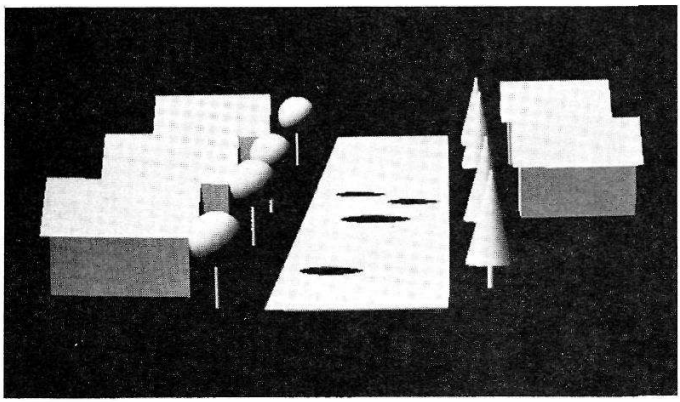
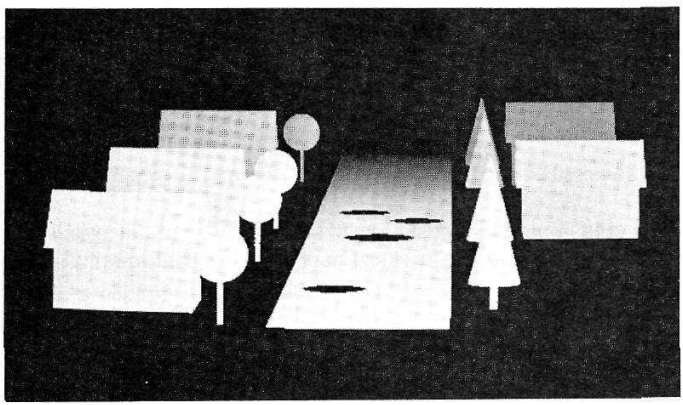


(a)



(b)

Fig. 2.26 Intensity and range images. (a) A (synthesized) intensity image of a street scene with potholes. The roofs all have the same intensity, which is different from the walls; (b) a corresponding range image. The wall and roof of each house have similar ranges, but the ranges differ from house to house.

One basic difference between sound and visible light ranging is that a light beam is usually reflected off just one surface, but that a sound beam is generally partially transmitted and partially reflected by "surfaces." The returning sound pulse has structure determined by the discontinuities in impedence to sound found in the medium through which it has passed. Roughly, a light beam returns information about a spot, whereas a sound beam can return information about the medium in the entire column of material. Thus, although sound itself travels relatively slowly, the data rate implicit in the returning structured sound pulse is quite high. Figure 2.27 shows an image made using the range data from ultrasound. The

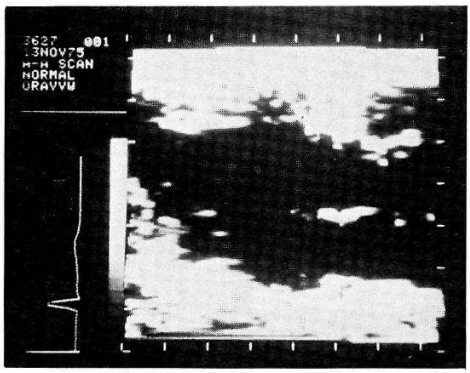


Fig. 2.27 Image made from ultrasound ranging.

sound pulses emanate from the top of the image and proceed toward the bottom, being partially reflected and transmitted along the way. In the figure, it is as if we were looking perpendicular to the beams, which are being displayed as brighter where strong reflectance is taking place. A single "scan line" of sound thus produces an image of an entire planar slice of medium.

2.3.3 Reconstruction Imaging

Two-dimensional reconstruction has been the focus of much research attention because of its important medical applications. High-quality images such as that shown in Fig. 1.2b can be formed by multiple images of x-ray projection data. This section contains the principles behind the most important reconstruction algorithms. These techniques are discussed in more detail with an expanded list of references in [Gordon and Herman 1974]. For a view of the many applications of two-dimensional reconstruction other than transmission scanning, the reader is referred to [Gordon et al. 1975].

Figure 2.28 shows the basic geometry to collect one-dimensional projections of two-dimensional data. (Most systems construct the image in a plane and repeat this technique for other planes; there are few true three-dimensional reconstruction systems that use planes of projection data simultaneously to construct volumes.)

In many applications sensors can measure the one-dimensional *projection* of two-dimensional image data. The projection $g(x')$ of an ideal image $f(x, y)$ in the direction θ is given by $\int f(x', y')\, dy'$ where $\mathbf{x}' = R_\theta \mathbf{x}$. If enough different projections are obtained, a good approximation to the image can be obtained with two-dimensional reconstruction techniques.

From Fig. 2.28, with the source at the first position along line AA', we can obtain the first projection datum from the detector at the first position along BB'. The line AB is termed a ray and the measurement at B a ray sum. Moving the source

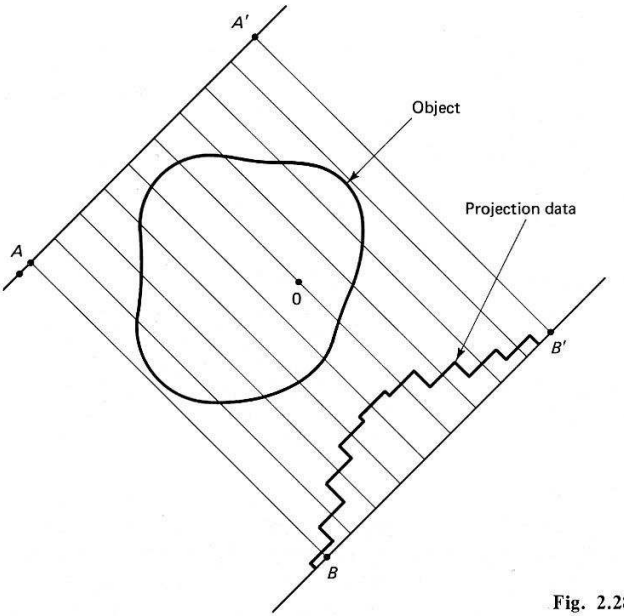


Fig. 2.28 Projection geometry.

and detector along lines AA' and BB' in synchrony allows us to obtain the entire data for projection 1. Now the lines AA' and BB' are rotated by a small angle $d\theta$ about 0 and the process is repeated. In the original x-ray systems $d\theta$ was 1° of angle, and 180 projections were taken. Each projection comprised 160 transmission measurements. The reconstruction problem is simply this: Given the projection data $g_k(x')$, $k = 0, \ldots, N-1$, construct the original image $f(\mathbf{x})$.

Systems in use today use a fan beam rather than the parallel rays shown. However, the mathematics is simpler for parallel rays and illustrates the fundamental ideas. We describe three related techniques: summation, Fourier interpolation, and convolution.

The Summation Method

The summation method is simple: Distribute every ray sum $g_k(x')$ over the image cells along the ray. Where there are N cells along a ray, each such cell is incremented by $\frac{1}{N} g(x')$. This step is termed *back projection.* Repeating this process for every ray results in an approximate version of the original [DeRosier 1971]. This technique is equivalent (within a scale factor) to blurring the image, or convolving it with a certain point-spread function. In the continuous case of infinitely many projections, this function is simply the radically symmetric $h(r) = 1/r$.

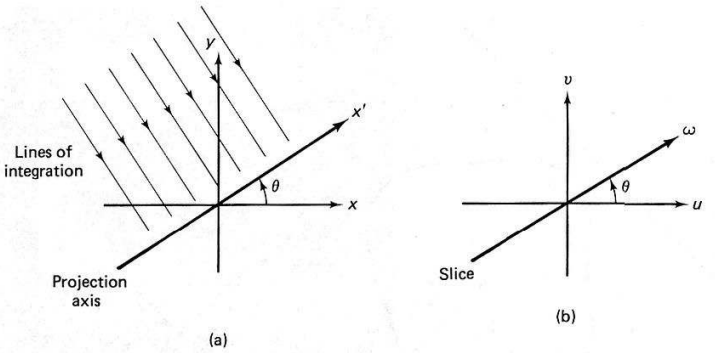


Fig. 2.29 Basis of Fourier techniques. (a) Projection axis x′; (b) corresponding axis in Fourier Space.

Fourier Algorithms

If a projection is Fourier-transformed, it defines a line through the origin in frequency space (Fig. 2.29). To show this formally, consider the expression for the two-dimensional transform

$$F(\mathbf{u}) = \int\int f(x, y) \exp[j2\pi(ux + vy)]\, dx\, dy \tag{2.47}$$

Now consider $y = 0$ (projection onto the x axis): $x' = x$ and

$$g_0(x') = \int f(x, y)\, dy \tag{2.48}$$

The Fourier transform of this equation is

$$\begin{aligned} \mathfrak{F}[g_0(x')] &= \int\int [f(x, y)\, dy] \exp j2\pi ux\, dx \\ &= \int\int f(x, y) \exp j2\pi ux\, dy\, dx \end{aligned} \tag{2.49}$$

which, by comparison with (2.47), is

$$\mathfrak{F}[g_0(x')] = F(u, 0) \tag{2.50}$$

Generalizing to any θ, the transform of an arbitrary $g(x')$ defines a line in the Fourier space representation of the cross section. Where $S_k(\omega)$ is the cross section of the Fourier transform along this line,

$$\begin{aligned} S_k(\omega) &= F(u\cos\theta,\ u\sin\theta) \\ &= \int g_k(x') \exp[-j2\pi u(x')]\, dx' \end{aligned} \tag{2.51}$$

Thus one way of reconstructing the original image is to use the Fourier transform of the projections to define points in the transform of $f(x)$, interpolate the undefined points of the transform from the known points, and finally take the inverse transform to obtain the reconstructed image.

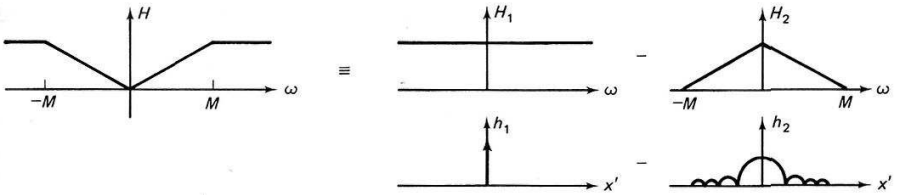


Fig. 2.30 Convolution method.

This technique can be applied with transforms other than the Fourier transform, and such methods are discussed in [DeRosier 1971; Crowther and Klug 1971].

The Convolution Method

The convolution method is the natural extension of the summation method. Since the summation method produces an image degraded from its convolution with some function h, one can remove the degradation by a "deconvolution." The straightforward way to accomplish this is to Fourier-transform the degraded image, multiply the result by an estimate of the transformed h^{-1}, and inverse-Fourier-transform the result. However, since all the operations are linear, a faster approach is to deconvolve the projections before performing the back projection. To show this formally, we use the inverse transform

$$f(\mathbf{x}) = \int\int F(u, v) \exp[j2\pi(ux + vy)]\,du\,dv \tag{2.52}$$

Changing to cylindrical coordinates (ω, θ) yields

$$f(\mathbf{x}) = \int\int F_\theta(\omega) \exp[j2\pi\omega(x\cos\theta + y\sin\theta)]|\omega|\,d\omega\,d\theta \tag{2.53}$$

Since $x' = x\cos\theta + y\sin\theta$, rewrite Eq. (2.53) as

$$f(\mathbf{x}) = \int \mathfrak{F}^{-1}\{F_\theta(\omega)H(\omega)\}\,d\theta \tag{2.54}$$

Since the image is bandlimited at some interval $(-\omega_m, \omega_m)$ one can define $H(\omega)$ arbitrarily outside of this interval. Therefore, $H(\omega)$ can be defined as a constant minus a triangular peak as shown in Fig. 2.30. Finally, the operation inside the integral in Eq. (2.54) is a convolution. Using the transforms shown in Fig. 2.30,

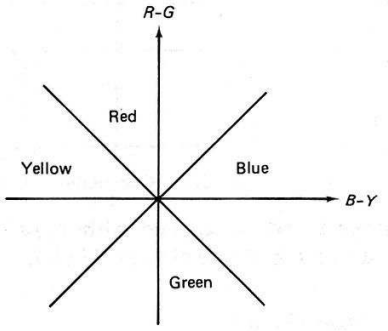
$$f(\mathbf{x}) = \int [f_\theta(x') - f_\theta(x')\omega_m \mathrm{sinc}^2(\omega_m x')]\,d\theta \tag{2.55}$$

Owing to its speed and the fact that the deconvolutions can be performed while the data are being acquired, the convolution method is the method employed in the majority of systems.

EXERCISES

2.1 In a binocular animal vision system, assume a focal length f of an eye of 50 mm and a separation distance d of 5 cm. Make a plot of Δx vs. $-z$ using Eq. (2.9). If the resolution of each eye is on the order of 50 line pairs/mm, what is the useful range of the binocular system?

2.2 In an opponent-process color vision system, assume that the following relations hold:



For example, if the $(R-G, B-Y, W-Bk)$ components of the opponent-process system are $(0.5, 3, 4)$, the perceived color will be blue.

Work out the perceived colors for the following (R,G,B) measurements:

(a) $(0.2, 0.3, 0.4)$ (b) $(0.2, 0.3, 0)$ (c) $(7, 4, 1)$

2.3 Develop an indexing scheme for a hexagonal array and define a Euclidean distance measure between points in the array.

2.4 Assume that a one-dimensional image has the following form:

$$f(x) = \cos(2\pi u_o x)$$

and is sampled with $u_s = u_o$. Using the graphical method of Section 2.2.6, find an expression for $f(x)$ as given by Eq. (2.49). Is this expression equal to the original image? Explain.

2.5 A certain image has the following Fourier transform:

$$F(\mathbf{u}) = \begin{cases} \text{nonzero} & \text{inside a hexagonal domain} \\ 0 & \text{otherwise} \end{cases}$$

(a) What are the smallest values for u and v so that $F(\mathbf{u})$ can be reconstructed from $F_x(\mathbf{u})$?

(b) Suppose now that rectangular sampling is *not* used but that now the u and v directions subtend an angle of $\pi/3$. Does this change your answer as to the smallest u and v? Explain.

2.6 Extend the binocular imaging model of Fig. 2.3 to include convergence: Let the two imaging systems pivot in the $y = 0$ plane about the viewpoint. Let the system have a baseline of $2d$ and be converged at some angle θ such that a point (x, y, z) appears at the origin of each image plane.

(a) Solve for z in terms of r and θ.

(b) Solve for z in this situation for points with nonzero disparity.

2.7 Compute the convolution of two Rect functions, where

$$\text{Rect}(x) = \begin{cases} 1 & 0 < x < 1 \\ 0 & \text{otherwise} \end{cases}$$

Show the steps in your calculations.

2.8

$$\text{Rect}(x) = \begin{cases} b & \text{for } |x| < a \\ 0 & \text{otherwise} \end{cases}$$

(a) What is $\text{Rect}(x) * \delta(x-a)$?

(b) What is the Fourier transform of $f(x)$ where $f(x) = \text{Rect}(x+c) + \text{Rect}(x-c)$ and $c >$ a?

2.9 A digitizer has a sampling interval of $\Delta x = \Delta y = \Delta$. Which of the following images can be represented unambiguously by their samples? (Assume that effects of a finite image domain can be neglected.)

(a) $(sin(\pi x/\Delta))/(\pi x/\Delta)$

(b) $cos(\pi/x/2\Delta)\cos(3\pi x/4\Delta)$

(c) $\text{Rect}(x)$ (see Problem 2.8)

(d) e^{-ax^2}

REFERENCES

AGIN, G. J. "Representation and description of curved objects" (Ph.D. dissertation). AIM-173, Stanford AI Lab, October 1972.

ANDREWS, H. C. and B. R. HUNT. *Digital Image Restoration.* Englewood Cliffs, NJ: Prentice-Hall, Inc., 1977.

CROWTHER, R. A. and A. KLUG. "ART and science, or, conditions for 3-d reconstruction from electron microscope images." *J. Theoretical Biology 32*, 1971.

DEROSIER, D. J. "The reconstruction of three-dimensional images from electron micrographs." *Contemporary Physics 12*, 1971.

DUDA, R. O. and P. E. HART. *Pattern Recognition and Scene Analysis.* New York: Wiley, 1973.

GARVEY, T. D. "Perceptual strategies for purposive vision." Technical Note 117, AI Center, SRI International, September 1976.

GONZALEZ, R. C. and P. WINTZ. *Digital Image Processing.* Reading, MA: Addison-Wesley, 1977.

GORDON, R. and G. T. HERMAN. "Three-dimensional reconstruction from projections: a review of algorithms." *International Review of Cytology 38*, 1974, 111–151.

GORDON, R., G. T. HERMAN, and S. A. JOHNSON. "Image reconstruction from projections." *Scientific American*, October 1975.

HERING, E. "Principles of a new theory of color sense." *In Color Vision*, R.C. Teevan and R.C. Birney (Eds.). Princeton, NJ: D. Van Nostrand, 1961.

HORN, B. K. P. "Understanding image intensities." *Artificial Intelligence 8*, 2, April 1977, 201–231.

HORN, B. K. P. and R. W. SJOBERG. "Calculating the reflectance map." *Proc.*, DARPA IU Workshop, November 1978, 115–126.

HURVICH, L. M. and D. JAMESON. "An opponent-process theory of color vision." *Psychological Review 64*, 1957, 384–390.

JAIN, A. K. "Advances in mathematical models for image processing." *Proc. IEEE 69*, 5, May 1981, 502-528.

JOBLOVE, G. H. and D. GREENBERG. "Color spaces for computer graphics." *Computer Graphics 12*, 3, August 1978, 20–25.

KENDER, J. R. "Saturation, hue, and normalized color: calculation, digitization effects, and use." Technical Report, Dept. of Computer Science, Carnegie-Mellon Univ., November 1976.

LAND, E. H. "The retinex theory of color vision." *Scientific American,* December 1977, 108–128.

MUNSELL, A. H. *A Color Notation*, 8th ed. Baltimore, MD: Munsell Color Co., 1939.

NICODEMUS, F. E., J. C. RICHMOND, J. J. HSIA, I. W. GINSBERG, and T. LIMPERIS. "Geometrical considerations and nomenclature for reflectance." NBS Monograph 160, National Bureau of Standards, U.S. Department of Commerce, Washington, DC, October 1977.

NITZAN, D., A. BRAIN, and R. DUDA. "The measurement and use of registered reflectance and range data in scene analysis." *Proc. IEEE 65*, 2, February 1977.

POPPLESTONE, R. J., C. M. BROWN, A. P. AMBLER, and G. F. CRAWFORD. "Forming models of plane-and-cylinder faceted bodies from light stripes." *Proc.*, 4th IJCAI, September 1975, 664-668.

PRATT, W. K. *Digital Image Processing.* New York: Wiley-Interscience, 1978.

ROSENFIELD A. and A. C. KAK. *Digital Picture Processing.* New York: Academic Press, 1976.

SMITH, A. R. "Color gamut transform pairs." *Computer Graphics 12*, 3, August 1978, 12–19.

SUGIHARA, K. "Dictionary-guided scene analysis based on depth information." In *Progress Report on 3-D Object Recognition.* Bionics Research Section, ETL, Tokyo, March 1977.

TENENBAUM, J. M. and S. WEYL. "A region-analysis subsystem for interactive scene analysis." *Proc.*, 4th IJCAI, September 1975, 682–687.

