In this example a derivative of Gaussian edge detector is used to generate the edge map of Figure 11.4-1*b*. Morphological thinning of this edge map results in Figure 17.4-1*c*. The resulting boundary appears visually to be correct when overlaid on the original image. If an image is noisy or if its region attributes differ by only a small amount between regions, a detected boundary may often be broken. *Edge linking* techniques can be employed to bridge short gaps in such a region boundary.

11.4.1. Curve-Fitting Edge Linking

In some instances, edge map points of a broken segment boundary can be linked together to form a closed contour by curve-fitting methods. If a priori information is available as to the expected shape of a region in an image (e.g., a rectangle or a circle), the fit may be made

directly to that closed contour. For more complexshaped regions, as illustrated in Figure 11.4-2, it is usually necessary to break up the supposed closed contour into chains with broken links. One such chain, shown in Figure 11.4-2 starting at point *A* and ending at point *B*, contains a single broken link. Classical curve-fitting methods (29) such as *Bezier polynomial* or *spline fitting* can
be used to fit the broken chain.

In their book, Duda and Hart (41) credit Forsen as being the developer of a simple piecewise linear curve-fitting procedure called the *iterative endpoint fit*. In the first stage of the algorithm, illustrated in Figure 11.4-3, data endpoints *A* and *B* are connected by a straight line. The point of greatest departure from the straight-line (point *C*) is examined. If the separation of this point is too large, the point becomes an anchor point for two straight-line segments (*A* to *C* and *C* to *B*). The procedure then continues until the data points are well fitted by line segments. The principal advantage of the algorithm is its simplicity; its disadvantage is error caused by incorrect data points. Ramer (42) has used a technique similar to the iterated endpoint procedure to determine a polynomial approximation to an arbitrary-shaped closed curve. Pavlidis and Horowitz (43) have developed related algorithms for polygonal curve fitting. The curve-fitting approach is reasonably effective for simply structured objects. Difficulties occur when an image contains many overlapping objects and its corresponding edge map contains branch structures.

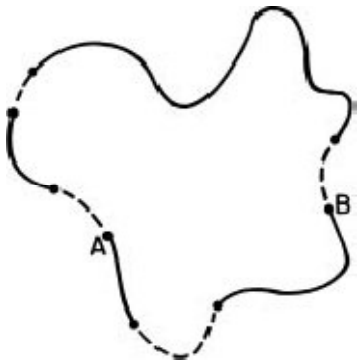


FIGURE 11.4-2. Region boundary with missing links indicated by dashed lines.

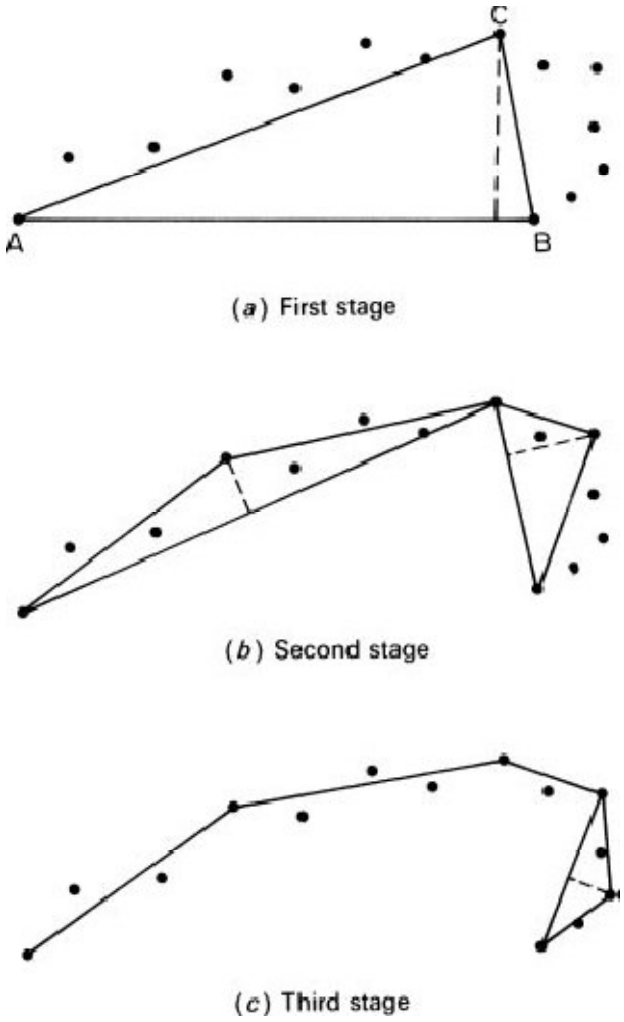
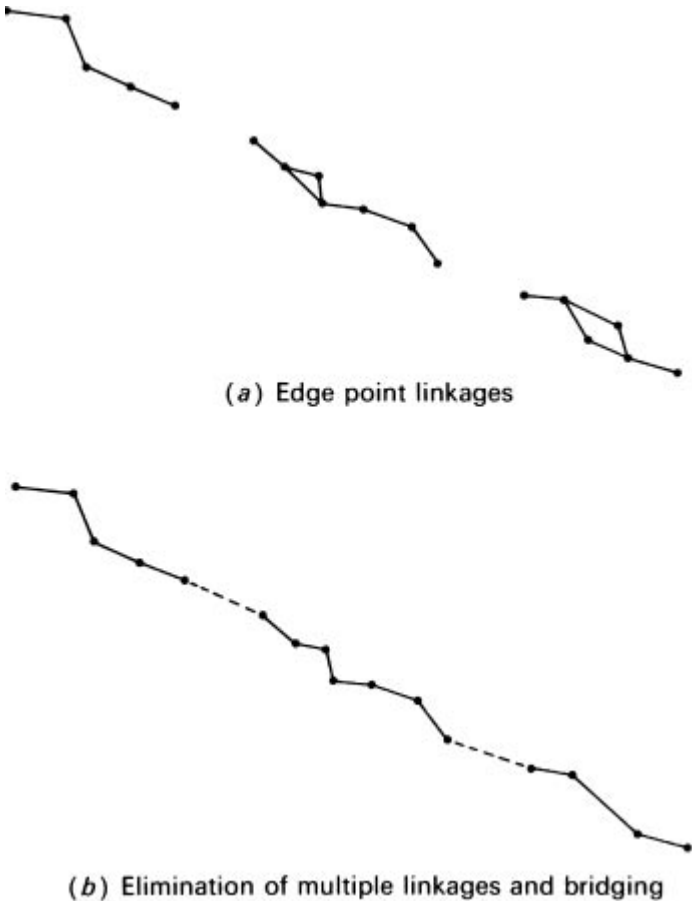