WAAG, R. B. and R. GRAMIAK. "Methods for ultrasonic imaging of the heart." *Ultrasound in Medicine and Biology 2*, 1976, 163–170.

WILL, P. M. and K. S. PENNINGTON. "Grid coding: a preprocessing technique for robot and machine vision." *Artificial Intelligence 2*, 3/4, Winter 1971, 319–329.

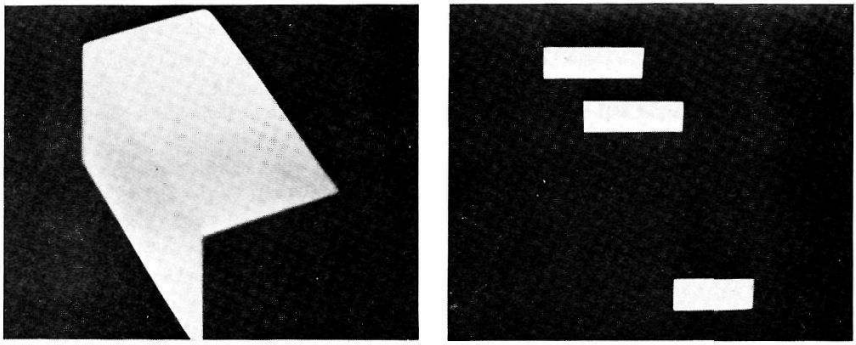
Early Processing 3

3.1 RECOVERING INTRINSIC STRUCTURE

The imaging process confounds much useful physical information into the gray-level array. In this respect, the imaging process is a collection of degenerate transformations. However, this information is not irrevocably lost, because there is much spatial redundancy: Neighboring pixels in the image have the same or nearly the same physical parameters. A collection of techniques, which we call *early processing*, exploits this redundancy in order to undo the degeneracies in the imaging process. These techniques have the character of transformations for changing the image into "parameter images" *or intrinsic images* [Barrow and Tenenbaum 1978; 1981] which reflect the spatial properties of the scene. Common intrinsic parameters are surface discontinuities, range, surface orientation, and velocity.

In this chapter we neglect high-level internal model information even though it is important and can affect early processing. Consider the case of the perceived central edge in Fig. 3.1a. As shown by Fig. 3.1b, which shows portions of the same image, the central edge of Fig. 3.1a is not present in the data. Nevertheless, the human perceiver "sees" the edge, and one reasonable explanation is that it is a product of an internal block model. Model-directed activity is taken up in later chapters. These examples show how high level models (e.g., circles) can affect low-level processors (e.g., edge finders). However, for the purposes of study it is often helpful to neglect these effects. These simplifications make it easier to derive the fundamental constraints between the physical parameters and gray levels. Once these are understood, they can be modified using the more abstract structures of later chapters.

Most early computer vision processing can be done with parallel computations whose inputs tend to be spatially localized. When computing intrinsic images



(a) (b)

Fig. 3.1 (a) A perceived edge. (b) Portions of image in (a) showing the lack of image data.

the parallel computations are iterated until the intrinsic parameter measurements converge to a set of values. A computation that falls in the parallel-iterative category is known in computer vision as *relaxation* [Rosenfeld et al. 1976]. Relaxation is a very general computational technique that is useful in computer vision. Specific examples of relaxation computations appear throughout the book; general observations on relaxation appear in Chapter 12.

This chapter covers six categories of early processing techniques:

1. *Filtering* is a generic name for techniques of changing image gray levels to enhance the appearance of objects. Most often this means transformations that make the intensity discontinuities between regions more prominent. These transformations are often dependent on gross object characteristics. For example, if the objects of interest are expected to be relatively large, the image can be blurred to erase small intensity discontinuities while retaining those of the object's boundary. Conversely, if the objects are relatively small, a transformation that selectively removes large discontinuities may be appropriate. Filtering can also compensate for spatially varying illumination.

2. *Edge operators* detect and measure very local discontinuities in intensity or its gradient. The result of an edge operator is usually the magnitude and orientation of the discontinuity.

3. *Range transforms* use known geometry about stereo images to infer the distance of points from the viewer. These transforms make use of the inverse perspective transform to interpret how points in three-dimensional space project onto stereo pairs. A correspondence between points in two stereo images of known geometry determines the range of those points. Relative range may also be derived from local correspondences without knowing the imaging geometry precisely.

4. *Surface orientation* can be calculated if the source illumination and reflectance properties of the surface are known. This calculation is sometimes called

"shape from shading." Surface orientation is particularly simple to calculate when the source illumination can be controlled.

5. *Optical flow*, or velocity fields of image points, can be calculated from local temporal and spatial variations in sequences of gray-level images.

6. A *pyramid* is a general structure for representing copies of the image at multiple resolutions. A pyramid is a "utility structure" which can dramatically improve the speed and effectiveness of many early processing and later segmentation algorithms.

3.2 FILTERING THE IMAGE

Filtering is a very general notion of transforming the image intensities in some way so as to enhance or deemphasize certain features. We consider only transforms that leave the image in its original format: a spatial array of gray levels. Spurred on by the needs of planetary probes and aerial reconnaissance, filtering initially received more attention than any other area of image processing and there are excellent detailed reference works (e.g., [Andrews and Hunt 1977; Pratt 1978; Gonzalez and Wintz 1977]). We cannot afford to examine these techniques in great detail here; instead, our intent is to describe a set of techniques that conveys the principal ideas.

Almost without exception, the best time to filter an image is at the image formation stage, before it has been sampled. A good example of this is the way chemical stains improve the effectiveness of microscopic tissue analysis by changing the image so that diagnostic features are obvious. In contrast, filtering after sampling often emphasizes random variations in the image, termed *noise*, that are undesirable effects introduced in the sampling stage. However, for cases where the image formation process cannot be changed, digital filtering techniques do exist. For example, one may want to suppress low spatial frequencies in an image and sharpen its edges. An image filtered in this way is shown in Fig. 3.2.

Note that in Fig. 3.2 the work of recognizing real-world objects still has to be done. Yet the edges in the image, which constitute object boundaries, have been made more prominent by the filtering operation. Good filtering functions are not easy to define. For example, one hazard with Fourier techniques is that sharp edges in the filter will produce unwanted "ringing" in the spatial domain, as evidenced by Fig. 2.5. Unfortunately, it would be too much of a digression to discuss techniques of filter design. Instead, the interested reader should refer to the references cited earlier.

3.2.1 Template Matching

Template matching is a simple filtering method of detecting a particular feature in an image. Provided that the appearance of this feature in the image is known accu-

(a) (b)

Fig. 3.2 Effects of high frequency filtering. (a) Original image. (b) Filtered image.

rately, one can try to detect it with an operator called a *template*. This template is, in effect, a subimage that looks just like the image of the object. A similarity measure is computed which reflects how well the image data match the template for each possible template location. The point of maximal match can be selected as the location of the feature. Figure 3.3 shows an industrial image and a relevant template.

Correlation

One standard similarity measure between a function $f(\mathbf{x})$ and a template $t(\mathbf{x})$ is the Euclidean distance $d(\mathbf{y})$ squared, given by

$$d(\mathbf{y})^2 = \sum_{\mathbf{x}} [f(\mathbf{x}) - t(\mathbf{x} - \mathbf{y})]^2 \tag{3.1}$$

By $\sum_{\mathbf{x}}$ we mean $\sum_{x=-M}^{M} \sum_{y=-N}^{N}$, for some M, N which define the size of the template extent. If the image at point $\mathbf{y}$ is an exact match, then $d(\mathbf{y}) = 0$; otherwise, $d(\mathbf{y}) > 0$. Expanding the expression for d^2, we can see that

$$d^2(\mathbf{y}) = \sum_{\mathbf{x}} [f^2(\mathbf{x}) - 2f(\mathbf{x})t(\mathbf{x} - \mathbf{y}) + t^2(\mathbf{x} - \mathbf{y})] \tag{3.2}$$

Notice that $\sum_{\mathbf{x}} t^2(\mathbf{x} - \mathbf{y})$ is a constant term and can be neglected. When $\sum_{\mathbf{x}} f^2(\mathbf{x})$ is approximately constant it too can be discounted, leaving what is called the *cross* correlation between f and t.

$$R_{ft}(\mathbf{y}) = \sum_{\mathbf{x}} f(\mathbf{x})t(\mathbf{x} - \mathbf{y}) \tag{3.3}$$

This is maximized when the portion of the image "under" t is identical to t.

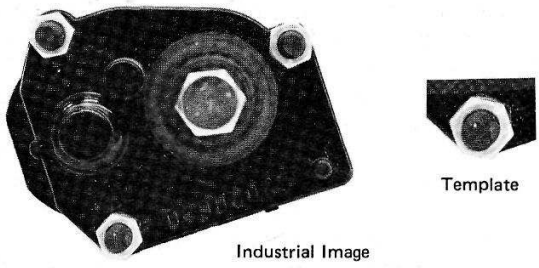


Fig. 3.3 An industrial image and template for a hexagonal nut.

One may visualize the template-matching calculations by imagining the template being *shifted* across the image to different offsets; then the superimposed values at this offset are *multiplied* together, and the products are *added.* The resulting sum of products forms an entry in the "correlation array" whose coordinates are the offsets attained by the source template.

If the template is allowed to take *all* offsets with respect to the image such that some overlap takes place, the correlation array is larger than either the template or the image. An $n \times n$ image with an $m \times m$ template yields an $(n + m - 1 \times n + m - 1)$ correlation array. If the template is not allowed to shift off the image, the correlation array is $(n - m + 1 \times n - m + 1)$; for $m < n$. Another form of correlation results from computing the offsets modulo the size of the image; in other words, the template "wraps around" the image. Being shifted off to the right, its right portion reappears on the left of the image. This sort of correlation is called *periodic* correlation, and those with no such wraparound properties are called *aperiodic.* We shall be concerned exclusively with aperiodic correlation. One can always modify the input to a periodic correlation algorithm by padding the outside with zeros so that the output is the aperiodic correlation.

Figure 3.4 provides an example of (aperiodic) "shift, add, multiply" template matching. This figure illustrates some difficulties with the simple correlation measure of similarity. Many of the advantages and disadvantages of this measure stem from the fact that it is linear. The advantages of this simplicity have mainly to do with the existence of algorithms for performing the calculation efficiently (in a transform domain) for the entire set of offsets. The disadvantages have to do with

Template	*Image*	*Correlation*
1 1 1	1 1 0 0 0	7 4 2 x x
1 1 1	1 1 1 0 0	5 3 2 x x
1 1 1	1 0 1 0 0	2 1 9 x x
	0 0 0 0 0	x x x x x
	0 0 0 0 8	x x x x x
		x = undefined

Fig. 3.4 (a) A simple template. (b) An image with noise. (c) The aperiodic correlation array of the template and image. Ideally peaks in the correlation indicate positions of good match. Here the correlation is only calculated for offsets that leave the template entirely within the image. The correct peak is the upper left one at 0, 0 offset. The "false alarm" at offset 2, 2 is caused by the bright "noise point" in the lower right of the image.

the fact that the metric is sensitive to properties of the image that may vary with the offset, such as its average brightness. Slight changes in the shape of the object, its size, orientation, or intensity values can also disturb the match.

Nonetheless, the idea of template matching is important, particularly if Eq. (3.3) is viewed as a *filtering* operation instead of an algorithm that does all the work of object detection. With this viewpoint one chooses one or more templates (filters) that transform the image so that certain features of an object are more readily apparent. These templates generally highlight subparts of the objects. One such class of templates is edge templates (discussed in detail in Section 3.3).

We showed in Section 2.2.4 that convolution and multiplication are Fourier transform pairs. Now note that the correlation operation in (3.3) is essentially the same as a convolution with a function $t'(\mathbf{x}) \equiv t(-\mathbf{x})$. Thus in a mathematical sense cross correlation and convolution are equivalent. Consequently, if the size of the template is sufficiently large, it is cheaper to perform the template matching operation in the spatial frequency domain, by the same transform techniques as for filtering.

Normalized Correlation

A crucial assumption in the development of Eq. (3.3) was that the image energy covered by the matching template at any offset was constant; this leads to a linear correlation matching technique. This assumption is approximately correct if the average image intensity varies slowly compared to the template size, but a bright spot in the image can heavily influence the correlation by affecting the sum of products violently in a small area (Fig. 3.4). Even if the image is well behaved, the range of values of the metric can vary with the size of the matching template. Are there ways of normalizing the correlation metric to make it insensitive to these variations?

There is a well-known treatment of the normalized correlation operation. It has been used for a variety of tasks involving registration and stereopsis of images [Quam and Hannah 1974]. Let us say that two input images are being matched to find the best offset that aligns them.

Let $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ be the images to be matched. q_2 is the patch of f_2 (possibly all of it) that is to be matched with a similar-sized patch of f_1. q_1 is the patch of f_1 that is covered by q_2 when q_2 is offset by $\mathbf{y}$.

Let $E()$ be the expectation operator. Then

$$\sigma(q_1) = [E(q_1^2) - (E(q_1))^2]^{1/2} \tag{3.4}$$

$$\sigma(q_2) = [E(q_2^2) - (E(q_2))^2]^{1/2} \tag{3.5}$$

give the standard deviations of points in patches q_1 and q_2. (For notational convenience, we have dropped the spatial arguments of q_1 and q_2.) Finally, the normalized correlation is

$$N(\mathbf{y}) = \frac{E(q_1 q_2) - E(q_1)E(q_2)}{\sigma(q_1)\sigma(q_2)} \tag{3.6}$$

and $E(q_1 q_2)$ is the expected value of the product of intensities of points that are superimposed by the translation by $\mathbf{y}$.

The normalized correlation metric is less dependent on the local properties of the reference and input images than is the unnormalized correlation, but it is sensitive to the signal-to-noise content of the images. High uncorrelated noise in the two images, or the image and the reference, decreases the value of the correlation. As a result, one should exercise some care in interpreting the metric. If the noise properties of the image are known, one indication of reliability is given by the "(signal + noise)-to-noise" ratio. For the normalized correlation to be useful, the standard deviation of the patches of images to be matched (i.e., of the areas of image including noise) should be significantly greater than that of the noise. Then a correlation value may be considered significant if it is approximately equal to the theoretically expected one. Consider uncorrelated noise of identical standard deviation, in a patch of true value $f(x, y)$. Let the noise component of the image be $n(x, y)$. Then the theoretical maximum correlation is

$$1 - \frac{\sigma^2(n)}{\sigma^2(f+n)} \tag{3.7}$$

In matching an idealized, noise-free reference pattern, the best expected value of the cross correlation is

$$\frac{\sigma(f)}{\sigma(f+n)} \tag{3.8}$$

If the noise and signal characteristics of the data are known, the patch size may be optimized by using that information and the simple statistical arguments above. However, such considerations leave out the effects of systematic, nonstatistical error (such as imaging distortions, rotations, and scale differences between images). These systematic errors grow with patch size, and may swamp the statistical advantages of large patches. In the worst case, they may vitiate the advantages of the correlation process altogether.

Since correlation is expensive, it is advantageous to ensure that there is enough information in the patches chosen for correlation before the operation is done. One way to do this is to apply a cheap "interest operator" before the relatively expensive correlation. The idea here is to make sure that the image varies enough to give a usable correlation image. If the image is of uniform intensity, even its correlation with itself (autocorrelation) is flat everywhere, and no information about where the image is registered with itself is derivable. The "interest operator" is a way of finding areas of image with high variance. In fact, a common and useful interest measure is exactly the (directional) variance over small areas of image. One directional variance algorithm works as follows.

The Moravec interest operator [Moravec 1977] produces candidate match points by measuring the distinctness of a local piece of the image from its surround. To explain the operator, we first define a variance measure at a pixel $(\mathbf{x})$ as

$$\text{var}(x, y) = \left\{ \sum_{k, l \text{ in } s} [f(x, y) - f(x + k, y + l)]^2 \right\}^{1/2} \tag{3.9}$$

$$s = \{(0, a), (0, -a), (a, 0), (-a, 0)\}$$

where a is a parameter. Now the interest operator value is initially the minimum of itself and surrounding points:

$$\text{IntOpVal}(\mathbf{x}) := \min_{y<1} [\text{var}(\mathbf{x} + \mathbf{y})] \tag{3.10}$$

Next a check is made to see if the operator is a local maximum by checking neighbors again. Only local maxima are kept.

$$\begin{aligned} &\text{IntOpVal}(\mathbf{x}) := 0 \text{ if} \\ &\text{IntOpVal}(\mathbf{x}) \geqslant \text{IntOpVal}(\mathbf{x} + \mathbf{y}) \\ &\qquad \text{for } \mathbf{y} \leqslant 1 \end{aligned} \tag{3.11}$$

Finally, candidate points are chosen from the IntOpVal array by thresholding.

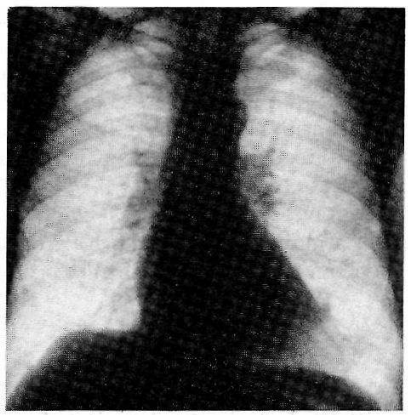
$$\mathbf{x} \text{ is a candidate point iff IntOpVal } (\mathbf{x}) > T \tag{3.12}$$

The threshold is chosen empirically to produce some fraction of the total image points.

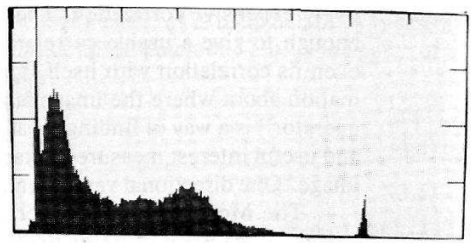
3.2.2 Histogram Transformations

A gray-level histogram of an image is a function that gives the frequency of occurrence of each gray level in the image. Where the gray levels are quantized from 0 to n, the value of the histogram at a particular gray level p, denoted $h(p)$, is the number or fraction of pixels in the image with that gray level. Figure 3.5 shows an image with its histogram.

A histogram is useful in many different ways. In this section we consider the histogram as a tool to guide gray-level transformation algorithms that are akin to filtering. A very useful image transform is called *histogram equalization.* Histogram equalization defines a mapping of gray levels p into gray levels q such that the distribution of gray levels q is uniform. This mapping stretches contrast (expands the



(a)



(b)

Fig. 3.5 (a) An image. (b) Its intensity histogram.

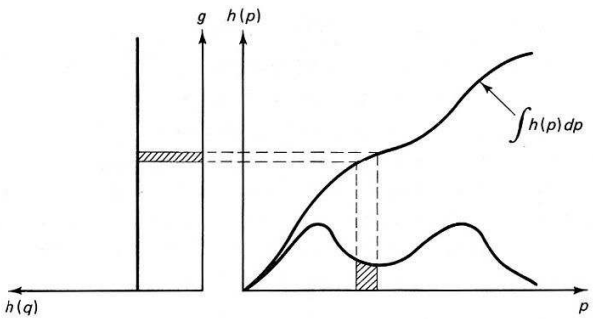
range of gray levels) for gray levels near histogram maxima and compresses contrast in areas with gray levels near histogram minima. Since contrast is expanded for most of the image pixels, the transformation usually improves the detectability of many image features.