FIGURE 11.4-3. Iterative endpoint curve fitting.

11.4.2. Heuristic Edge-Linking Methods

The edge segmentation technique developed by Roberts (44) is typical of the philosophy of many heuristic edge-linking methods. In Roberts' method, edge gradients are examined in pixels blocks. The pixel whose magnitude gradient is largest is declared a tentative edge point if its magnitude is greater than a threshold value.

Then north-, east-, south-, and west-oriented lines of length 5 are fitted to the gradient data about the tentative edge point. If the ratio of the best fit to the worst fit, measured in terms of the fit correlation, is greater than a second threshold, the tentative edge point is declared valid, and it is assigned the direction of the best fit. Next, straight lines are fitted between pairs of edge points if they are in adjacent blocks and if the line direction is within degrees of the edge direction of either edge point. Those points failing to meet the linking criteria are discarded. A typical boundary at this stage, shown in Figure 11.4-4*a*, will contain gaps and multiply connected edge points. Small triangles are eliminated by deleting the longest side; small



(*a*) Edge point linkages

(*b*) Elimination of multiple linkages and bridging

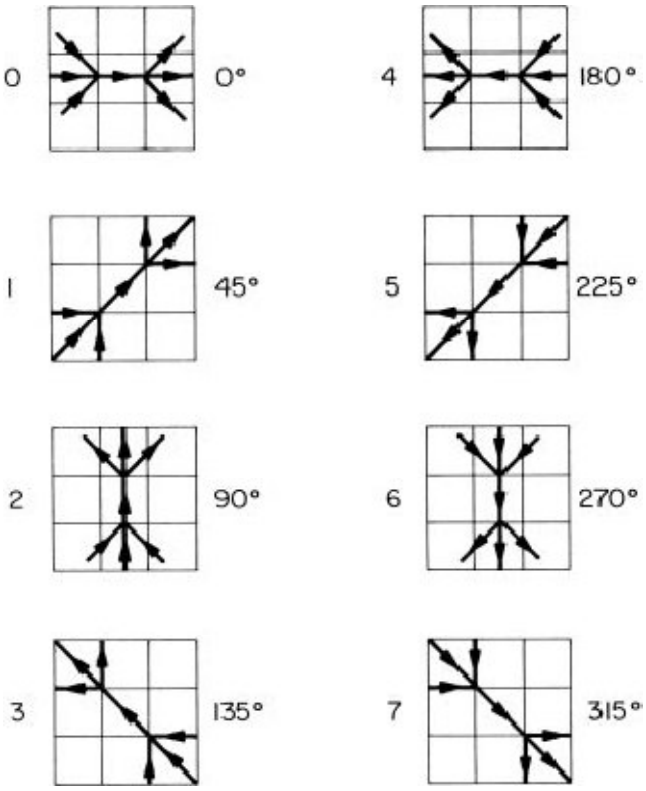
FIGURE 11.4-4. Roberts edge linking.

rectangles are replaced by their longest diagonal, as indicated in Figure 17.4-4*b*. Short spur lines are also deleted. At this stage, short gaps are bridged by straight-line connection. This form of edge linking can be used with a wide variety of edge detectors. Nevatia (45) has used a similar method for edge linking of edges produced by a
Heuckel edge detector.
Robinson (46) has suggested a simple but effective edge-linking algorithm in which edge points from an edge detector providing eight edge compass directions are examined in blocks as indicated in Figure 11.4-5. The edge point in the center of the block is declared a valid edge if it

possesses directional neighbors in the proper orientation. Extensions to larger windows should be beneficial, but the number of potential valid edge connections will grow rapidly with window size.

11.4.3. Hough Transform Edge Linking

The *Hough transform* (47–49) can be used as a means of edge linking. The Hough transform involves the transformation of a line in Cartesian coordinate space to a



point in polar coordinate space. With reference to Figure 17.4-6*a*, a straight line can be described parametrically as

$$p = x\cos\theta + y\sin\theta \qquad (11.4\text{-}1)$$

where p is the normal distance of the line from the origin and θis the angle of the origin with respect to the *x* axis. The Hough transform of the line is simply a point at coordinate in the polar domain as shown in Figure 17.4-6*b*. A family of lines passing through a common point, as shown in Figure 11.4-6*c*, maps into the connected set of points of Figure 17.4-6*d*. Now consider the three collinear points of Figure 11.4-6*e*. The Hough transform of the family of curves passing through the three points results in the set of three parametric curves in the space of Figure 11.4-6*f.* These three curves cross at a single point corresponding to the dashed line passing through the collinear points.

12.Literature Review

Because of its significant applicability, automated road information extraction from satellite and aerial imagery has been an important subject of research in remote sensing. The current effort is particularly interested in road network extraction using pattern recognition approaches. The reason is that most applications involved in safety, hazards, and disaster assessment requires detailed and accurate representations of road networks, which can only be obtained from high resolution images. However, these types of images are mostly panchromatic, such as Digital Orthophoto Quadrangles (DOQ) from U.S. Geological Survey (USGS), and one-meter resolution IKONOS images from Space Imaging. Their spectral signatures are weak and cannot be effectively utilized to identify road features. Instead, image interpretation has to rely on image texture, shapes, patterns and changes of local image intensity.

In the last twenty years or so, a variety of extraction methods have been developed. Some of them can be generally applied for linear feature extractions, while others are particularly designed for the road extraction task. The report classifies these methods into five categories: ridge finding, heuristic reasoning, dynamic programming, statistical tracking, and map matching.

It is worth noting that such a classification is mainly for generalization and convenience of description. Actually, it is very difficult to classify some of these methods into specific categories. For instance, some of the existing algorithms may use a combination of different methods. And in some other cases, a method may fall into several method categories.

12.1. Ridge Detection

Intuitively roads are linear features that are shown as ridges or valleys on an image. Therefore road finding can also be considered as a task of ridge finding. The procedure developed by Nevatia and Babu (1980) is a classic one, which starts with the convolution of an image with a number of edge filters to obtain edge magnitude and direction, then goes through a thresholding and thinning process in order to delineate edge pixels, and finally connects the delineated edge pixels to form linked line segments. Gradient Direction Profile Analysis (GDPA) (Wang et al.1992; Gong and Wang, 1997) is another representative method used for ridge finding. This method first calculates the gradient direction for each pixel, which is defined as the direction of maximum slope among the four defined directions near the pixel. As the ridge direction has the same direction as a road segment or a linear segment, and it is perpendicular to the gradient directions of the pixels with the ridge, an analysis of the gradient profile will generate the ridge pixels, which correspond to the highest points of the profile. Linking the ridge points then produces the ridgelines, or in the road detection case, the road segments.

The concept of differential geometry has also been applied to ridge detection (Steger, 1996). This method uses curve or surface fitting techniques to derive ridge locations on an image. Once the image intensity surface is represented with a mathematical equation, the first and the second derivatives of the equation can be analyzed to locate ridgelines. Image filtering methods have also been extensively discussed in the literature for edge or ridge delineation. Most frequently referenced methods in this category include Canny's edge detector (Canny, 1986) and Marr and Hildreth's zero crossing operator (Marr and Hildreth, 1980). These methods filter out the low frequency information in the image and preserve the high-frequency structures, such as roads, which are particularly useful for high-level recognition of road networks because through image filtering, a large amount of information that is irrelevant to road recognition can be thrown away at the very beginning, and then image analysis can focus on these structures that contain road networks.

12.2. Heuristic Method

The heuristic method makes use of a reasoning process similar to the human vision system. Sometimes it is also referred to as rule- or knowledge-based method. Melsels and Mintz (1990) considered a three-stage reasoning concept and applied the concept to road network extraction: a low-level stage that deals mainly with pixel segmentation, a high-level stage that models and labels feature objects, and an intermediate level of representation that interfaces between low level and high-level processing. Image primitives that are considered as building blocks of roads are first identified with pixel value checking at neighborhood levels. Intermediate level analysis will combine image primitives into line segments with the use of some reasoning mechanism (e.g., how far the combined primitives are apart; what directions each of the primitives follow; whether they parallel with each other; etc). High level processing allows further gap filling and segment grouping by considering distances, brightness, and uniformity among the grouped tokens and gaps between them.

One of the advantages of the heuristic approach is in its flexibility in dealing with problems such as linear feature alignment and fragmentation. For example, McKeown and Pane (1985) suggested that trend and the relative distances between fragments can be utilized as factors while fragmented primitives are aligned and connected. With some extension, McKeown and Denlinger (1985) also developed a road tracking system that relies on road texture correlation and road edge following alternatively: when one fails, the other will be utilized. Later, Mckeown et al (1992) and (Zlotnick, 1993) introduced methods that track roads by searching for antiparallel edges as starting points for road tracking and linking.