The histogram equalization mapping may be defined in terms of the *cumulative* histogram for the image. To see this, consider Fig. 3.6a. To map a small interval of gray levels dp onto an interval dq in the general case, it must be true that

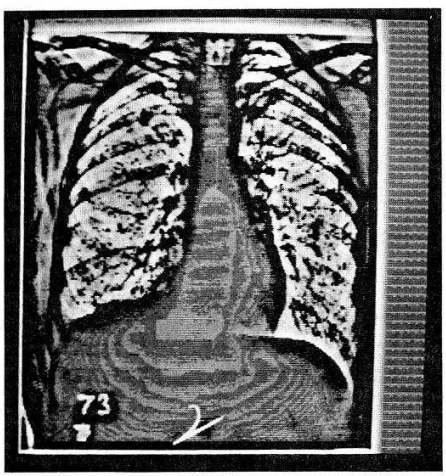
$$g(q)\,dq = h(p)\,dp \tag{3.13}$$

where $g(q)$ is the new histogram. If, in the histogram equalization case, $g(q)$ is to be uniform, then

$$g(q_2) = \frac{N^2}{M} \tag{3.14}$$



(a)



(b)

Fig. 3.6 (a) Basis for a histogram equalization technique. (b) Results of histogram equalization.

where N^2 is the number of pixels in the image and M is the number of gray levels. Thus combining Eqs. (3.13) and (3.14) and integrating, we have

$$g(q) = \frac{M}{N^2}\int_0^q h(p)\,dp \tag{3.15}$$

But Eq. (3.15) is simply the equation for the normalized cumulative histogram. Figure 3.6b shows the histogram-equalized image.

3.2.3 Background Subtraction

Background subtraction can be another important filtering step in early processing. Many images can have slowly varying background gray levels which are incidental to the task at hand. Examples of such variations are:

- Solution gradients in cell slides
- Lighting variations on surfaces in office scenes
- Lung images in a chest radiograph

Note that the last example is only a "background" in the context of looking for some smaller variations such as tumors or pneumoconiosis.

Background subtraction attempts to remove these variations by first approximating them (perhaps analytically) with a background image f_b and then subtracting this approximation from the original image. That is, the new image f_n is

$$f_n(\mathbf{x}) = f(\mathbf{x}) - f_b(\mathbf{x}) \tag{3.16}$$

Various functional forms have been tried for analytic representations of slowly varying backgrounds. In the simplest cases, $f_b(x)$ may be a constant,

$$f_b(\mathbf{x}) = c \tag{3.17}$$

or linear,

$$f_b(\mathbf{x}) = \mathbf{m}\cdot\mathbf{x} + c \tag{3.18}$$

A more sophisticated background model is to use a low-pass filtered variant of the original image:

$$f_b(\mathbf{x}) = \mathfrak{F}^{-1}[H(\mathbf{u})\,F(\mathbf{u})] \tag{3.19}$$

where $H(\mathbf{u})$ is a low-pass filtering function. The problem with this technique is that it is global; one cannot count on the "best" effect in any local area since the filter treats all parts of the image identically. For the same reason, it is difficult to design a Fourier filter that works for a number of very different images.

A workable alternative is to approximate $f_b(\mathbf{x})$, using *splines*, which are piecewise polynomial approximation functions. The mathematics of splines is treated in Chapter 8 since they find more general application as representations of shape. The filtering application is important but specialized. The attractive feature of a spline approximation for filtering is that it is *variation diminishing* and *spatially variant*. The spline approximation is guaranteed to be "smoother" than the origi-

nal function and will approximate the background differently in different parts of the image. The latter feature distinguishes the method from Fourier-domain techniques which are spatially invariant. Figure 3.7 shows the results of spline filtering.

3.2.4 Filtering and Reflectance Models

Leaving the effects of imaging geometry implicit (Section 2.2.2), the definitions in Section 2.2.3 imply that the image irradiance (gray level) at the image point x' is proportional to the product of the scene irradiance E and the *reflectance* r at its corresponding world point $\mathbf{x}$.

$$f(\mathbf{x}') = E(\mathbf{x})r(\mathbf{x}) \tag{3.20}$$

The irradiance at x is the sum of contributions from all illumination sources, and the reflectance is that portion of the irradiance which is reflected toward the observer (camera). Usually E changes slowly over a scene, whereas r changes quickly over edges, due to varying face angles, paint, and so forth. In many cases one would like to detect these changes in r while ignoring changes in E. One way of doing this is to filter the image $f(\mathbf{x}')$ to eliminate the slowly varying component. However, as f is the *product* of illumination and reflectance, it is difficult to define an operation that selectively diminishes E while retaining r. Furthermore, such an operation must retain the positivity of f. One solution is to take the logarithm of Eq. (3.20). Then

$$\log f = \log E + \log r \tag{3.21}$$

Equation (3.21) shows two desirable properties of the logarithmic transformation: (1) the logarithmic image is positive in sign, and (2) the image is a superposition of the irradiance component and reflectance component. Since reflectance is an in-

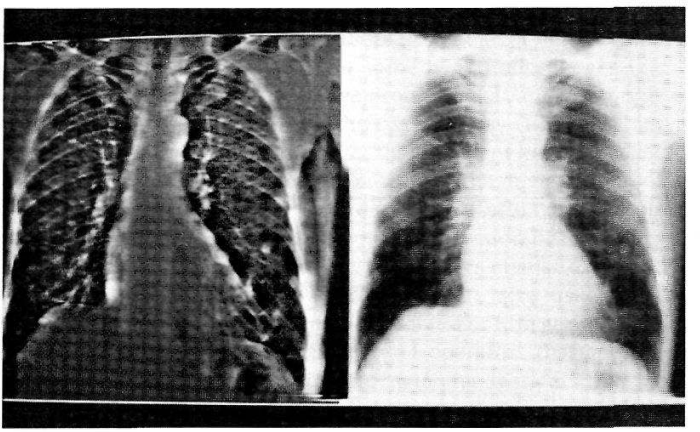


Fig. 3.7 The results of spline filtering to remove background variation.

trinsic characteristic of objects, the obvious goal of image analysis is to recognize the reflectance component under various conditions of illumination. Since the separation of two components is preserved under linear transformations and the irradiance component is usually of low spatial frequency compared to the reflectance component, filtering techniques can suppress the irradiance component of the signal relative to the reflectance component.

If the changes in r occur over very short distances in the images, r may be isolated by a three-step process [Horn 1974]. First, to enhance reflectance changes, the image function is differentiated (Section 3.3.1). The second step removes the low irradiance gradients by thresholding. Finally, the resultant image is integrated to obtain an image of perceived "lightness" or reflectance. Figure 3.8 shows these steps for the one-dimensional case.

A basic film parameter is density, which is proportional to the logarithm of transmitted intensity; the logarithmically transformed image is effectively a *density image.* In addition to facilitating the extraction of lightness, another advantage of the density image is that it is well matched to our visual experience. The ideas for many image analysis programs stem from our visual inspection of the image. However, the human visual system responds logarithmically to light intensity and also enhances high spatial frequencies [Stockham 1972]. Algorithms derived from

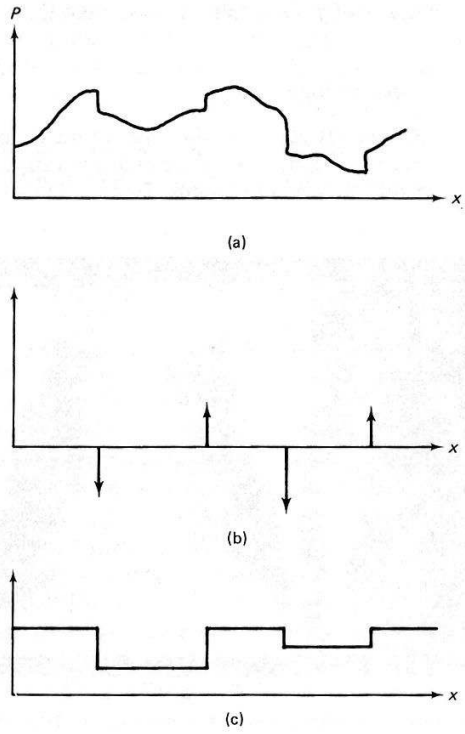


Fig. 3.8 Steps in processing an image to detect reflectance. (a) Original image. (b) Differentiation followed by thresholding. (c) Integration of function in (b).

introspective reasoning about the perceived image (which has been transformed by our visual system) will not necessarily be successful when applied to an unmodified intensity image. Thus one argument for using a density transformation followed by high spatial frequency emphasis filtering is that the computer is then "seeing" more like the human image analyzer.

3.3 FINDING LOCAL EDGES

Boundaries of objects tend to show up as intensity discontinuities in an image. Experiments with the human visual system show that boundaries in images are extremely important; often an object can be recognized from only a crude outline [Attneave 1954]. This fact provides the principal motivation for representing objects by their boundaries. Also, the boundary representation is easy to integrate into a large variety of object recognition algorithms.

One might expect that algorithms could be designed that find the boundaries of objects directly from the gray-level values in the image. But when the boundaries have complicated shapes, this is difficult. Much greater success has been obtained by first transforming the image into an intermediate image of *local* gray-level discontinuities, or edges, and then composing these into a more elaborate boundary. This strategy reflects the principle: When the gap between representations becomes too large, introduce intermediate representations. In this case, boundaries that are highly model-dependent may be decomposed into a series of local edges that are highly model-independent.

A local edge is a small area in the image where the local gray levels are changing rapidly in a simple (e.g., monotonic) way. An *edge operator* is a mathematical operator (or its computational equivalent) with a small spatial extent designed to detect the presence of a local edge in the image function.

It is difficult to specify a priori which local edges correspond to relevant boundaries in the image. Depending on the particular task domain, different local changes will be regarded as likely edges. Plots of gray level versus distance along the direction perpendicular to the edge for some hypothetical edges (Fig. 3.9a-e) demonstrate some different kinds of "edge profiles" that are commonly encountered. Of course, in most practical cases, the edge is noisy (Fig. 3.9d) and may appear as a composite of profile types. The fact that different kinds of edge operators perform best in different task domains has prompted the development of a variety of operators. However, the unifying feature of most useful edge operators is that they compute a *direction* which is aligned with the direction of maximal gray-level change, and a *magnitude* describing the severity of this change. Since edges are a high-spatial-frequency phenomenon, edge finders are also usually sensitive to high-frequency noise, such as "snow" on a TV screen or film grain.

Operators fall into three main classes: (1) operators that approximate the mathematical gradient operator, (2) template matching operators that use multiple templates at different orientations, and (3) operators that fit local intensities with parametric edge models. Representative examples from the first two of these categories appear in this section. The computer vision literature abounds with edge

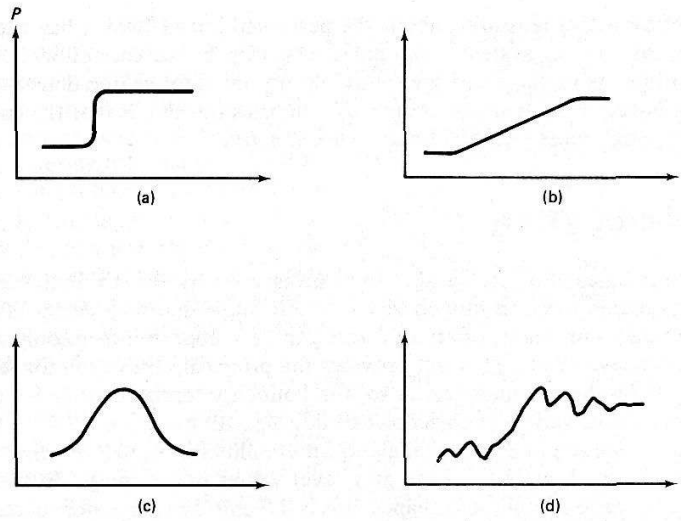


Fig. 3.9 Edge profiles.

operators, and we make no attempt to summarize them all here. For a guide to this literature, see [Rosenfeld and Kak 1976].

Parametric models generally capture more detailed edge structure than the two-parameter direction and magnitude vector; as a result, they can be more computationally complicated. For this reason and others discussed in Section 3.3.4, we shall omit a detailed discussion of these kinds of edge operators. One of the best known parametric models is Hueckel's [Hueckel 1971, 1973], but several others have been developed since [Mero and Vassy 1975; Nevatia 1977; Abdou 1978; Tretiak 1979].

3.3.1 Types of Edge Operators

Gradient and Laplacian

The most common and historically earliest edge operator is the gradient [Roberts 1965]. For an image function $f(\mathbf{x})$, the gradient magnitude $s(\mathbf{x})$ and direction $\phi(\mathbf{x})$ can be computed as

$$s(\mathbf{x}) = (\Delta_1^2 + \Delta_2^2)^{1/2} \tag{3.22}$$

$$\phi(\mathbf{x}) = \operatorname{atan}(\Delta_2, \Delta_1) \tag{3.23}$$

where

$$\Delta_1 = f(x + n, y) - f(x, y) \tag{3.24}$$

$$\Delta_2 = f(x, y + n) - f(x, y)$$

n is a small integer, usually unity, and atan (x, y) returns $\tan^{-1}(x/y)$ adjusted to the proper quadrant. The parameter n is called the "span" of the gradient. Roughly, n should be small enough so that the gradient is a good approximation to the local changes in the image function, yet large enough to overcome the effects of small variations in f.

Equation (3.24) is only one *difference operator*, or way of measuring gray-level intensities along orthogonal directions using Δ_1 and Δ_2. Figure 3.10 shows the gradient difference operators compared to other operators [Roberts 1965; Prewitt 1970]. The reason for the modified operators of Prewitt and Sobel is that the local averaging tends to reduce the effects of noise. These operators do, in fact, perform better than the Roberts operator for a step edge model.

One way to study an edge operator's performance is to use an ideal edge such as the step edge shown in Fig. 3.11. This edge has two gray levels: zero and h units. If the edge goes through the finite area associated with a pixel, the pixel is given a value between zero and h, depending on the proportion of its area covered. Comparative edge operator performance has been carried out [Abdou 1978]. In the case of the Sobel operator (Fig. 3.10c) the measured orientation ϕ' is given by

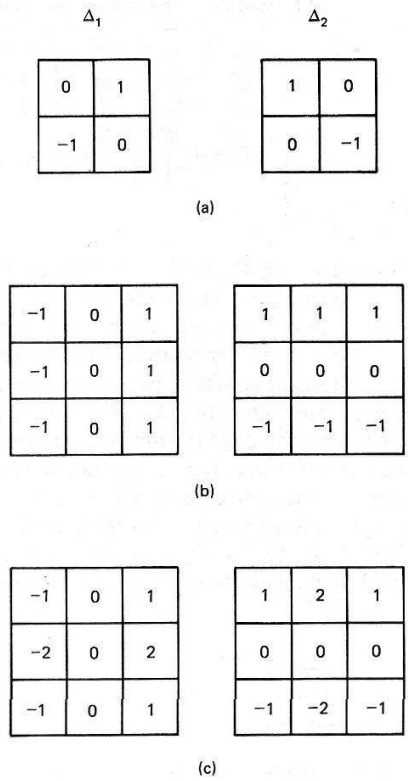


Fig. 3.10 Gradient operators.

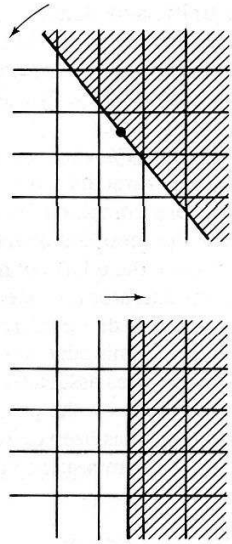


Fig. 3.11 Edge models for orientation and displacement sensitivity analyses.

$$\phi' = \begin{cases} \phi & \text{if } 0 \leqslant \phi \leqslant \tan^{-1}\left(\frac{1}{3}\right) \\ \tan^{-1}\left(\dfrac{7\tan^2\phi + 6\tan\phi - 1}{-9\tan^2\phi + 22\tan\phi - 1}\right) & \text{if } \tan^{-1}\left(\frac{1}{3}\right) \leqslant 0 \leqslant \phi \leqslant \pi/4 \end{cases} \quad (3.25)$$

Arguments from symmetry show that only the $0 \leqslant \phi < \pi/4$ cases need be examined. Similar studies could be made using ramp edge models.

A rather specialized kind of gradient is that taken "between pixels." This scheme is shown in Fig. 3.12. Here a pixel may be thought of as having four *crack edges* surrounding it, whose directions of are fixed by the pixel to be multiples of $\pi/2$. The magnitude of the edge is determined by $|f(\mathbf{x}) - f(\mathbf{y})|$, where $\mathbf{x}$ and $\mathbf{y}$ are the coordinates of the pixels that have the edge in common. One advantage of this formulation is that it provides an effective way of separating regions and their boundaries. The disadvantage is that the edge orientation is crude.

The *Laplacian* is an edge detection operator that is an approximation to the mathematical Laplacian $\partial^2 f/\partial x^2 + \partial^2 f/\partial y^2$ in the same way that the gradient is an approximation to the first partial derivatives. One version of the discrete Laplacian is given by

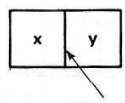


Fig. 3.12 "Crack" edge representation.

$$L(x, y) = f(x, y) - \tfrac{1}{4}[f(x, y+1) + f(x, y-1) + f(x+1, y) + f(x-1, y)] \tag{3.26}$$

The Laplacian has two disadvantages as an edge measure: (1) useful directional information is not available, and (2) the Laplacian, being an approximation to the second derivative, doubly enhances any noise in the image. Because of these disadvantages, the Laplacian has fallen into disuse, although some authors have used it as an adjunct to the gradient [Wechsler and Sklansky 1977; Akatsuka 1974] in the following manner: There is an edge at $\mathbf{x}$ with magnitude $g(\mathbf{x})$ and direction $\phi(\mathbf{x})$ if $g(\mathbf{x}) > T_1$ and $L(\mathbf{x}) > T_2$.

Edge Templates

The *Kirsch operator* [Kirsch 1971] is related to the edge gradient and is given by

$$S(\mathbf{x}) = \max\left[1, \max_k \sum_{k-1}^{k+1} f(\mathbf{x}_k)\right] \tag{3.27}$$

where $f(\mathbf{x}_k)$ are the eight neighboring pixels to $\mathbf{x}$ and where subscripts are computed modulo 8. A 3-bit direction can also be extracted from the value of k that yields the maximum in (3.27). In practice, "pure" template matching has replaced the use of (3.27). Four separate templates are matched with the image and the operator reports the magnitude and direction associated with the maximum match. As one might expect, the operator is sensitive to the magnitude of $f(\mathbf{x})$, so that in practice variants using large templates are generally used. Figure 3.13 shows Kirsch-motivated templates with different spans.