12.3. Dynamic Programming

Dynamic Programming (DP) has been frequently utilized in road network extraction and has been described in detail by Gruen and Li (1996). The most appealing aspect of the method is that the road recognition problem can be formulated as an optimization program, and the

solution to this program will result in the delineation of the road pixels. According to Gruen and Li, roads can be modeled with a set of mathematical equations. That is, derivatives of the gray values in the direction normal to the roads tend to be maximized, while derivatives along the road direction tend to be minimized. At the same time, roads tend to be straight lines or smooth curves, normally circular arcs, and their local curvature has an upper bound. These properties, then, can be characterized with a merit function, which can be solved using the DP technique.

The introduction of the DP technique in pattern recognition dates back to the late 1960s. Since then, the technique has been used on raw images for pixel based processing (e.g., Montanari, 1970), and also used on tokens that are based upon higher level representations (Fischler et al, 1981; Geman and Jedunak, 1996). The technique is considered to be a good approach to finding curves in noisy pictures, because it can bridge weakly connected feature elements automatically while the program searches for optimal solutions. This property is also preserved when applied to the grouping of feature elements with higher level of representations. In either case, regulatory constraints can be utilized to derive curves that will meet certain predefined geometric requirements (e.g., smoothness or curvature). More importantly, DP provides a way to serialize the optimization procedure to allow computationally attainable solutions.

12.4. Statistical Inference

Due to complexities of road images, an ability to handle uncertainties (e.g., bridge crossing, road width changing, cars and shadows on the roads, image noises, etc.) is essential. For this reason, statistical models are particularly attractive for road image representation. Cooper (1979) first came up with the idea of modeling a blob boundary or a linear feature as a Markov process.

Based upon this modeling scheme, he was able to use maximum likelihood estimation to derive the boundary representation that has the same formulation as other deterministic methods. The underlying significance of Cooper's scheme is that it provides an elegant and effective methodology for incorporating uncertainties into the linear feature recognition process.

Barzohar and Cooper (1996) explored the idea further and developed a stochastic approach that can be more sophisticatedly applied to automatic road finding. This approach makes use of the so-called geometric-stochastic model that formulates road width, direction, gray level intensity, and background intensity as a stochastic process using the Gibbs Distribution. Then road detection is achieved through the maximum a posteriori probability (MAP) estimation. The method demonstrated promising results and can be further extended to consider possibly many other types of uncertainties in road extraction. A more recent study by Yuille and Coughlan (2000) provides additional enhancements to Barzohar and Cooper's approach. This study not only allows the analysis of the time and memory complexity in the MAP estimation, but also the determination of whether roads are detectable in a given image.

The work of Geman and Jedynak (1996) is also based upon a statistic model that tracks roads through hypothesis testing. Their approach uses the testing rule that is computed from the

empirical joint distributions of tests (matched filters for short road segments) to determine whether the hypothesis (road position) is true or not. The tests are performed sequentially and a uncertainty or entropy minimization procedure is devised to facilitate testing decisions, so that new tests can be analytically identified. The method appears to work reasonably well on low resolution images, but likely adaptable to high resolution images as well.

12.5. Map Matching

The rationale behind map matching in road extraction is that there already exists a large amount of data on road systems in many parts of the world, especially in the United States. Once in house, this data can serve as a starting point for road network extraction. There are many advantages to combining remotely sensed data with existing databases (Wang at al.,1992). That is, existing data can be used as the interpretation key at the beginning, then verified, or updated, and the attributes of existing network can be also transferred to the newly updated network.

The map matching approach has been explored by several researchers (Maillard and Cavayas, 1989; Stilla, 1995; Fiest et al., 1998; Zafiropoulos, 1998). Maillard and Cavayas first introduced the map-matching approach. Using this approach, they established a road updating procedure that consists of two major algorithms. The first algorithm focuses on image-map matching to identify roads that can be found on the map and the image. The second algorithm is then to search new roads based upon the assumption that these new roads are connected to the old ones.

This approach was studied further and applied in different geographic settings (Fiset and Cavayas, 1997). To overcome some of the map matching problems of this approach, Fiset et al (1998) used a multi-layer perceptron based upon template matching to improve its performance further.

Stilla developed a syntax-oriented method to use the map knowledge as a supportive aid for image interpretation. In this study, representations of road network structures are first obtained through map analysis. Then image object models are defined and utilized to search for objects that fulfil model expectations with a given tolerance. Assessment on image objects with respect to its correspondence to the map representations results in road object identification for a given image scene. Zafiropoulos (1998) considered the concepts of deformable contours, B-Spines, Ribbon Snakes and presented a mathematical-modeling-based road verification and revision framework. With this framework, road centerlines are localized using a global least square adjustment.

12.6. Summary

From the above review of the existing methods on road network extraction, several observations can be made. First, the concept for road network extraction is relatively simple. That is, roads are shown as ridges and valleys; image intensity changes little along a road, but change drastically while cutting across a road; road direction changes tend to be smooth and have a upper bound; and roads follow some topological regularities in terms of connectivity among themselves.

Basically, many of the existing methods make use of one or several of these characteristics.

Second, reliable road network extraction remains a difficult task, and there exists no algorithm sufficiently reliable for practical use (Geman, and Jedynak, 1996). This is primarily due to the fact that the real world is too complex, and many of the existing tools can only handle very specific cases. Some may be able to handle more complex situations, but still can not be compared with a trained human operator. Mathematical models such as statistical models and dynamic programming have enhanced the ability to deal with this complexity significantly, but factors such as computational performance, implementation difficulty, and limitation of sophistication make more ambitious efforts difficult. Theoretically, a road recognition algorithm can consider all possible listed road characteristics. Practically, a road recognition algorithm can only consider a limited set of characteristics, and when these characteristics change beyond a limit, the algorithm may fail.

Finally, even though automated road extraction is a difficult problem to solve, improvement has been constantly made and there are still opportunities to be explored to further improve its performance. It appears that the semi-automated and the map-matching-based approaches tend to be more practical for short-term implementations. In the long run, studies of road image characteristics, their changes with respect to geographic background, image types, image resolutions, and development of mathematical models to represent these characteristics are critical in order to make substantive progress in this area.

13.STRATEGY OF MAIN ROADS DETECTION AND VERIFICATION

13.1 Strategy

We firstly define the main roads in particular resolution image. The definition depends on the geometric attributes. Geometric attributes include road width and road length limit. The road whose width and length is greater than a limit will be considered as 'main road'. For IKONOS 1m/pixel high resolution image, as a case study we set the width and length limit as 12m and 400m. The main road extraction is based on hierarchical strategy. Firstly, image pyramid is generated according to image resolution and main road definition. Actually a 2-layers image pyramid in which the low resolution is re-sampled to ¼ resolution so as to extract the main road in this layer is generated.

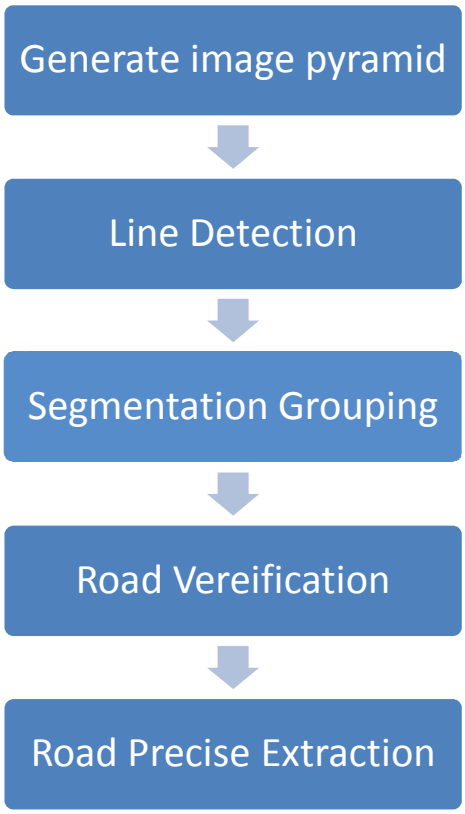


Figure 1. Workflow of automatic main road extraction

13.2 Line Detection

A 2-direction (horizontal and vertical) template matching is applied to detect road central point candidates in low resolution layer of the image pyramid. The procedure is shown in figure 2.

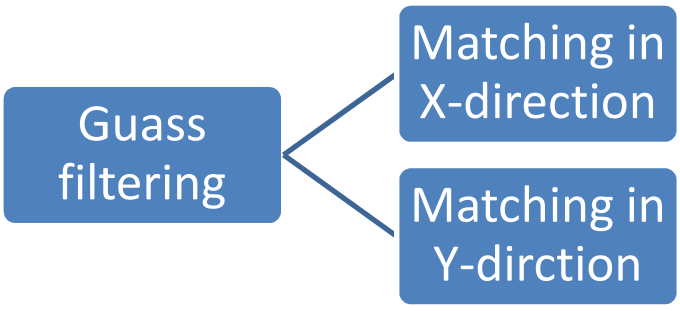


Figure 2. Line detection based on template matching

(a) IKONOS image

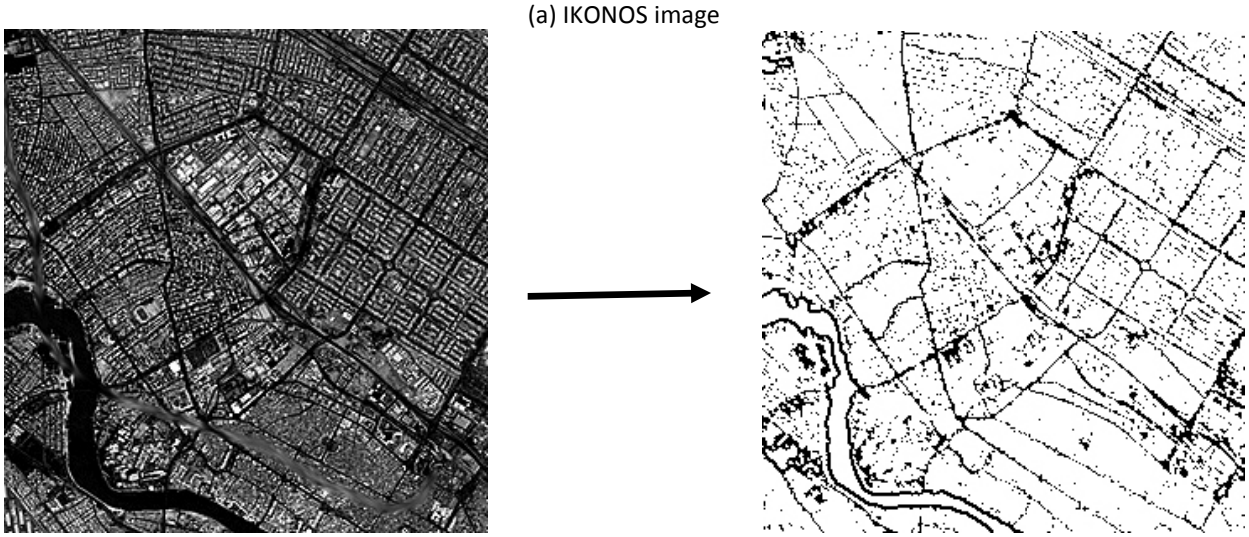


Figure 3 . Binary detection from IKONOS image after thresholding (19).