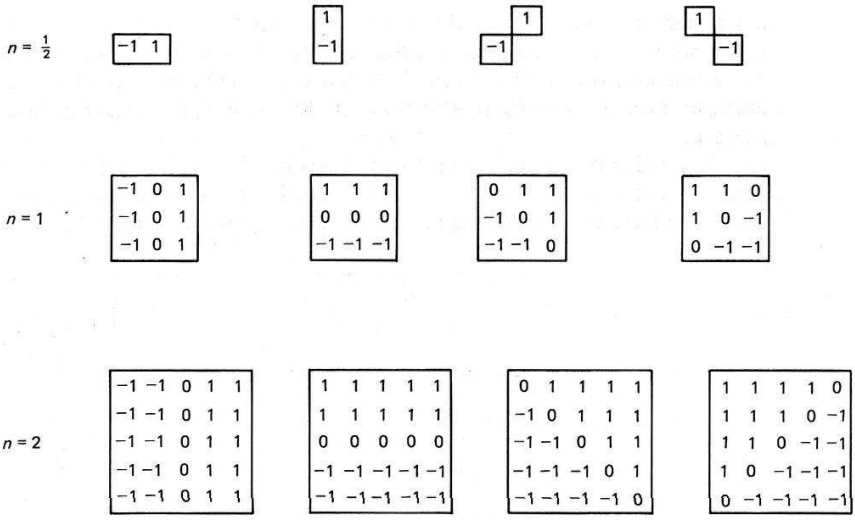


Fig. 3.13 Kirsch templates.

This brief discussion of edge templates should not be construed as a comment on their appropriateness or popularity. In fact, they are widely used, and the template-matching concept is the essence of the other approaches. There is also evidence that the mammalian visual system responds to edges through special low-level template-matching edge detectors [Hubel and Wiesel 1979].

3.3.2 Edge Thresholding Strategies

For most images there will be but few places where the gradient magnitude is equal to zero. Furthermore, in the absence of any special context, small magnitudes are most likely to be due to random fluctuations, or noise in the image function f. Thus in practical cases one may use the expedient of only reporting an edge element at $\mathbf{x}$ if $g(\mathbf{x})$ is greater than some threshold, in order to reduce these noise effects.

This strategy is computationally efficient but may not be the best. An alternative thresholding strategy [Frei and Chen 1977] views difference operators as part of a set of orthogonal basis functions analogous to the Fourier basis of Section 2.2.4. Figure 3.14 shows the nine Frei–Chen basis functions. Using this basis, the image near a point $\mathbf{x}_0$ can be represented as

$$f(\mathbf{x}) = \sum_{k=1}^{8} (f, h_k) h_k(\mathbf{x} - \mathbf{x}_0) / (h_k, h_k) \tag{3.28}$$

where the (f, h_k) is the correlation operation given by

$$(f, h_k) = \sum_{D} f(\mathbf{x}_0) h_k(\mathbf{x} - \mathbf{x}_0) \tag{3.29}$$

and D is the nonzero domain of the basis functions. This operation is also regarded as the *projection* of the image into the basis function h_k. When the image can be reconstructed from the basis functions and their coefficients, the basis functions span the space. In the case of a smaller set of functions, the basis functions span a subspace.

The value of a projection into any basis function is highest when the image function is identical to the basis function. Thus one way of measuring the "edgeness" of a local area in an image is to measure the relative projection of the image

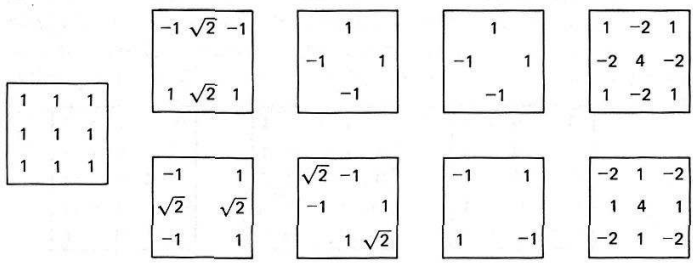


Fig. 3.14 Frei-Chen orthogonal basis.

into the edge basis functions. The relative projection into the particular "edge subspace" is given by

$$\cos\theta = \left(\frac{E}{S}\right)^{1/2} \tag{3.30}$$

where

$$E = \sum_{k=1}^{2} (f, h_k)^2$$

and

$$S = \sum_{k=0}^{8} (f, h_k)^2$$

Thus if $\theta < T$, report an edge; otherwise, not. Figure 3.15 shows the potential advantage of this technique compared to the technique of thresholding the gradient magnitude, using two hypothetical projections B_1 and B_2. Even though B_2 has a small magnitude, its relative projection into edge subspace is large and thus would be counted as an edge with the Frei–Chen criterion. This is not true for B_1.

Under many circumstances it is appropriate to use model information about the image edges. This information can affect the way the edges are interpreted after they have been computed or it may affect the computation process itself. As an example of the first case, one may still use a gradient operator, but vary the threshold for reporting an edge. Many versions of the second, more extreme strategies of using special spatially variant detection methods have been tried [Pingle and Tenenbaum 1971; Griffith 1973; Shirai 1975]. The basic idea is illustrated in Fig. 3.16. Knowledge of the orientation of an edge allows a special orientation-sensitive operator to be brought to bear on it.

3.3.3 Three-Dimensional Edge Operators

In many imaging applications, particularly medicine, the images are three-dimensional. Consider the examples of the reconstructed planes described in Sections 1.1 and 2.3.4. The medical scanner that acquires these data follows several parallel image planes, effectively producing a three-dimensional volume of data.

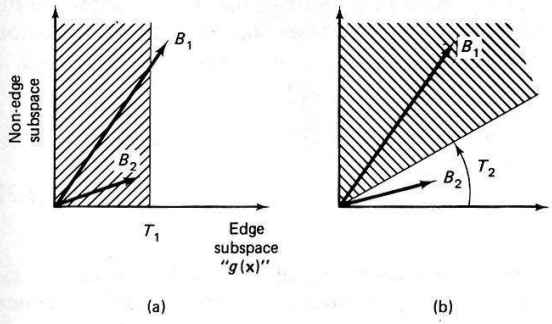
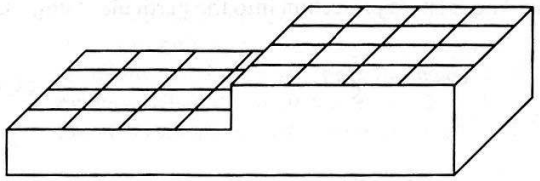
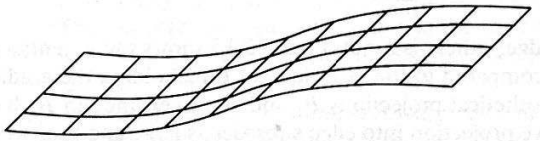


Fig. 3.15 Comparison of thresholding techniques.



(a)



(b)

Fig. 3.16 Model-directed edge detection.

In three-dimensional data, boundaries of objects are surfaces. Edge elements in two dimensions become surface elements in three dimensions. The two-dimensional image gradient, when generalized to three dimensions, is the local surface normal. Just as in the two-dimensional case, many different basis operators can be used [Liu 1977; Zucker and Hummel 1979]. That of Zucker and Hummel uses an optimal basis assuming an underlying continuous model. We shall just describe the operator here; the proof of its correctness given the continuous image model may be found in the reference. The basis functions for the three-dimensional operator are given by

$$g_1(x, y, z) = \frac{x}{r} \tag{3.31}$$

$$g_2(x, y, z) = \frac{y}{r}$$

$$g_3(x, y, z) = \frac{z}{r}$$

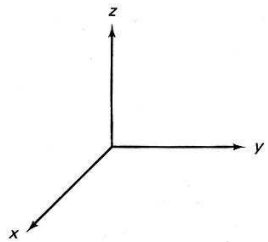
where $r = (x^2 + y^2 + z^2)^{1/2}$. The discrete form of these operators is shown in Fig. 3.17 for a $3 \times 3 \times 3$ pixel domain D. Only g_1 is shown since the others are obvious by symmetry. To apply the operator at a point x_0, y_0, z_0 compute projections a, b, and c, where

$$a = (g_1, f) = \sum_D g_1(\mathbf{x}) f(\mathbf{x}-\mathbf{x}_0)$$

$$b = (g_2, f) \tag{3.32}$$

$$c = (g_3, f)$$

The result of these computations is the surface normal $\boldsymbol{n} = (a, b, c)$ at (x_0, y_0, z_0). Surface thresholding is analogous to edge thresholding: Report a surface element



$-\frac{\sqrt{3}}{3}$	$-\frac{\sqrt{2}}{2}$	$-\frac{\sqrt{3}}{3}$
$-\frac{\sqrt{2}}{2}$	-1	$-\frac{\sqrt{2}}{2}$
$-\frac{\sqrt{3}}{3}$	$-\frac{\sqrt{2}}{2}$	$-\frac{\sqrt{3}}{3}$

0	0	0
0	0	0
0	0	0

$\frac{\sqrt{3}}{3}$	$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{3}}{3}$
$\frac{\sqrt{2}}{2}$	1	$\frac{\sqrt{2}}{2}$
$\frac{\sqrt{3}}{3}$	$\frac{\sqrt{2}}{2}$	$\frac{\sqrt{3}}{3}$

Fig. 3.17 The $3 \times 3 \times 3$ edge basis function $g_1(x, y, z)$.

only if $s(x, y, z) = |\boldsymbol{n}|$ exceeds some threshold. Figure 3.18 shows the results of applying the operator to a synthetic three-dimensional image of a torus. The display shows small detected surface patches.

3.3.4 How Good are Edge Operators?

The plethora of edge operators is very difficult to compare and evaluate. For example, some operators may find most edges but also respond to noise; others may be

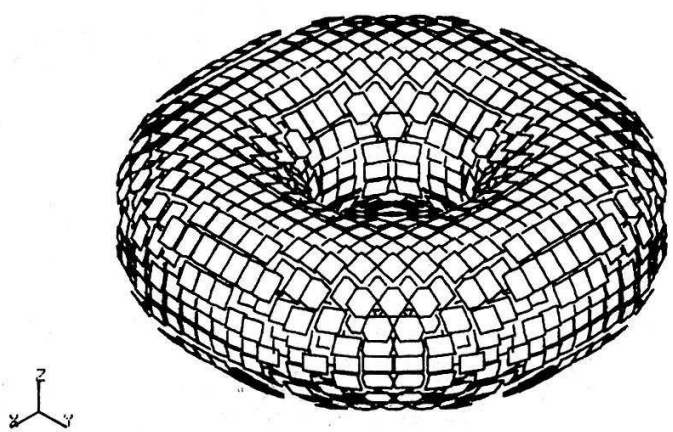


Fig. 3.18 Results of applying the Zucker-Hummel 3-D operator to synthetic image data in the shape of a torus.

noise-insensitive but miss some crucial edges. The following figure of merit [Pratt 1978] may be used to compare edge operators:

$$F = \frac{1}{\max(N_A, N_I)} \sum_{i=1}^{N_A} \frac{1}{1 + (ad_i^2)} \tag{3.33}$$

where N_A and N_I represent the number of actual and ideal edge points, respectively, a is a scaling constant, and d is the signed separation distance of an actual edge point normal to a line of ideal edge points. The term ad_i^2 penalizes detected edges which are offset from their true position; the penalty can be adjusted via a. Using this measure, all operators have surprisingly similar behaviors. Unsurprisingly, the performance of each deteriorates in the presence of noise [Abdou 1978]. (Pratt defines a signal-to-noise ratio as the square of the step edge amplitude divided by the standard deviation of Gaussian white noise.) Figure 3.19 shows some typical curves for different operators. To make this figure, the threshold for reporting an edge was chosen independently for each operator so as to maximize Eq. (3.33).

These comparisons are important as they provide a gross measure of differences in performance of operators even though each operator embodies a specific edge model and may be best in special circumstances. But perhaps the more important point is that since all real-world images have significant amounts of noise, all edge operators will generally produce imperfect results. This means that in considering the overall computer vision problem, that of building descriptions of objects, the efforts are usually best spent in developing methods that can use or improve the measurements from unreliable edges rather than in a search for the ideal edge detector.

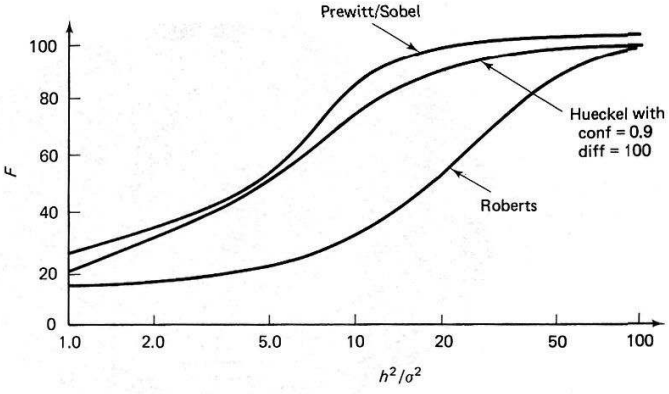


Fig. 3.19 Edge operator performance using Pratt's measure (Eq. 3.33).

3.3.5 Edge Relaxation

One way to improve edge operator measurements is to adjust them based on measurements of neighboring edges. This is a natural thing to want to do: If a weak horizontal edge is positioned between two strong horizontal edges, it should gain credibility. The edges can be adjusted based on local information using parallel-iterative techniques. This sort of process is related to more global analysis and is complementary to sequential approaches such as edge tracking (Chapter 4).

Early cooperative edge detection techniques used pairwise measurements between pixels [Zucker et al. 1977]. A later version [Prager 1980] allows for more complicated adjustment formulas. In describing the edge relaxation scheme, we essentially follow Prager's development and use the crack edges described at the end of the discussion on gradients (Sec. 3.31). The development can be extended to the other kinds of edges and the reader is invited to do just this in the Exercises.

The overall strategy is to recognize local edge patterns which cause the confidence in an edge to be modified. Prager recognizes three groups of patterns: patterns where the confidence of an edge can be increased, decreased, or left the same. The overall structure of the algorithm is as follows:

Algorithm 3.1 Edge Relaxation

0. Compute the initial confidence of each edge $C^0(e)$ as the normalized gradient magnitude normalized by the maximum gradient magnitude in the image.

1. $k = 1$;

2. Compute each edge type based on the confidence of edge neighbors;

3. Modify the confidence of each edge $C^k(e)$ based on its edge type and its previous confidence $C^{k-1}(e)$;

4. Test the $C^k(e)$'s to see if they have all converged to either 0 or 1. If so, stop; else, increment k and go to 2.

The two important parts of the algorithm are step 2, computing the edge type, and step 3, modifying the edge confidence.

The edge-type classification relies on the notation for edges (Fig. 3.20). The edge type is a concatenation of the left and right vertex types. Vertex types are computed from the strength of edges emanating from a vertex. Vertical edges are handled in the same way, exploiting the obvious symmetries with the horizontal case. Besides the central edge e, the left vertex is the end point for three other possible edges. Classifying these possible edges into "edge" and "no-edge" provides the underpinnings for the vertex types in Fig. 3.21.

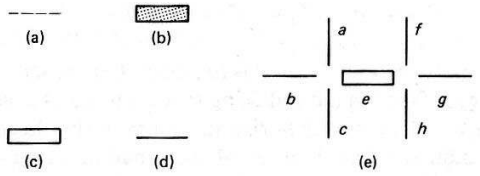


Fig. 3.20 Edge notation. (a) Edge position with no edge. (b) Edge position with edge. (c) Edge to be updated. (d) Edge of unknown strength. (e) Configuration of edges around a central edge e.

To compute vertex type, choose the maximum confidence vertex, i.e., the vertex is type j where j maximizes conf(j)

and

$$\begin{aligned}\text{conf}(0) &= (m-a)(m-b)(m-c)\\ \text{conf}(1) &= a(m-b)(m-c)\\ \text{conf}(2) &= ab(m-c)\\ \text{conf}(3) &= abc\end{aligned}$$

where

$m = \max(a, b, c, q)$
q is a constant (0.1 is about right)

and a, b, and c are the normalized gradient magnitudes for the three edges. Without loss of generality, $a \geqslant b \geqslant c$. The parameter m adjusts the vertex classification so that it is relative to the local maximum. Thus $(a, b, c) = (0.25, 0.01, 0.01)$ is a type 1 vertex. The parameter q forces weak vertices to type zero [e.g., (0.01, 0.001, 0.001) is type zero].

Once the vertex type has been computed, the edge type is simple. It is merely the concatenation of the two vertex types. That is, the edge type is (ij), where i and j are the vertex types. (From symmetry, only consider $i \geqslant j$.)

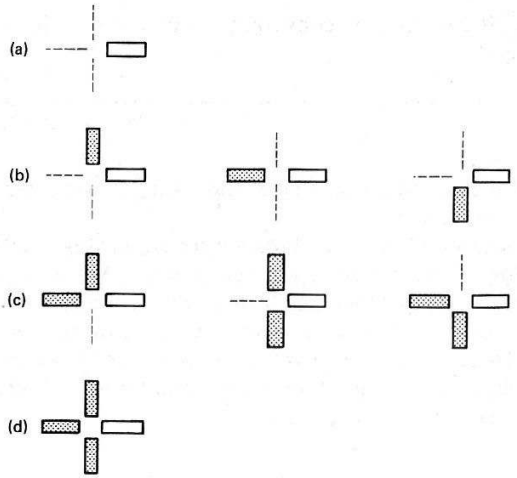


Fig. 3.21 Classification of vertex type of left-hand endpoint of edge e, Fig. 3.20.

Decisions in the second step of modifying edge confidence based on edge type appear in Table 3.1. The updating formula is:

increment: $C^{k+1}(e) = \min(1, C^k(e) + \delta)$
decrement: $C^{k+1}(e) = \max(0, C^k(e) - \delta)$
leave as is: $C^{k+1}(e) = C^k(e)$

where δ is a constant (values from 0.1 to 0.3 are appropriate). The result of using the relaxation scheme is shown in Fig. 3.22. The figures on the left-hand side show

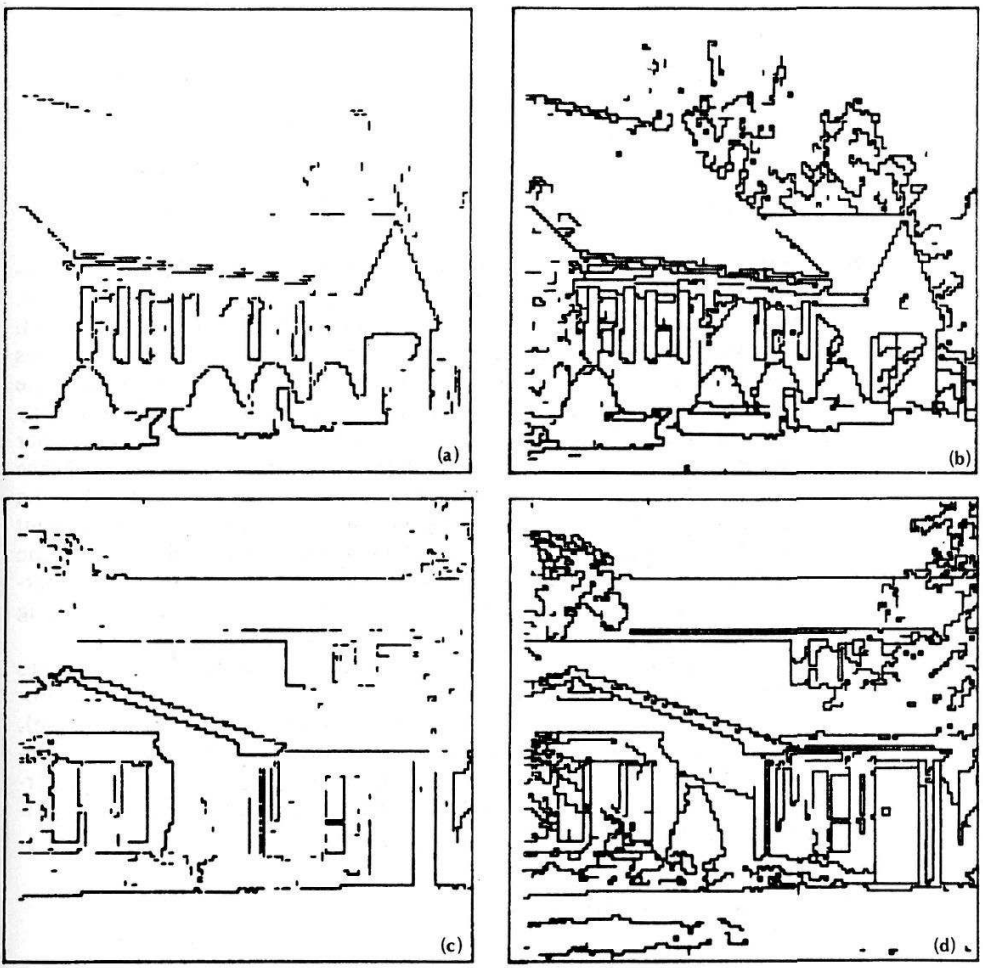


Fig. 3.22 Edge relaxation results. (a) Raw edge data. Edge strengths have been thresholded at 0.25 for display purposes only. (b) Results after five iterations of relaxation applied to (a). (c) Different version of (a). Edge strengths have been thresholded at 0.25 for display purposes only. (d) Results after five iterations of relaxation applied to (c).

the edges with normalized magnitudes greater than 0.25. Weak edges cause many gaps in the boundaries. The figures on the right side show the results of five iterations of edge relaxation. Here the confidence of the weak edges has been increased owing to the proximity of other edges, using the rules in Table 3.1.