This is a very simple and 'easy to use' detector. At first, Gaussian filtering smoothes the image for reducing noise and then corresponding template is generated by road profile. Road profile

template is used to matching image signal by correlating along x (horizontal) and y (vertical) direction, the local peak correlation coefficient indicates the existing of candidate of road centre point. Figure 3 is an example of the detection in a ¼ resolution layer of the image pyramid. The detection from low resolution layer is much more effective, but it cannot get so accurate geometric location of road point. However, finally the precise delineation will be applied to adjust the road central lines based on original resolution image. After Line detection, candidate lines should be traced to vector form for next step of line segments grouping.

13.3 Lines Grouping

Grouping plays a very important role in road extraction. Due to the alignment and fragmentation of the candidate segments gotten from line detection step, grouping only by geometric attributes is often unreliable, so we should take more information into account. On the other hand, if one tries to group all detected candidate segments, it will be too complex and the computational cost is too high because of the great amount of segments. Now the question is: how to guarantee the 'reliability' of road grouping? In this paper, the grouping is based on a multi-level grouping according sorted segments by 'labeling' possibility of being road segment' to each segment. Figure 4 shows the grouping strategy.

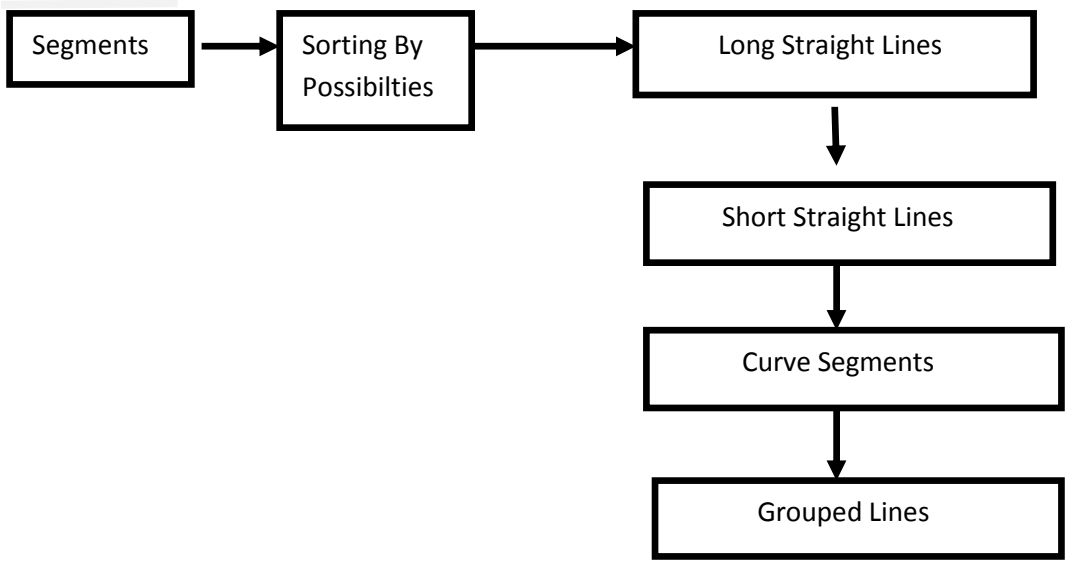


Figure 4. Grouping Procedure

Grouping starts from segments with higher possibility of being road segment. Long, straight, and bright line segment has the higher score while short, curve, and dark line has lower score. From the viewpoint of vision, one tries to find salient, bigger and more global structuralized features when one recognizes objects from image. This is the ability of so called 'perceptual organization or grouping' which can be defined as imposing

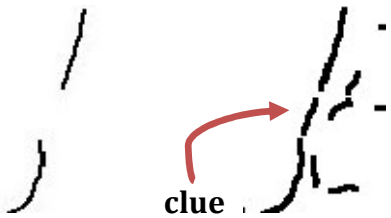


Figure 5. Link clue in the gap

structural regularity on sensory data. To evaluate the possibility of each candidate segment an evaluation function is built depend on geometric and photometric attributes, and then segments are sorted by the possibilities. Grouping starts from longer straight line segment, sequentially shorter, curve segments are grouped.

Grouping takes advantage of clues from detailed line detection result. The link clue between grouped longer segments mainly refers to short lines. See figure 5, shorter lines in the gap between segments strongly indicate the linkage. The strategy can not only reduce the computational complexity by starting grouping from segments having higher possibility of being road centreline, but also enhance the reliability of grouping via using the clues between segment gaps which are detected by line detector.

Grouping is based on the Gastalt principles of similarity, continuality and proximity. The salient linear feature on the image – the main road can be easily perceived by human vision system even the feature is locally occluded, perceptual grouping plays core role in segment linking.