Table 3.1

Decrement	*Increment*	*Leave as is*
0-0	*1-1*	*0-1*
0-2	*1-2*	*2-2*
0-3	*1-3*	*2-3*
		3-3

3.4 RANGE INFORMATION FROM GEOMETRY

Neither the perspective or orthogonal projection operations, which take the three-dimensional world to a two-dimensional image, is invertible in the usual sense. Since projection maps an infinite line onto a point in the image, information is lost. For a fixed viewpoint and direction, infinitely many continuous and discontinuous three-dimensional configurations of points could project on our retina in an image of, say, our grandmother. Simple cases are grandmothers of various sizes cleverly placed at varying distances so as to project onto the same area. An astronomer might imagine millions of points distributed perhaps through light-years of space which happen to line up into a "grandmother constellation." All that can be mathematically guaranteed by imaging geometry is that the image point corresponds to one of the infinite number of points on that three-dimensional line of sight. The "inverse perspective" transformation (Appendix 1) simply determines the equation of the infinite line of sight from the parameters of the imaging process modeled as a point projection.

However, a line and a plane not including it intersect in just one point. Lines of sight are easy to compute, and so it is possible to tell where any image point projects on to any known plane (the supporting ground or table plane is a favorite). Similarly, if two images from different viewpoints can be placed in correspondence, the intersection of the lines of sight from two matching image points determines a point in three-space. These simple observations are the basis of light-striping ranging (Section 2.3.3) and are important in stereo imaging.

3.4.1. Stereo Vision and Triangulation

One of the first ideas that occurs to one who wants to do three-dimensional sensing is the biologically motivated one of stereo vision. Two cameras, or one camera from two positions, can give relative depth or absolute three-dimensional location, depending on the elaboration of the processing and measurement. There has been

considerable effort in this direction [Moravec 1977; Quam and Hannah 1974; Binford 1971; Turner 1974; Shapira 1974]. The technique is conceptually simple:

1. Take two images separated by a baseline.
2. Identify points between the two images.
3. Use the inverse perspective transform (Appendix 1) or simple triangulation (Section 2.2.2) to derive the two lines on which the world point lies.
4. Intersect the lines.

The resulting point is in three-dimensional world coordinates.

The hardest part of this method is step 2, that of identifying corresponding points in the two images. One way of doing this is to use correlation, or template matching, as described in Section 3.2.1. The idea is to take a patch of one image and match it against the other image, finding the place of best match in the second image, and assigning a related "disparity" (the amount the patch has been displaced) to the patch.

Correlation is a relatively expensive operation, its naive implementation requiring $0(n^2m^2)$ multiplications and additions for an $m \times m$ patch and $n \times n$ image. This requirement can be drastically improved by capitalizing on the idea of variable resolution; the improved technique is described in Section 3.7.2.

Efficient correlation is of technological concern, but even if it were free and instantaneous, it would still be inadequate. The basic problems with correlation in stereo imaging have to do with the fact that things can look significantly different from different points of view. It is possible for the two stereo views to be sufficiently different that corresponding areas may not be matched correctly. Worse, in scenes with much obscuration, very important features of the scene may be present in only one view. This problem is alleviated by decreasing the baseline, but of course then the accuracy of depth determinations suffers; at a baseline length of zero there is no problem, but no stereo either. One solution is to identify world features, not image appearance, in the two views, and match those (the nose of a person, the corner of a cube). However, if three-dimensional information is sought as a help in perception, it is unreasonable to have to do perception first in order to do stereo.

3.4.2 A Relaxation Algorithm for Stereo

Human *stereopsis*, or fusing the inputs from the eyes into a stereo image, does not necessarily involve being aware of features to match in either view. Most human beings can fuse quite efficiently stereo pairs which individually consist of randomly placed dots, and thus can perceive three-dimensional shapes without recognizing monocular clues in either image. For example, consider the stereo pair of Fig. 3.23. In either frame by itself, nothing but a randomly speckled rectangle can be perceived. All the stereo information is present in the relative displacement of dots in the two rectangles. To make the right-hand member of the stereo pair, a patch of

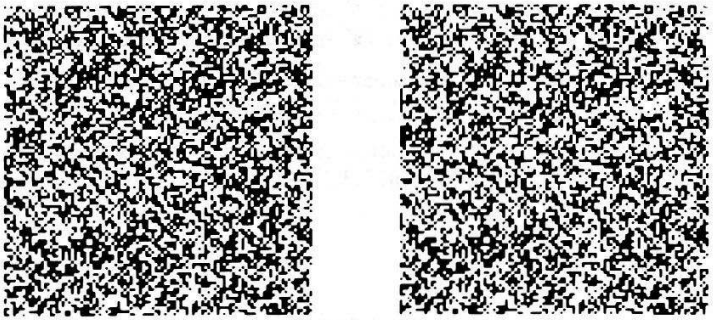


Fig. 3.23 A random-dot stereogram.

the randomly placed dots of the left-hand image is displaced sideways. The dots which are thus covered are lost, and the space left by displacing the patch is filled in with random dots.

Interestingly enough, a very simple algorithm [Marr and Poggio 1976] can be formulated that computes disparity from random dot stereograms. First consider the simpler problem of matching one-dimensional images of four points as depicted in Fig. 3.24. Although only one depth plane allows all four points to be placed in correspondence, lesser numbers of points can be matched in other planes.

The crux of the algorithm is the rules, which help determine, on a local basis, the appropriateness of a match. Two rules arise from the observation that most images are of opaque objects with smooth surfaces and depth discontinuities only at object boundaries:

1. Each point in an image may have only one depth value.
2. A point is almost sure to have a depth value near the values of its neighbors.

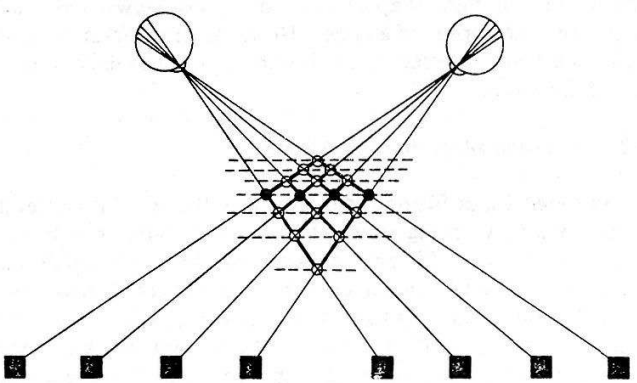


Fig. 3.24 The stereo matching problem.

Figure 3.24 can be viewed as a binary network where each possible match is represented by a binary state. Matches have value 1 and nonmatches value 0. Figure 3.25 shows an expanded version of Fig. 3.24. The connections of alternative matches for a point inhibit each other and connections between matches of equal depth reinforce each other. To extend this idea to two dimensions, use parallel arrays for different values of y where equal depth matches have reinforcing connections. Thus the extended array is modeled as the matrix $C(x, y, d)$ where the point x, y, d corresponds to a particular match between a point (x_1, y_1) in the right image and a point (x_2, y_2) in the left image. The stereopsis algorithm produces a series of matrices C_n which converges to the correct solution for most cases. The initial matrix $C_0(x, y, d)$ has values of one where x, y, d correspond to a match in the original data and has values of zero or otherwise.

Algorithm 3.2 [Marr and Poggio 1976]

Until C satisfies some convergence criterion, do

$$C_{n+1}(x, y, d) = \left\{ \sum_{x',y',d' \in S} C_n(x', y', d') - \sum_{x',y',d' \in \theta} C_n(x', y', d') + C_0(x, y, d) \right\} \quad (3.34)$$

where the term in braces is handled as follows:

$$\{t\} = \begin{cases} 1 & \text{if } t > T \\ 0 & \text{otherwise} \end{cases}$$

S = set of points x', y', d' such that $|\mathbf{x} - \mathbf{x}'| \leqslant 1$ and $d = d'$

θ = set of points x', y', d' such that $|\mathbf{x} - \mathbf{x}'| \leqslant 1$ and $|d - d'| = 1$

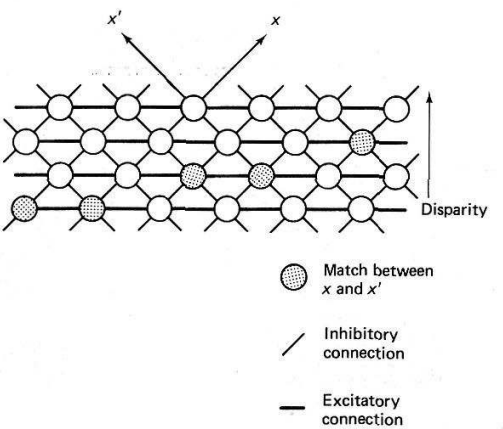


Fig. 3.25 Extension of stereo matching.

One convergence criterion is that the number of points modified on an iteration must be less than some threshold T. Fig. 3.26 shows the results of this computation; the disparity is encoded as a gray level and displayed as an image for different values of n.

A more general version of this algorithm matches image features such as edges rather than points (in the random-dot stereogram, the only features are

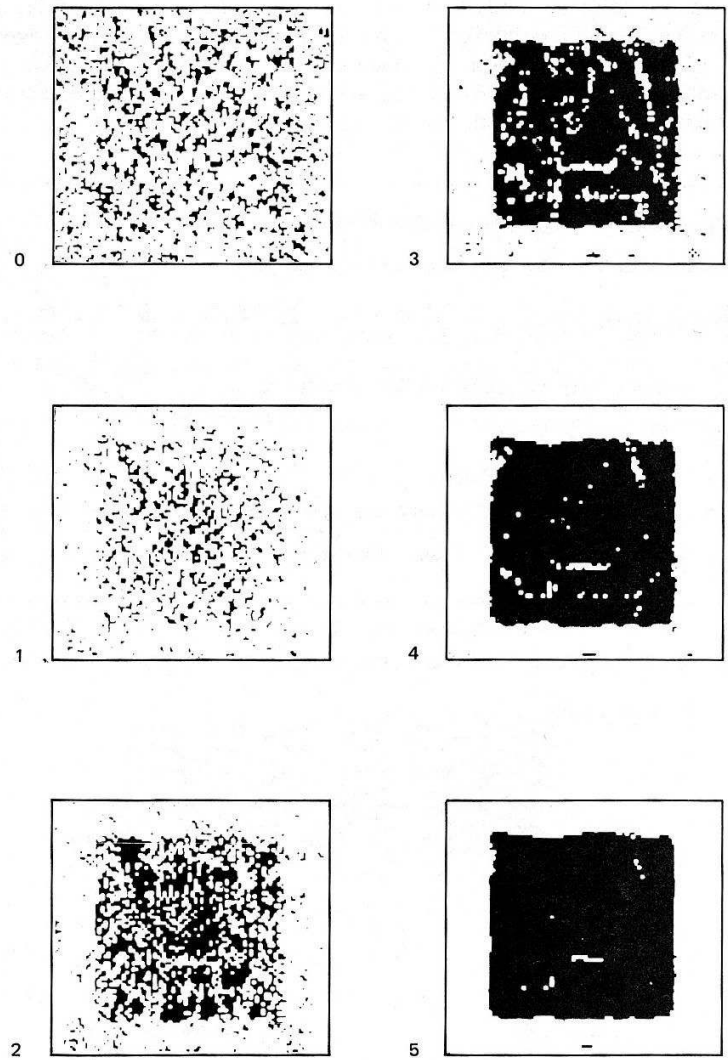


Fig. 3.26 The results of relaxation computations for stereo.

points), but the principles are the same. The extraction of features more complicated than edges or points is itself a thorny problem and the subject of Part II. It should be mentioned that Marr and Poggio have refined their stereopsis algorithm to agree better with psychological data [Marr and Poggio 1977].

3.5 SURFACE ORIENTATION FROM REFLECTANCE MODELS

The ordinary visual world is mostly composed of opaque three-dimensional objects. The intensity (gray level) of a pixel in a digital image is produced by the light reflected by a small area of surface near the corresponding point on the object.

It is easiest to get consistent shape (orientation) information from an image if the lighting and surface reflectance do not change from one scene location to another. Analytically, it is possible to treat such lighting as uniform illumination, a point source at infinity, or an infinite linear source. Practically, the human shape-from-shading transform is relatively robust. Of course, the perception of shape may be manipulated by changing the surface shading in calculated ways. In part, cosmetics work by changing the reflectivity properties of the skin and misdirecting our human shape-from-shading algorithms.

The recovery transformation to obtain information about surface orientation is possible if some information about the light source and the object's reflectivity is known. General algorithms to obtain and quantify this information are complicated but practical simplifications can be made [Horn 1975; Woodham 1978; Ikeuchi 1980]. The main complicating factor is that even with mathematically tractable object surface properties, a single image intensity does not uniquely define the surface orientation. We shall study two ways of overcoming this difficulty. The first algorithm uses intensity images as input and determines the surface orientation by using multiple light source positions to remove ambiguity in surface orientation. The second algorithm uses a single source but exploits constraints between neighboring surface elements. Such an algorithm assigns initial ranges of orientations to surface elements (actually to their corresponding image pixels) on the basis of intensity. The neighboring orientations are "relaxed" against each other until each converges to a unique orientation (Section 3.5.4).

3.5.1 Reflectivity Functions

For all these derivations, consider a distant point source of light impinging on a small patch of surface; several angles from this situation are important (Fig. 3.27).

A surface's reflectance is the fraction of a given incident energy flux (irradiance) it reflects in any given direction. Formally, the *reflectivity function* is defined as $r = \frac{dL}{dE}$, where L is exitant radiance and E is incident flux. In general, for anisotropic reflecting surfaces, the reflectivity function (hence L) is a function of all three angles i, e, and g. The quantity of interest to us is image irradiance, which is proportional to scene radiance, given by $L = \int r\, dE$. In general, the evaluation of this integral can be quite complicated, and the reader is referred to [Horn and

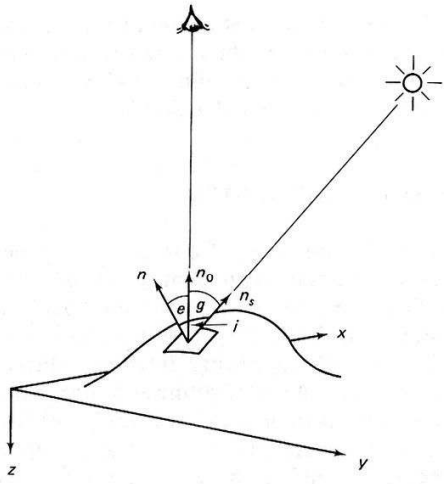


Fig. 3.27 Important reflectance angles: *i*, incidence; *e*, emittance; *g*, phase.

Sjoberg 1978] for a more detailed study. For our purposes we consider surfaces with simple reflectivity functions.

Lambertian surfaces, those with an ideal matte finish, have a very simple reflectivity function which is proportional only to the cosine of the incident angle. These surfaces have the property that under uniform or collimated illumination they look equally bright from any direction. This is because the amount of light reflected from a unit area goes down as the cosine of the viewing angle, but the amount of area seen in any solid angle goes up as the reciprocal of the cosine of the viewing angle. Thus the perceived intensity of a surface element is constant with respect to viewer position. Other surfaces with simple reflectivity functions are "dusty" and "specular" surfaces. An example of a dusty surface is the lunar surface, which reflects in all directions equally. Specular (purely mirror-like) surfaces such as polished metal reflect only at the angle of reflection = angle of incidence, and in a direction such that the incidence, normal, and emittance vectors are coplanar.

Most smooth things have a specular component to their reflection, but in general some light is reflected at all angles in decreasing amounts from the specular angle. One way to achieve this effect is to use the cosine of the angle between the predicted specular angle and the viewing angle, which is given by C where

$$C = 2\cos(i)\cos(e) - \cos(g)$$

This quantity is unity in the pure specular direction and falls off to zero at $\frac{\pi}{2}$ radians away from it. Convincing specular contributions of greater or less sharpness are produced by taking powers of C. A simple radiance formula that allows the simulation of both matte and specular effects is

$$L(i, e, g) = s(C)^n + (1 - s)\cos(i) \tag{3.35}$$

Here s varies between 0 and 1 and determines the fraction of specularly reflected light; n determines the sharpness of specularity peaks. As n increases, the specular peak gets sharper and sharper. Computer graphics research is constantly extending the frontiers of realistic and detailed reflectance, refractance, and illumination calculations [Blinn 1978; Phong 1975; Whitted 1980].

3.5.2 Surface Gradient

The reflectance functions described above are defined in terms of angles measured with respect to a local coordinate frame. For our development, it is more useful to relate the reflectivity function to surface gradients measured with respect to a viewer-oriented coordinate frame.

The concept of *gradient space*, which is defined in a viewer-oriented frame [Horn 1975], is extremely useful in understanding the recovery transformation algorithm for the surface normal. This gradient refers to the orientation of a physical surface, *not* to local intensities. It must not be confused with the *intensity* gradients discussed in Section 3.3 and elsewhere in this book.

Gradient space is a two-dimensional space of slants of scene surfaces. It measures a basic "intrinsic" (three-dimensional) property of surfaces. Consider the point-projection imaging geometry of Fig. 2.2, with the viewpoint at infinity (far from the scene relative to the scene dimensions). The image projection is then orthographic, nor perspective.

The surface gradient is defined for a surface expressed as $-z = f(x, y)$. The gradient is a vector (p, q), where

$$p = \frac{\partial(-z)}{\partial x} \qquad q = \frac{\partial(-z)}{\partial y} \tag{3.36}$$

Any plane in the image (such as the face plane of a polyhedral face) may be expressed in terms of its gradient. The general plane equation is

$$Ax + By + Cz + D = 0 \tag{3.37}$$

Thus

$$-z = \frac{A}{C}x + \frac{B}{C}y + \frac{D}{C} \tag{3.38}$$

and from (3.36) the gradient may be related to the plane equation:

$$-z = px + qy + K \tag{3.39}$$

Gradient space is thus the two-dimensional space of (p, q) vectors. The p and q axes are often considered to be superimposed on the x and y image plane coordinate axes. Then the (p, q) vector is "in the direction" of the surface slant of imaged surfaces. Any plane perpendicular to the viewing direction has a (p, q) vector of (0,0). Vectors on the q (or y) axis correspond to planes tilted about the x axis in an "upward" or "downward" ("$y$ward") direction (like the tilt of a dressing table

mirror). The direction arctan (q/p) is the direction of fastest change of surface depth $(-z)$ as x and y change. $(p^2 + q^2)^{1/2}$ is the rate of this change. For instance, a vertical plane "edge on" to the viewer has a (p, q) of $(I\infty, 0)$.