The implementation of grouping is depended on a global link matrix for extracting collinear segment (Daniel C 1999). The segment connectivity matrix ***A*** can be formed by considering the valid collinear junctions extracted in the previous steps. If *n* is the number of segments, ***A*** is of dimension *n * n*, with *ai, j*, if segments *i* and *j* are collinear, and 0 otherwise. The candidate collinear chains are extracted from the matrix ***A*** by tracing the minimal-cost route (the sum of link possibilities is to be maximum). Figure 6 shows the collinear grouping work flow.

It's an iteration procedure that ends when there is no new link generated.

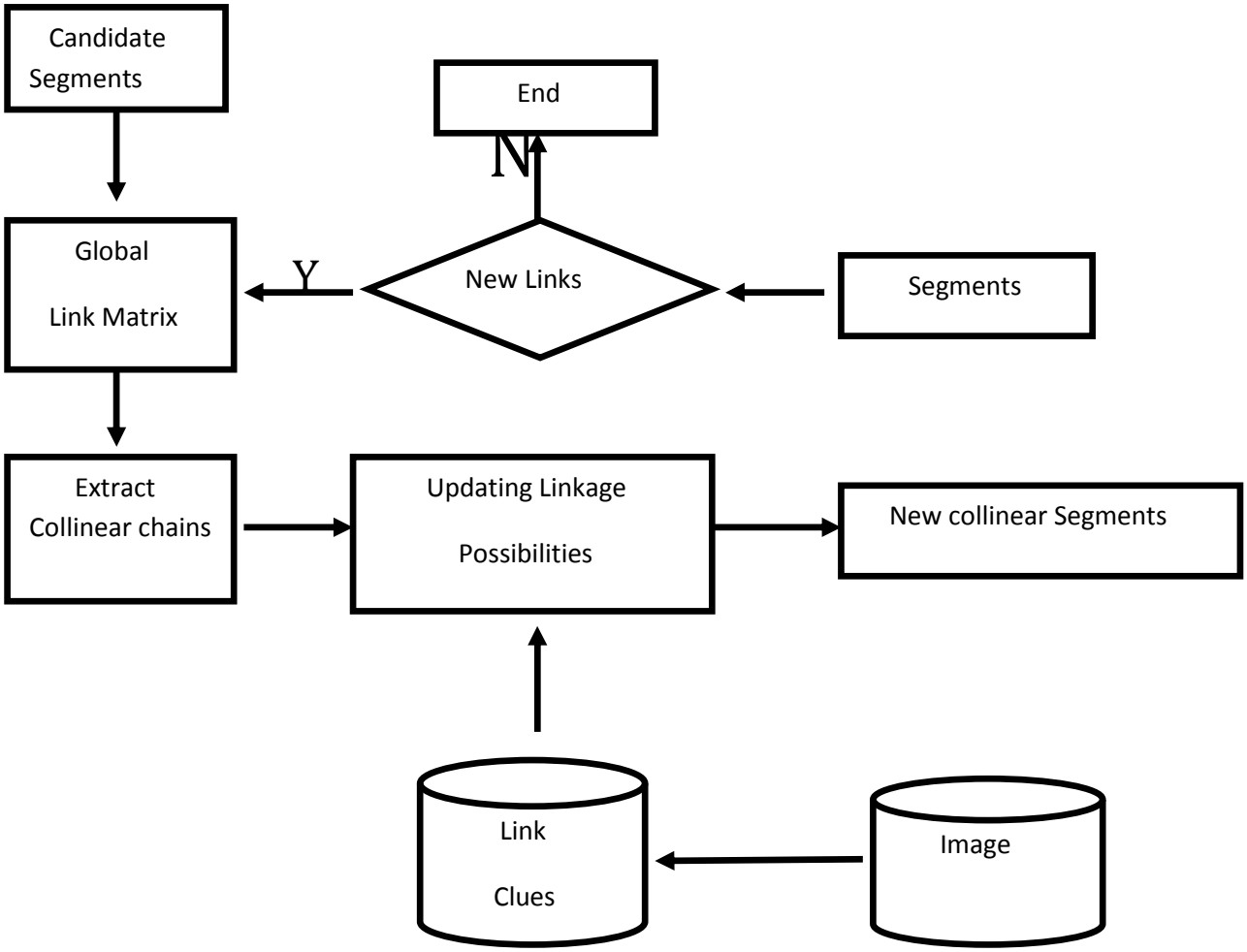


Figure 6. Collinear grouping work flow

3.4. Verification and Precise Delineation of Extracted Road Centerline

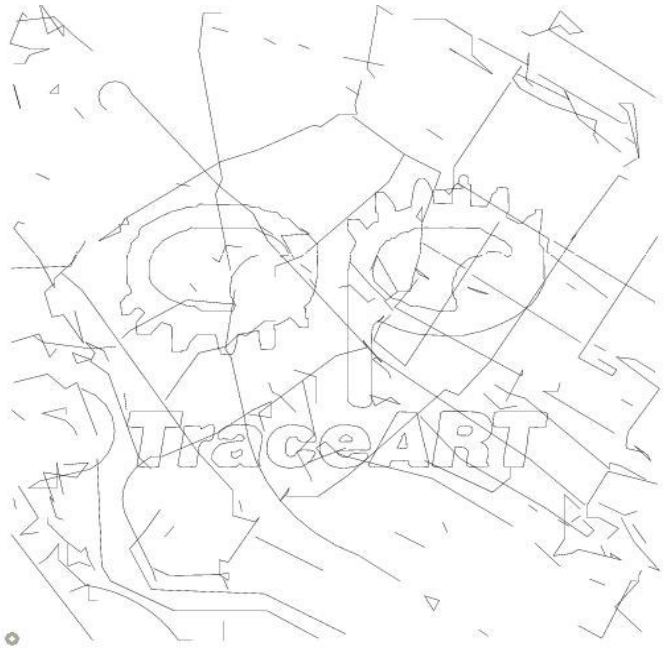
To verify the extracted centerlines, original resolution image is used to evaluate the 'score' of each line. Firstly, a length threshold is set to remove all short, isolated lines by applying a connection analyzer because the main road means that it has definitely length or is connected with other road parts; and then other longer lines are evaluated by peak value of profiletemplate matching. Correlation mean of each segment indicates the photometric possibility of being road, and the length and curvature of the centerline indicates the geometric attribute of being road. An evaluation function is set up to combine the geometric and photometric attribute together by weighted summing of the 2 items. So for each maintained

centerline, there is an assessment value to indicate the reliability or possibility of road detection.

To get precise delineation of the road centerline, least square template matching is applied into the correction of centerline curve in original resolution image. The algorithm is similar to semiautomatic road extraction from initial seed points and image (Gruen, A.and H. Li, 1997).

14.IMPLEMENTATION

Figure 7 (a) is the extraction result from image shown in figure 3(a), the black lines are the identified road centrelines; Figure 7 (b) shows the collinear grouping procedure, it starts from the segments with higher possibilities and the link threshold is higher as mentioned in section 2.3, and then shorter and curve segments are grouped by setting lower link threshold and referencing link clues. In (b), it shows the starting segment number is 393, after grouping it becomes 296; in second stage, more segments with lower possibility are added into grouping procedure and the segment number is from 719 varying to 637, as final processing, simple topology analysis is used to remove the isolated and short segments, so the remaining main road centrelines number reduces to 31



(a) Extraction result

Checking the result, we find the extracted main road is quite correc, while it is not so correct and completed in the complex building site because in this case the extraction strategy and algorithm based on segment grouping is not so reliable because even by human vision system to identify the complicated structure of road or street network is still difficult due to the dependence of contextual and more information needed.