The *reflectance map* $R(p, q)$ represents this variation of perceived brightness with surface orientation. $R(p, q)$ gives scene radiance (Section 2.2.3) as a function of surface gradient (in our usual viewer-centered coordinate system). (Figure 3.27 showed the situation and defined some important angles.) $R(p, q)$ is usually shown as contours of constant scene radiance (Fig. 3.28). The following are a few useful cases.

In the case of a Lambertian surface with the source in the direction of the viewer $(i = e)$, the gradient space image looks like Fig. 3.28. Remember that Lambertian surfaces have constant intensity for constant illumination angle; these constant angles occur on the concentric circles of Fig. 3.28, since the direction of tilt does not affect the magnitude of the angle. The brightest surfaces are those illuminated from a normal direction—they are facing the viewer and so their gradients are $(0, 0)$.

Working this out from first principles, the incident angle and emittance angle are the same in this case, since the light is near the viewer. Both are the angle between the surface normal and the view vector. Looking at the x–y plane means a vector to the light source of $(0, 0, -1)$, and at a gradient point (p, q), the surface normal is $(p, q, -1)$. Also,

$$R = r_0 \cos i \tag{3.40}$$

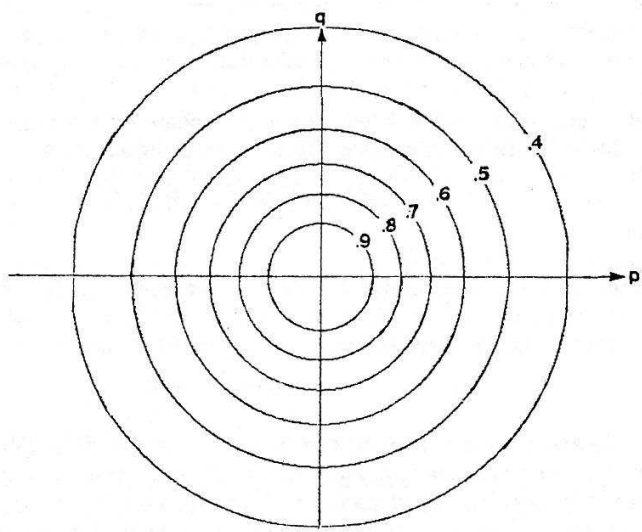


Fig. 3.28 Contours of constant radiance in gradient space for Lambertian surfaces; single light source near the viewpoint.

where r_o is a proportionality constant, and we conventionally use R to denote radiance in a viewer-centered frame. Let $\mathbf{n}_s$ and $\mathbf{n}$ be unit vectors in the source and surface normal directions. Since $\cos i = \mathbf{n}_s \cdot \mathbf{n}$

$$R = \frac{r_0}{(1 + p^2 + q^2)^{1/2}} \tag{3.41}$$

Thus $\cos(i)$ determines the image brightness, and so a plot of it is the gradient space image (Figs. 3.29 and 3.30).

For a more general light position, the mathematics is the same; if the light source is in the $(p_s, q_s, -1)$ direction, take the dot product of this direction and the surface normal.

$$R = r_o \mathbf{n} \cdot \mathbf{n}_s \tag{3.42}$$

Or, in other words,

$$R = \frac{r_o(p_s p + q_s q + 1)}{[(1 + p^2 + q^2)(1 + p_s^2 + q_s^2)]^{1/2}}$$

The phase angle g is constant throughout gradient space with orthographic projection (viewer distant from scene) and light source distant from scene.

Setting R constant to obtain contour lines gives a second-order equation, producing conic sections. In fact, the contours are produced by a set of cones of varying angles, whose axis is in the direction of the light source, intersecting a plane at unit distance from the origin. The resulting contours appear in Fig. 3.29. Here the dark line is the terminator, and represents all those planes that are edge-

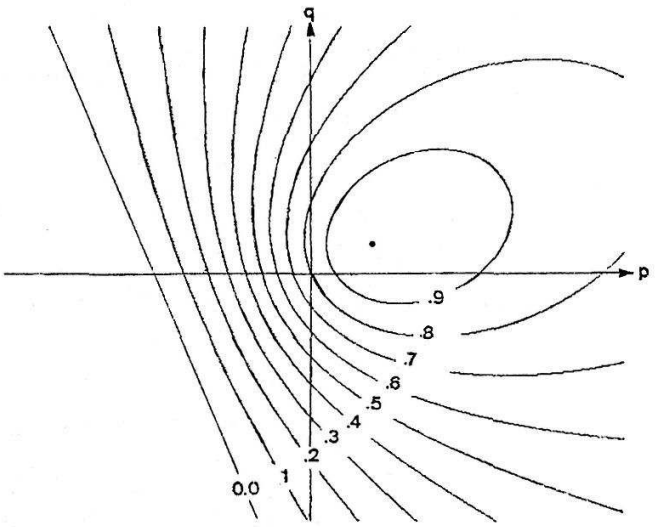


Fig. 3.29 Contours of constant radiance in gradient space. Lambertian surfaces; light not near viewpoint.

on to the light source; gradients on the back side of the terminator represent self-shadowed surfaces (facing away from the light). One intensity determines a contour and so gives a cone whose tangent planes all have that emittance. For a surface with specularity, contours of constant $I(i, e, g)$ could appear as in Fig. 3.30.

The point of specularity is between the matte component maximum brightness gradient and the origin. The brightest matte surface normal points at the light source and the origin points at the viewer. Pure specular reflection can occur if the vector tilts halfway toward the viewer maintaining the direction of tilt. Thus its gradient is on a line between the origin and the light-source direction gradient point.

3.5.3 Photometric Stereo

The reflectance equation (3.42) constrains the possible surface orientation to a locus on the reflectance map. Multiple light-source positions can determine the orientation uniquely [Woodham 1978]. Each separate light position gives a separate value for the intensity (proportional to radiance) at each point $f(\mathbf{x})$. If the surface reflectance r_o is unknown, three equations are needed to determine the reflectance together with the unit normal $\boldsymbol{n}$. If each source position vector is denoted by n_k, $k = 1, \ldots, 3$, the following equations result:

$$I_k(x, y) = r_o(\mathbf{n}_k \cdot \mathbf{n}), \qquad k = 1, \ldots, 3 \tag{3.43}$$

where I is normalized intensity. In matrix form

$$\mathbf{I} = r_o N \mathbf{n} \tag{3.44}$$

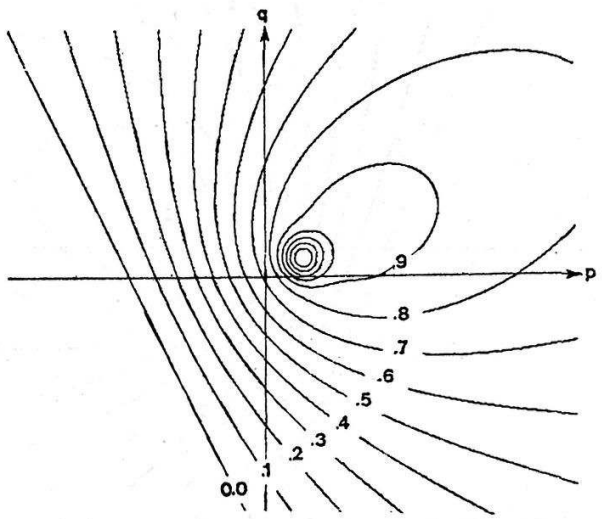


Fig. 3.30 Contours of constant radiance for a specular/matte surface.

where

$$\mathbf{I} = [I_1(x, y), I_2(x, y), I_3(x, y)]^T,$$

and

$$N = \begin{bmatrix} n_{11} & n_{12} & n_{13} \\ n_{21} & n_{22} & n_{23} \\ n_{31} & n_{32} & n_{33} \end{bmatrix} \tag{3.45}$$

and $I = fc$ where c is the appropriate normalization constant. If c is not known, it can be regarded as being part of r_o without affecting the normal direction calculation. As long as the three source positions $\mathbf{n}_1$, $\mathbf{n}_2$, $\mathbf{n}_3$ are not coplanar, the matrix N will have an inverse. Then solve for r_o and $\mathbf{n}$ by using (3.44), first using the fact that $\mathbf{n}$ is a unit vector to derive

$$r_0 = |N^{-1}\mathbf{I}| \tag{3.46}$$

and then solving for $\boldsymbol{n}$ to obtain

$$\mathbf{n} = \frac{1}{r_o} N^{-1}\mathbf{I} \tag{3.47}$$

Examples of a particular solution are shown in Fig. 3.31. Of course, a prerequisite for using this method is that the surface point not be in shadows for any of the sources.

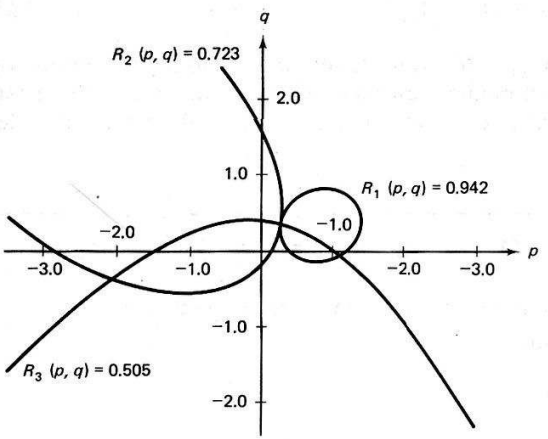


Fig. 3.31 A particular solution for photometric stereo.

3.5.4 Shape from Shading by Relaxation

Combining local information allows improved estimates for edges (Section 3.3.5) and for disparity (Section 3.4.2). In a similar manner local information can help in computing surface orientation [Ikeuchi 1980]. Basically, the reflectance equation

provides one constraint on the surface orientation and another is provided by the heuristic requirement that the surface be smooth.

Suppose there is an estimate of the surface normal at a point $(p(x, y), q(x, y))$. If the normal is not accurate, the reflectivity equation $I(x, y) = R(p, q)$ will not hold. Thus it seems reasonable to seek p and q that minimize $(I - R)^2$. The other requirement is that $p(x, y)$ and $q(x, y)$ be smooth, and this can be measured by their Laplacians $\nabla^2 p$ and $\nabla^2 q$. For a smooth curve both of these terms should be small. The goal is to minimize the error at a point,

$$E(x, y) = [I(x, y) - R(p, q)]^2 + \lambda[(\nabla^2 p)^2 + (\nabla^2 q)^2] \tag{3.48}$$

where the Lagrange multiplier λ [Russell 1976] incorporates the smoothness constraint. Differentiating $E(x, y)$ with respect to p and q and approximating derivatives numerically gives the following equations for $p(x, y)$ and $q(x, y)$:

$$p(x, y) = p_{av}(x, y) + T(x, y, p, q)\frac{\partial R}{\partial q} \tag{3.49}$$

$$q(x, y) = q_{av}(x, y) + T(x, y, p, q)\frac{\partial R}{\partial q} \tag{3.50}$$

where

$$T(x, y, p, q) = (1/\lambda)[I(x, y) - R(p, q)]$$

using

$$p_{av}(x, y) = \frac{1}{4}[p(x + 1, y) + p(x - 1, y) + p(x, y + 1) + p(x,y - 1)] \tag{3.51}$$

and a similar expression for q_{av}. Now Eqs. (3.49) and (3.50) lend themselves to solution by the Gauss-Seidel method: calculate the left-hand sides with an estimate for p and q and use them to derive a new estimate for the right-hand sides. More formally,

---

Algorithm 3.3: Shape from Shading [Ikeuchi 1980].

Step 0. k = 0. Pick an initial $p^0(x, y)$ and $q^0(x, y)$ near boundaries.

Step 1. k = k + 1; compute

$$p^k = p_{av}^{k-1} + T\frac{\partial R}{\partial p}$$

$$q^k = q_{av}^{k-1} + T\frac{\partial R}{\partial p}$$

Step 2. If the sum of all the E's is sufficiently small, stop. Else, go to step 1.

A loose end in this algorithm is that boundary conditions must be specified. These are values of p and q determined a priori that remain constant throughout each iteration. The simplest place to specify a surface gradient is at an occluding contour (see Fig. 3.32) where the gradient is nearly 90° to the line of sight. Unfortunately, p and q are infinite at these points. Ikeuchi's elegant solution to this is to use a different coordinate system for gradient space, that of a Gaussian sphere (Appendix 1). In this system, the surface normal is described relative to where it intersects the sphere if the tail of the normal is at the sphere's origin. This is the point at which a plane perpendicular to the normal would touch the sphere if translated toward it (Fig. 3.32b).

In this system the radiance may be described in terms of the spherical coordinates θ, ϕ. For a Lambertian surface

$$R(\theta, \phi) = \cos\theta \cos\theta_s + \sin\theta \sin\theta_s \cos(\phi - \phi_s) \tag{3.52}$$

At an occluding contour $\phi = \pi/2$ and θ is given by $\tan^{-1}(\partial y / \partial x)$, where the derivatives are calculated at the occluding contour (Fig. 3.32c).

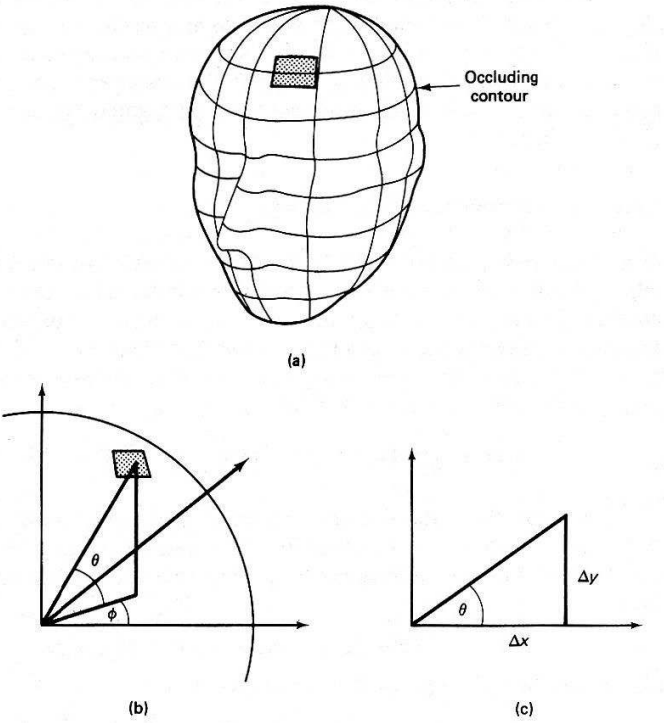


Fig. 3.32 (a) Occluding contour. (b) Gaussian sphere. (c) Calculating θ from occluding contour.

To use the (θ, ϕ) formulation instead of the (p, q) formulation is an easy matter. Simply substitute θ for p and ϕ for q in all instances of the formula in Algorithm 3.3.

3.6 OPTICAL FLOW

Much of the work on computer analysis of visual motion assumes a stationary observer and a stationary background. In contrast, biological systems typically move relatively continuously through the world, and the image projected on their retinas varies essentially continuously while they move. Human beings perceive smooth continuous motion as such.

Although biological visual systems are discrete, this quantization is so fine that it is capable of producing essentially continuous outputs. These outputs can mirror the continuous flow of the imaged world across the retina. Such continuous information is called *optical flow*. Postulating optical flow as an input to a perceptual system leads to interesting methods of motion perception.

The optical flow, or instantaneous velocity field, assigns to every point on the visual field a two-dimensional "retinal velocity" at which it is moving across the visual field. This section describes how approximations to instantaneous flow may be computed from the usual input situation in a sequence of discrete images. Methods of using optical flow to compute the observer's motion, a relative depth map, surface normals of his or her surroundings, and other useful information are given in Chapter 7.

3.6.1 The Fundamental Flow Constraint

One of the important features of optical flow is that it can be calculated simply, using local information. One way of doing this is to model the motion image by a continuous variation of image intensity as a function of position and time, then expand the intensity function $f(x, y, t)$ in a Taylor series.

$$f(x + dx, y + dy, t + dt) = \tag{3.53}$$
$$f(x, y, t) + \frac{\partial f}{\partial x} dx + \frac{\partial f}{\partial y} dy + \frac{\partial f}{\partial t} dt + \text{higher-order terms}$$

As usual, the higher-order terms are henceforth ignored. The crucial observation to be exploited is the following: If indeed the image at some time $t + dt$ is the result of the original image at time t being moved translationally by dx and dy, then in fact

$$f(x + dx, y + dy, t + dt) = f(x, y, t) \tag{3.54}$$

Consequently, from Eqs. (3.53) and (3.54),

$$-\frac{\partial f}{\partial t} = \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt} \tag{3.55}$$

Now $\frac{\partial f}{\partial t}$, $\frac{\partial f}{\partial x}$, and $\frac{\partial f}{\partial y}$ are all measurable quantities, and $\frac{dx}{dt}$ and $\frac{dy}{dt}$ are estimates of what we are looking for—the velocity in the x and y directions. Writing

$$\frac{dx}{dt} = u, \qquad \frac{dy}{dt} = v$$

gives

$$-\frac{\partial f}{\partial t} = \frac{\partial f}{\partial x}u + \frac{\partial f}{\partial y}v \tag{3.56}$$

or equivalently,

$$-\frac{\partial f}{\partial t} = \nabla f \cdot \mathbf{u} \tag{3.57}$$

where ∇f is the spatial gradient of the image and $\mathbf{u} = (u, v)$ the velocity.

The implications of (3.57) are interesting. Consider a fixed camera with a scene moving past it. The equations say that the *time* rate of change in intensity of a point in the image is (to first order) explained as the *spatial* rate of change in the intensity of the scene multiplied by the *velocity* that points of the scene move past the camera.

This equation also indicates that the velocity (u, v) must lie on a line perpendicular to the vector (f_x, f_y) where f_x and f_y are the partial derivatives with respect to x and y, respectively (Fig. 3.33). In fact, if the partial derivatives are very accurate the magnitude component of the velocity in the direction (f_x, f_y) is (from 3.57):

$$\frac{-f_t}{[(f_x^2 + f_y^2)]^{1/2}}$$

3.6.2 Calculating Optical Flow by Relaxation

Equation (3.57) constrains the velocity but does not determine it uniquely. The development of Section 3.5.4 motivates the search for a solution that satisfies Eq.

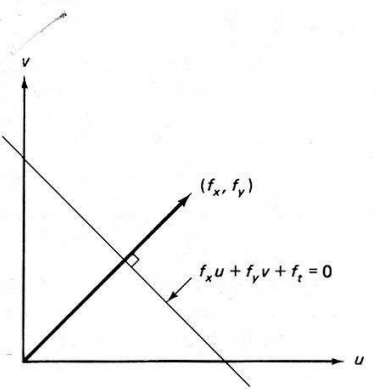


Fig. 3.33 Relation between (u, v) and (f_x, f_y).

(3.57) as closely as possible and also is locally smooth [Horn and Schunck 1980]. In this case as well, the Laplacians of the two velocity components, $\nabla^2 u$ and $\nabla^2 v$, can measure local smoothness.

Again using the method of Lagrange multipliers, minimize the flow error

$$E^2(x, y) = (f_x u + f_y v + f_t)^2 + \lambda^2[(\nabla^2 u)^2 + (\nabla^2 v)^2] \tag{3.58}$$

Differentiating this equation with respect to u and v provides equations for the change in error with respect to u and v, which must be zero for a minimum. Writing $\nabla^2 u$ as $u - u_{av}$ and $\nabla^2 v$ as $v - v_{av}$, these equations are

$$(\lambda^2 + f_x^2)u + f_x f_y v = \lambda^2 u_{av} - f_x f_t \tag{3.59}$$

$$f_x f_y u + (\lambda^2 + f_y^2)v = \lambda^2 v_{av} - f_y f_t \tag{3.60}$$

These equations may be solved for u and v, yielding

$$u = u_{av} - f_x \frac{P}{D} \tag{3.61}$$

$$v = v_{av} - f_y \frac{P}{D} \tag{3.62}$$

where

$$P = f_x u_{av} + f_y v_{av} + f_t$$

$$D = \lambda^2 + f_x^2 + f_y^2$$

To turn this into an iterative equation for solving $u(x, y)$ and $v(x, y)$, again use the Gauss-Seidel method.

Algorithm 3.4: Optical Flow [Horn and Schunck 1980].

$k = 0$.
Initialize all u^k and v^k to zero.
Until some error measure is satisfied, do

$$u^k = u_{av}^{k-1} - f_x \frac{P}{D}$$

$$v^k = v_{av}^{k-1} - f_y \frac{P}{D}$$

As Horn and Schunck demonstrate, this method derives the flow for two time frames, but it can be improved by using several time frames and using the final solution after one iteration at one time for the initial solution at the following time frame. That is:

Algorithm 3.5: Multiframe Optical Flow.

$t = 0$.
Initialize all $u(x, y, 0)$, $v(x, y, 0)$
for $t = 1$ *until* maxframes *do*

$$u(x, y, t) = u_{av}(x, y, t-1) - f_x \frac{P}{D}$$

$$v(x, y, t) = v_{av}(x, y, t-1) - f_y \frac{P}{D}$$

The results of using synthetic data from a rotating checkered sphere are shown in Fig. 3.34.

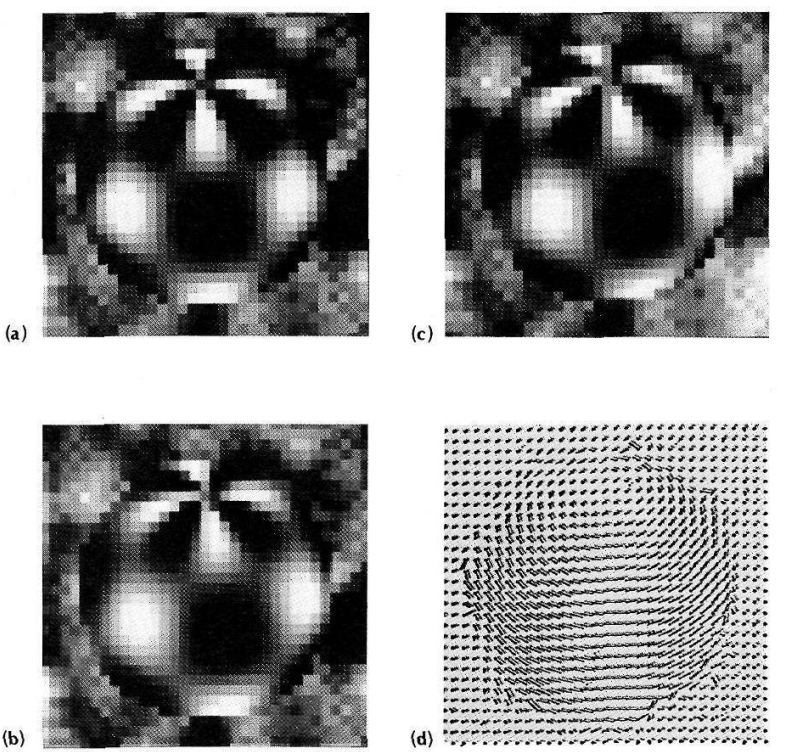


Fig. 3.34 Optical flow results. (a), (b) and (c) are three frames from the rotating sphere, (d) is the derived three-dimensional flow after 32 such time frames.

3.7 RESOLUTION PYRAMIDS

What is the best spatial resolution for an image? The sampling theorem states that the maximum spatial frequency in the image data must be less than half the sampling frequency in order that the sampled image represent the original unambiguously. However, the sampling theorem is not a good predictor of how easily objects can be recognized by computer programs. Often objects can be more easily recognized in images that have a very low sampling rate. There are two reasons for this. First, the computations are fewer because of the reduction in dimensionality. Second, confusing detail present in the high-resolution versions of the images may not appear at the reduced resolution. But even though some objects are more easily found at low resolutions, usually an object description needs detail only revealed at the higher resolutions. This leads naturally to the notion of a *pyramidal* image data structure in which the search for objects is begun at a low resolution, and refined at ever-increasing resolutions until one reaches the highest resolution of interest. Figure 3.35 shows the correspondence between pixels for the pyramidal structure.

In the next three sections, pyramids are applied to gray-level images and edge images. Pyramids, however, are a very general tool and can be used to represent any image at varying levels of detail.

3.7.1 Gray-level Consolidation

In some applications, redigitizing the image with a different sampling rate is a way to reduce the number of samples. However, most digitizer parameters are difficult to change, so that often computational means of reduction are needed. A straightforward method is to partition the digitized image into nonoverlapping

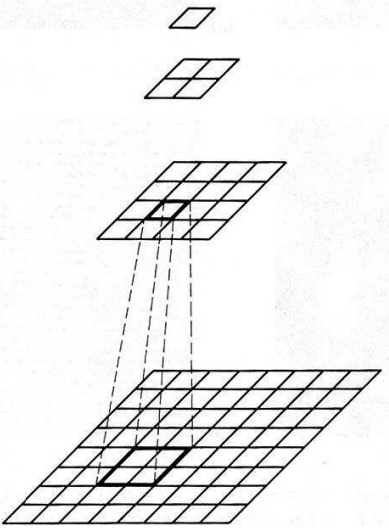


Fig. 3.35 Pyramidal image structure.

neighborhoods of equal size and shape and to replace each of those neighborhoods by the average pixel densities in that neighborhood. This operation is *consolidation.* For an $n \times n$ neighborhood, consolidation is equivalent to averaging the original image over the neighborhood followed by sampling at intervals n units apart.

Consolidation tends to offset the aliasing that would be introduced by sampling the sensed data at a reduced rate. This is due to the effects of the averaging step in the consolidation process. For the one-dimensional case where

$$f'(x) = \frac{1}{2}[f(x) + f(x + \Delta)] \tag{3.63}$$

the corresponding Fourier transform [Steiglitz 1974] is

$$H(u) = \frac{1}{2}\left[1 + e^{-j2\pi\Delta}\right]F(u) \tag{3.64}$$

which has magnitude $|H(u)| = \cos[\pi(u/u_o)]$ and phase $-\pi(u/u_o)$. The sampling frequency $u_o = 1/\Delta$ where Δ is the spacing between samples. Thus the averaging step has the effect of attenuating the higher frequencies of $F(u)$ as shown in Fig. 3.36. Since the higher frequencies are involved in aliasing, attenuating these frequencies reduces the aliasing effects.

3.7.2 Pyramidal Structures in Correlation

With correlation matching, the use of multiple resolution techniques can sometimes provide significant functional and computational advantages [Moravec 1977]. Binary search correlation uses pyramids of the input image and reference

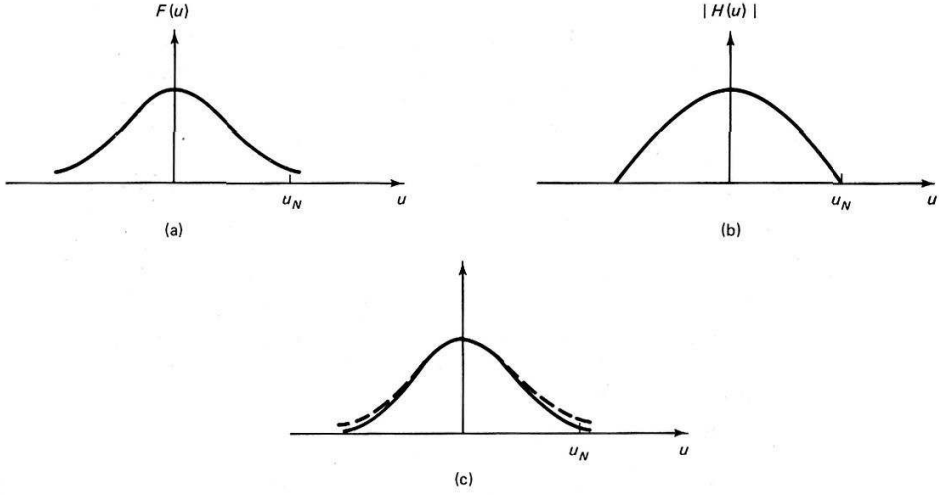


Fig. 3.36 Consolidation effects viewed in the spatial frequency domain. (a) Original transform. (b) Transform of averaging operator. (c) Transform of averaged image.

patterns. The algorithm partakes of the computational efficiency of binary (as opposed to linear) search [Knuth 1973]. Further, the low-resolution correlation operations at high levels in the pyramid ensure that the earlier correlations are on gross image features rather than details.

In binary search correlation a feature to be located is at some unknown location in the input image. The reference version of the feature originates in another image, the reference image. The feature in the reference image is contained in a window of $n \times n$ pixels. The task of the correlator is to find an $n \times n$ window in the input image that best matches the reference image window containing the feature. The details of the correlation processes are given in the following algorithm.

Algorithm 3.6: Binary Search Correlation Control Algorithm

Definitions

OrigReference: an $N \times N$ image containing a feature centered at (FeatureX, FeatureY).

OrigInput: an $M \times M$ array in which an instance of the Feature is to be located. For simplicity, assume that it is at the same resolution as OrigReference.

n: a window size; an $n \times n$ window in OrigReference is large enough to contain the Feature.

Window: an $n \times n$ array containing a varying-resolution subimage of OrigReference centered on the Feature.

Input: a $2n \times 2n$ array containing a varying-resolution subimage of OrigInput, centered on the best match for the Feature.

Reference: a temporary array.

Algorithm

1. Input := Consolidate OrigInput by a factor of $2n/M$ to size $2n \times 2n$.
2. Reference := Consolidate OrigReference by the same factor $2n/M$ to size $2nN/M \times 2nN/M$. This consolidation takes the Feature to a new (Feature$X$, Feature$Y$).
3. Window := $n \times n$ window from Reference centered on the new (Feature$X$, Feature$Y$).
4. Calculate the match metric of the window at the $(n + 1)^2$ locations in Input at which it is wholly contained. Say that the best match occurs at (BestMatch$X$, BestMatch$Y$) in Input.

5. Input := $n \times n$ window from Input centered at (BestMatch$X$, BestMatch$Y$), enlarged by a factor of 2.
6. Reference := Reference enlarged by a factor of 2. This takes Feature to a new (Feature$X$, Feature$Y$).
7. *Go to* 3.

Through time, the algorithm uses a reference image for matching that is always centered on the feature to be matched, but that homes in on the feature by being increased in resolution and thus reduced in linear image coverage by a factor of 2 each time. In the input image, a similar homing-in is going on, but the search area is usually twice the linear dimension of the reference window. Further, the center of the search area varies in the input image as the improved resolution refines the point of best match.

Binary search correlation is for matching features with context. The template at low resolution possibly corresponds to much of the area around the feature, while the feature may be so small in the initial consolidated images as to be invisible. The coarse-to-fine strategy is perfect for such conditions, since it allows gross features to be matched first and to guide the later high-resolution search for best match. Such matching with context is less useful for locating several instances of a shape dotted at random around an image.

3.7.3 Pyramidal Structures in Edge Detection

As an example of the use of pyramidal structures in processing, consider the use of such structures in edge detection. This application, after [Tanimoto and Pavlidis 1975], uses two pyramids, one to store the image and another to store the image edges. The idea of the algorithm is that a neighborhood in the low-resolution image where the gray-level values are the same is taken to imply that in fact there is no gray-level change (edge) in the neighborhood. Of course, the low-resolution levels in the pyramid tend to blur the image and thus attenuate the gray-level changes that denote edges. Thus the starting level in the pyramid must be picked judiciously to ensure that the important edges are detected.

Algorithm 3.7: Hierarchical Edge Detection

```
recursive procedure refine (k, x, y)
  begin
  if k < MaxLevel then
    for dx = 0 until 1 do
    for dy = 0 until 1 do
      if EdgeOp (k, x + dx, y + dy) > Threshold(x)
      then refine (k + 1, x + dx, y + dy)
  end;
```

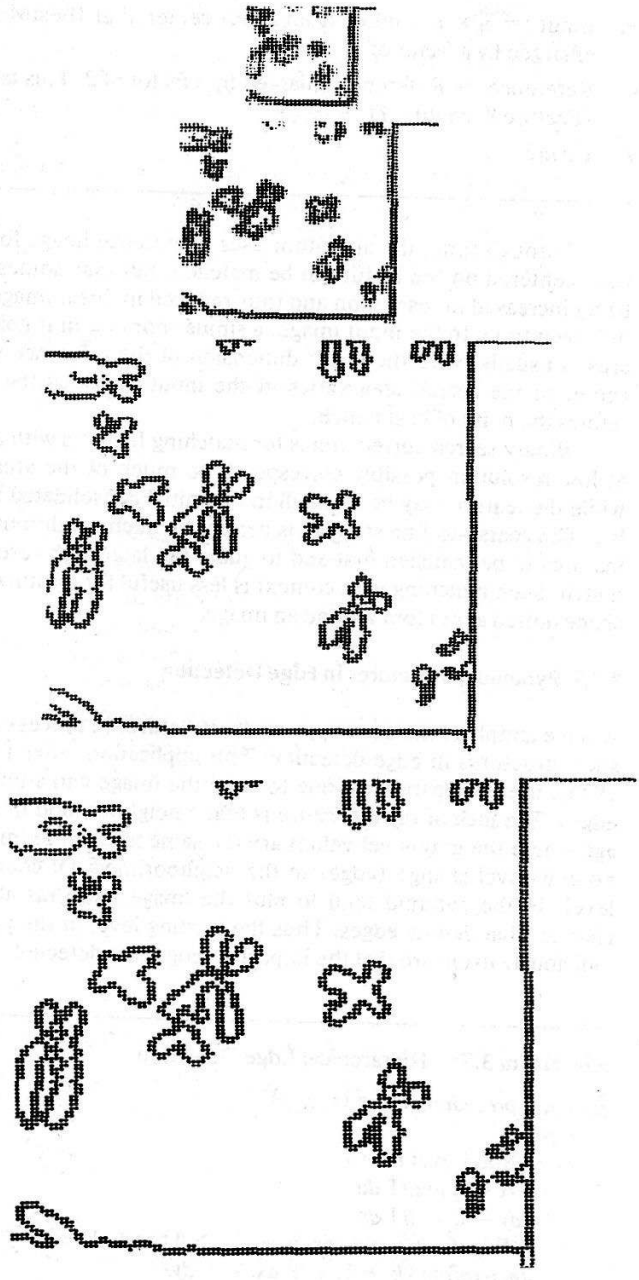


Fig. 3.37 Pyramidal edge detection.

```
procedure FindEdges:
  begin
  comment apply operator to every pixel in the
      starting level s, refining where necessary;
    for x := 0 until 2^S − 1 do
    for y := 0 until 2^S − 1 do
      if EdgeOp (s, x, y) > Threshold(s)
      then refine (s, x, y);
  end;
```

Figure 3.37 shows Tanimoto's results for a chromosome image. The table inset shows the computational advantage in terms of the calls to the edge operator as a function of the starting level s.

Similar kinds of edge detection strategies based on pyramids have been pursued by [Levine 1978; Hanson and Riseman 1978]. The latter effort is a little different in that processing within the pyramid is bidirectional; information from edges detected at a high-resolution level is projected to low-resolution levels of the pyramid.

EXERCISES

3.1 Derive an analytical expression for the response of the Sobel operator to a vertical step edge as a function of the distance of the edge to the center of the operator.

3.2 Use the formulas of Eqs. (3.31) to derive the digital template function for g_1 in a 5^3 pixel domain.

3.3 Specify a version of Algorithm 3.1 that uses the gradient edge operator instead of the "crack" edge operator.

3.4 In photometric stereo, three or more light source positions are used to determine a surface orientation. The dual of this problem uses surface orientations to determine light source position. What is the usefulness of the latter formulation? In particular, how does it relate to Algorithm 3.3?

3.5 Using any one of Algorithms 3.1 through 3.4 as an example, show how it could be modified to use pyramidal data structures.

3.6 Write a reflectance function to capture the "grazing incidence" phenomenon—surfaces become more mirror-like at small angles of incidence (and reflectance).

3.7 Equations 3.49 and 3.50 were derived by minimizing the local error. Show how these equations are modified when total error [i.e., $\sum_{x,y} E(x, y)$] is minimized.

REFERENCES

ABDOU, I. E. "Quantitative methods of edge detection." USCIPI Report 830, Image Processing Institute, Univ. Southern California, July 1978.

AKATSUKA, T., T. ISOBE, and O. TAKATANI. "Feature extraction of stomach radiograph." *Proc.*, 2nd IJCPR, August 1974, 324-328.

ANDREWS, H. C. and B. R. HUNT. *Digital Image Restoration.* Englewood Cliffs, NJ: Prentice-Hall, 1977.

ATTNEAVE, F. "Some informational aspects of visual perception." *Psychological Review 61*, 1954.

BARROW, H. G. and J. M. TENENBAUM. "Computational Vision." *Proc. IEEE 69*, 5, May 1981, 572-595

BARROW, H. G. and J. M. TENENBAUM. "Recovering intrinsic scene characteristics from images." Technical Note 157, AI Center, SRI International, April 1978.

BINFORD, T. O. "Visual perception by computer." *Proc.*, IEEE Conf. on Systems and Control, Miami, December 1971.

BLINN, J. E. "Computer display of curved surfaces." Ph.D. dissertation, Computer Science Dept., Univ. Utah, 1978.

FREI, W. and C. C. CHEN. "Fast boundary detection: a generalization and a new algorithm." *IEEE Trans. Computers 26*, 2, October 1977, 988-998.

GONZALEZ, R. C. and P. WINTZ. *Digital Image Processing.* Reading, MA: Addison-Wesley, 1977.

GRIFFITH, A. K. "Edge detection in simple scenes using a priori information." *IEEE Trans. Computers 22*, 4, April 1973.

HANSON, A. R. and E. M. RISEMAN (Eds.). *Computer Vision Systems (CVS).* New York: Academic Press, 1978.

HORN, B. K. P. "Determining lightness from an image." *CGIP 3*, 4, December 1974, 277-299.

HORN, B. K. P. "Shape from shading." In *PCV*, 1975.

HORN, B. K. P. and B. G. SCHUNCK. "Determining optical flow." AI Memo 572, AI Lab, MIT, April 1980.

HORN, B. K. P. and R. W. SJOBERG. "Calculating the reflectance map." *Proc.*, DARPA IU Workshop, November 1978, 115-126.