CONCLUSION AND FUTURE WORK

Main road (centerline) extraction from high resolution satellite image is very useful for GIS database revision, change detection, and some applications that transportation infrastructure information, such as disaster management, urban planning plays an important role. Presented method in this paper is trying to automatically extract the prior defined main road centerline.
The hierarchical grouping strategy applied could reliably extract most of main road centerlines in the open rural areas even though there are some occlusions (caused by clouds, trees etc). The method attempts to simulate the perceptual capability of human vision system to find salient linear feature from image.

The experimental result indicates the potential utility of the method. For complex scenes, more information should be integrated in order to obtain more robust result. For acquiring more reliable result (no wrong extraction), we should take more road type (means more profile template) and more reliable verification algorithm into account, that can be done by combining kinds of template types and more segment attributes together for line detection and final assessment.

Future work also includes extend the main road extraction to full road network extraction, because the main road network is very strong indication and contextual information for smaller road extraction and identification.

REFERENCES

1. Baumgartner, A., C. Steger, H. Mayer, and W. Eckstein, 1999.*Automatic road extraction based on multi-scale,grouping and context*, Photogrammetric Engineering & Remote Sensing,

2. Vol .65 (7), pp. 777-785. Daniel C, 1999. *A probabilistic method for extracting chains of collinear segments*. Computer Vision and Image Understanding, Vol 76(1), pp. 36-53.

3. Gruen, A., E. Baltsavias, and O. Henricsson (editors), 1997. Automatic Extraction of Man-Made Objects from Aerial and Space Images II, Birkhaeuser Verlag, Basel, Sweezerland.

4. Gruen, A.and H. Li, 1995. Road extraction from aerial and satellite images by dynamic programming, ISPRS Journal of Photogrammetry and Remote Sensing, Vol .50 (4), pp. 11-21.

5. Heipke, C., C. Steger and R. Multhammer, 1996. A hierarchical approach to automatic road extraction from aerial imagery, In: *Integrating Photogrammetric Techniques with scene analysis and machine vision II*, Procceding of SPIE (D.M. Mckeown and I.J. Dowman, editors) (2486):222-231.

6. Mayer H. and C. Steger, 1998. Scale-space events and their link to abstraction, *ISPRS Journal of Photogrammetry and RemoteSensing*, Vol .53, pp. 62-75, McKeown, D.M, Harvey, and J, McDermott, 1985. Rule-based interpretation of aerial imagery, *IEEE Tran On PAMI-7*, No 5,pp. 570-585.

7. Treash K, Amaratunga K, 2000. Automatic road detection in grayscale aerial images, *Journal of Computing in Civil Engineering*, Vol 14 (1),pp.60-69.

8. Trinder JC, Wang Y.D, 1998. Automatic road extraction from aerial images, *Digital Signal Processing*, Vol 8 (4), pp.215-224. OCT