HUBEL, D. H. and T. N. WIESEL. "Brain mechanisms of vision." *Scientific American*, September 1979, 150-162.

HUECKEL, M. "An operator which locates edges in digitized pictures." *J. ACM 18*, 1, January 1971, 113-125.

HUECKEL, M. "A local visual operator which recognizes edges and lines." *J. ACM 20*, 4, October 1973, 634-647.

IKEUCHI, K. "Numerical shape from shading and occluding contours in a single view." AI Memo 566, AI Lab, MIT, revised February 1980.

KIRSCH, R. A. "Computer determination of the constituent structure of biological images." *Computers and Biomedical Research 4*, 3, June 1971, 315-328.

KNUTH, D. E. *The Art of Computer Programming.* Reading, MA: Addison-Wesley, 1973.

LEVINE, M. D. "A knowledge-based computer vision system." In *CVS*, 1978.

LIU, H. K. "Two- and three-dimensional boundary detection." *CGIP 6*, 2, 1977, 123-134.

MARR, D. and T. POGGIO. "Cooperative computation of stereo disparity." *Science 194*, 1976, 283-287.

MARR, D. and T. POGGIO. "A theory of human stereo vision." AI Memo 451, AI Lab, MIT, November 1977.

MERO, L. and Z. VASSY. "A simplified and fast version of the Hueckel operator for finding optimal edges in pictures." *Proc.*, 4th IJCAI, September 1975, 650-655.

MORAVEC, H. P. "Towards automatic visual obstacle avoidance." *Proc.*, 5th IJCAI, August 1977, 584.

NEVATIA, R. "Evaluation of a simplified Hueckel edge-line detector." Note, *CGIP 6*, 6, December 1977, 582-588.

PHONG, B-T. "Illumination for computer generated pictures." *Commun. ACM 18*, 6, June 1975, 311-317.

PINGLE, K. K. and J. M. TENENBAUM. "An accommodating edge follower." *Proc.*, 2nd IJCAI, September 1971, 1-7.

PRAGER, J. M. "Extracting and labeling boundary segments in natural scenes." *IEEE Trans. PAMI 2*, 1, January 1980, 16-27.

PRATT, W. K. *Digital Image Processing.* New York: Wiley-Interscience, 1978.

PREWITT, J. M. S. "Object enhancement and extraction." In *Picture Processing and Psychopictorics*, B. S. Lipkin and A. Rosenfeld (Eds.). New York: Academic Press, 1970.

QUAM, L. and M. J. HANNAH. "Stanford automated photogrammetry research." AIM-254, Stanford AI Lab, November 1974.

ROBERTS, L. G. "Machine perception of three-dimensional solids." In *Optical and Electro-optical Information Processing*, J. P. Tippett et al. (Eds.). Cambridge, MA: MIT Press, 1965.

ROSENFELD, A. and A. C. KAK. *Digital Picture Processing.* New York: Academic Press, 1976.

ROSENFELD, A., R. A. HUMMEL, and S. W. ZUCKER. "Scene labelling by relaxation operations." *IEEE Trans. SMC 6*, 1976, 430.

RUSSELL, D. L. (Ed.). *Calculus of Variations and Control Theory.* New York: Academic Press, 1976.

SHAPIRA, R. "A technique for the reconstruction of a straight-edge, wire-frame object from two or more central projections." *CGIP 3*, 4, December 1974, 318-326.

SHIRAI, V. "Analyzing intensity arrays using knowledge about scenes." In *PCV*, 1975.

STEIGLITZ, K. *An Introduction to Discrete Systems.* New York: Wiley, 1974.

STOCKHAM, T. J., Jr. "Image processing in the context of a visual model." *Proc. IEEE 60*, 7, July 1972, 828-842.

TANIMOTO, S. and T. PAVLIDIS. "A hierarchical data structure for picture processing." *CGIP 4*, 2, June 1975, 104-119.

TRETIAK, O. J. "A parameteric model for edge detection." *Proc.*, 3rd COMPSAC, November 1979, 884-887.

TURNER, K. J. "Computer perception of curved objects using a television camera." Ph.D. dissertation, Univ. Edinburgh, 1974.

WECHSLER, H. and J. SKLANSKY. "Finding the rib cage in chest radiographs." *Pattern Recognition 9*, 1977, 21-30.

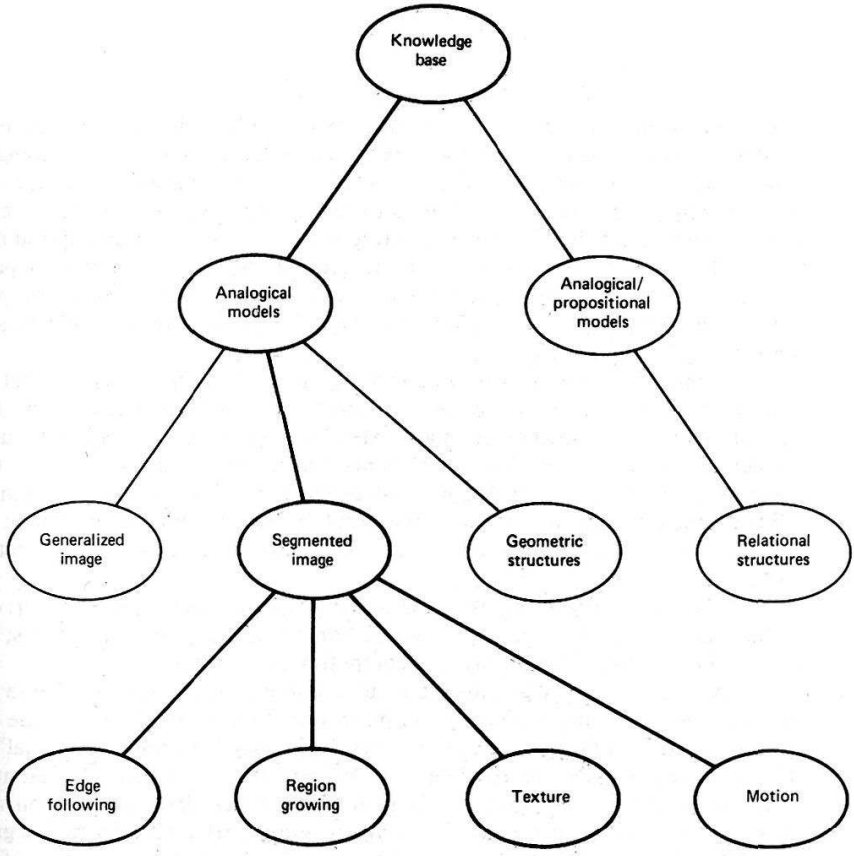
WHITTED, T. "An improved illumination model for shaded display." *Comm. ACM 23*, 6, June 1980, 343-349.

WOODHAM, R. J. "Photometric stereo: A reflectance map technique for determining surface orientation from image intensity." *Proc.*, 22nd International Symp., Society of Photo-optical Instrumentation Engineers, San Diego, CA, August 1978, 136-143.

ZUCKER, S. W. and R. A. HUMMEL. "An optimal three-dimensional edge operator." Report 79-10, McGill Univ., April 1979.

ZUCKER, S. W., R. A. HUMMEL, and A. ROSENFELD. "An application of relaxation labeling to line and curve enhancement." *IEEE Trans. Computers 26*, 1977. Apral N°4 pp 394

SEGMENTED IMAGES II



The idea of segmentation has its roots in work by the Gestalt psychologists (e.g., Kohler), who studied the preferences exhibited by human beings in grouping or organizing sets of shapes arranged in the visual field. Gestalt principles dictate certain grouping preferences based on features such as proximity, similarity, and continuity. Other results had to do with figure/ground discrimination and optical illusions. The latter have provided a fertile ground for vision theories to post-Gestaltists such as Gibson and Gregory, who emphasize that these grouping mechanisms organize the scene into *meaningful units* that are a significant step toward image understanding.

In computer vision, grouping parts of a generalized image into units that are homogeneous with respect to one or more characteristics (or features) results in a *segmented image.* The segmented image extends the generalized image in a crucial respect: it contains the beginnings of domain-dependent interpretation. At this descriptive level the internal domain-dependent models of objects begin to influence the grouping of generalized image structures into units meaningful in the domain. For instance, the model may supply crucial parameters to segmentation procedures.

In the segmentation process there are two important aspects to consider: one is the data structure used to keep track of homogeneous groups of features; the other is the transformation involved in computing the features.

Two basic sorts of segments are natural: boundaries and regions. These can be used combined into a single descriptive structure, a set of nodes (one per region), connected by arcs representing the "adjacency" relation. The "dual" of this structure has arcs corresponding to boundaries connecting nodes representing points where several regions meet. Chapters 4 and 5 describe segmentation with respect to boundaries and regions respectively, emphasizing gray levels and gray-level differences as indicators of segments. Of course, from the standpoint of the

algorithms involved, it is irrelevant whether the features are intensity gray levels or intrinsic image values perhaps representing motion, color, or range.

Texture and motion images are addressed in Chapters 6 and 7. Each has several computationally difficult aspects, and neither has received the attention given static, nontextured images. However, each is very important in the segmentation enterprise.

Boundary Detection

4

4.1 ON ASSOCIATING EDGE ELEMENTS

Boundaries of objects are perhaps the most important part of the hierarchy of structures that links raw image data with their interpretation [Marr 1975]. Chapter 3 described how various operators applied to raw image data can yield primitive edge elements. However, an image of only disconnected edge elements is relatively featureless; additional processing must be done to group edge elements into structures better suited to the process of interpretation. The goal of the techniques in this chapter is to perform a level of *segmentation*, that is, to make a coherent one-dimensional (*edge*) feature from many individual local edge elements. The feature could correspond to an object boundary or to any meaningful boundary between scene entities. The problems that edge-based segmentation algorithms have to contend with are shown by Fig. 4.1, which is an image of the local edge elements yielded by one common edge operator applied to a chest radiograph. As can be seen, the edge elements often exist where no meaningful scene boundary does, and conversely often are absent where a boundary is. For example, consider the boundaries of ribs as revealed by the edge elements. Missing edge elements and extra edge elements both tend to frustrate the segmentation process.

The methods in this chapter are ordered according to the amount of knowledge incorporated into the grouping operation that maps edge elements into boundaries. "Knowledge" means implicit or explicit constraints on the likelihood of a given grouping. Such constraints may arise from general physical arguments or (more often) from stronger restrictions placed on the image arising from domain-dependent considerations. If there is much knowledge, this implies that the global form of the boundary and its relation to other image structures is very constrained. Little prior knowledge means that the segmentation must proceed more on the basis of local clues and evidence and general (domain-dependent) assumptions with fewer expectations and constraints on the final resulting boundary.

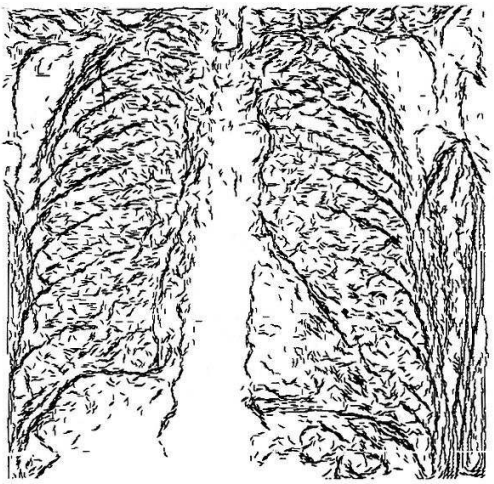


Fig. 4.1 Edge elements in a chest radiograph.

These constraints take many forms. Knowledge of where to expect a boundary allows very restricted searches to verify the edge. In many such cases, the domain knowledge determines the type of curve (its parameterization or functional form) as well as the relevant "noise processes." In images of polyhedra, only straight-edged boundaries are meaningful, and they will come together at various sorts of vertices arising from corners, shadows of corners, and occlusions. Human rib boundaries appear approximately like conic sections in chest radiographs, and radiographs have complex edge structures that can compete with rib edges. All this specific knowledge can and should guide our choice of grouping method.

If less is known about the specific image content, one may have to fall back on general world knowledge or heuristics that are true for most domains. For instance, in the absence of evidence to the contrary, the shorter line between two points might be selected over a longer line. This sort of general principle is easily built into evaluation functions for boundaries, and used in segmentation algorithms that proceed by methodically searching for such groupings. If there are no a priori restrictions on boundary shapes, a general contour-extraction method is called for, such as edge following or linking of edge elements.

The methods we shall examine are the following:

1. *Searching near an approximate location.* These are methods for refining a boundary given an initial estimate.
2. *The Hough transform.* This elegant and versatile technique appears in various guises throughout computer vision. In this chapter it is used to detect boundaries whose shape can be described in an analytical or tabular form.
3. *Graph searching.* This method represents the image of edge elements as a graph. Thus a boundary is a path through a graph. Like the Hough transform, these techniques are quite generally applicable.

4. *Dynamic programming.* This method is also very general. It uses a mathematical formulation of the globally best boundary and can find boundaries in noisy images.
5. *Contour following.* This hill-climbing technique works best with good image data.

4.2 SEARCHING NEAR AN APPROXIMATE LOCATION

If the approximate or a priori likely location of a boundary has been determined somehow, it may be used to guide the effort to refine that boundary [Kelly 1971]. The approximate location may have been found by one of the techniques below applied to a lower resolution image, or it may have been determined using high-level knowledge.

4.2.1 Adjusting A Priori Boundaries

This idea was described by [Bolles 1977] (see Fig. 4.2). Local searches are carried out at regular intervals along directions perpendicular to the approximate (a priori) boundary. An edge operator is applied to each of the discrete points along each of these perpendicular directions. For each such direction, the edge with the highest magnitude is selected from among those whose orientations are nearly parallel to the tangent at the point on the nearby a priori boundary. If sufficiently many elements are found, their locations are fit with an analytic curve such as a low-degree polynomial, and this curve becomes the representation of the boundary.

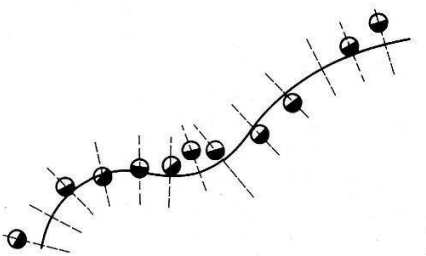


Fig. 4.2 Search orientations from an approximate boundary location.

4.2.2 Non-linear Correlation in Edge Space

In this correlation-like technique, the a priori boundary is treated as a rigid template, or piece of rigid wire along which edge operators are attached like beads. The a priori representation thus also contains relative locations at which the existence of edges will be tested (Fig. 4.3). An edge element returned by the edge-operator application "matches" the a priori boundary if its contour is tangent to the template and its magnitude exceeds some threshold. The template is to be moved around the image, and for each location, the number of matches is computed. If the number of matches exceeds a threshold, the boundary location is declared to

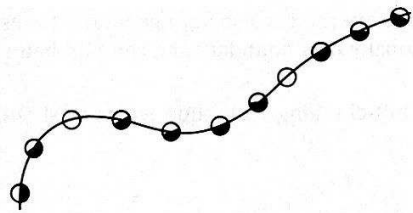


Fig. 4.3 A template for edge-operator application.

be the current template location. If not, the template is moved to a different image point and the process is repeated. Either the boundary will be located or there will eventually be no more image points to try.

4.2.3 Divide-and-Conquer Boundary Detection

This is a technique that is useful in the case that a low-curvature boundary is known to exist between two edge elements and the noise levels in the image are low (Algorithm 8.1). In this case, to find a boundary point in between the two known points, search along the perpendiculars of the line joining the two points. The point of maximum magnitude (if it is over some threshold) becomes a break point on the boundary and the technique is applied recursively to the two line segments formed between the three known boundary points. (Some fix must be applied if the maximum is not unique.) Figure 4.4 shows one step in this process. Divide-and-conquer boundary detection has been used to outline kidney boundaries on computed tomograms (these images were described in Section 2.3.4) [Selfridge et al. 1979].

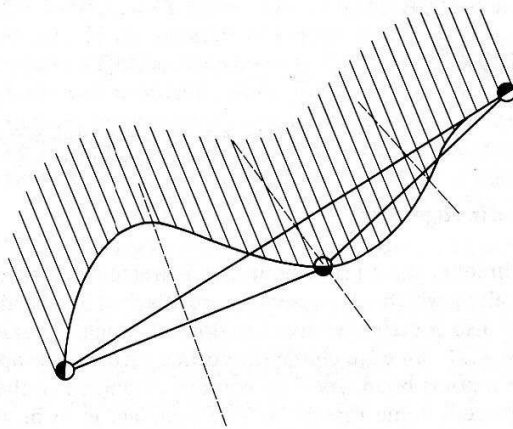


Fig. 4.4 Divide and conquer technique.

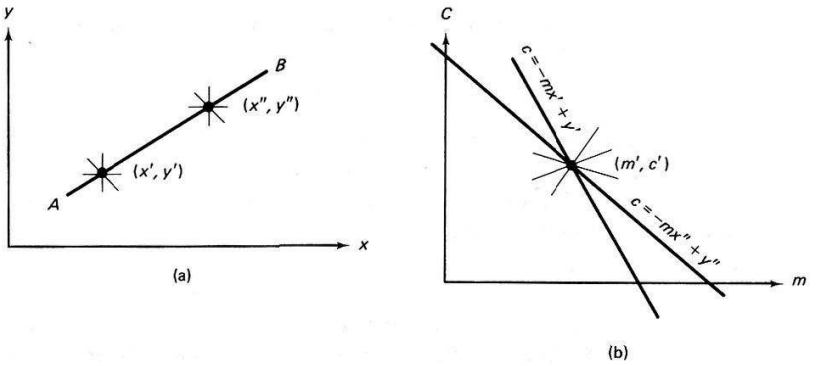


Fig. 4.5 A line (a) in image space; (b) in parameter space.

4.3 THE HOUGH METHOD FOR CURVE DETECTION

The classical Hough technique for curve detection is applicable if little is known about the location of a boundary, but its shape can be described as a parametric curve (e.g., a straight line or conic). Its main advantages are that it is relatively unaffected by gaps in curves and by noise.

To introduce the method [Duda and Hart 1972], consider the problem of detecting straight lines in images. Assume that by some process image points have been selected that have a high likelihood of being on linear boundaries. The Hough technique organizes these points into straight lines, basically by considering all possible straight lines at once and rating each on how well it explains the data.

Consider the point $\mathbf{x}'$ in Fig. 4.5a, and the equation for a line $y = mx + c$. What are the lines that could pass through $\mathbf{x}'$? The answer is simply all the lines with m and c satisfying $y' = mx' + c$. Regarding (x', y') as fixed, the last equation is that of a line in $m-c$ space, or parameter space. Repeating this reasoning, a second point $(\mathbf{x}'', \mathbf{y}'')$ will also have an associated line in parameter space and, furthermore, these lines will intersect at the point (m', c') which corresponds to the line AB connecting these points. In fact, all points on the line AB will yield lines in parameter space which intersect at the point (m', c'), as shown in Fig. 4.5b.

This relation between image space $\mathbf{x}$ and parameter space suggests the following algorithm for detecting lines:

Algorithm 4.1: Line Detection with the Hough Algorithm

1. Quantize parameter space between appropriate maximum and minimum values for c and m.
2. Form an accumulator array $A(c, m)$ whose elements are initially zero.
3. For each point (x,y) in a *gradient* image such that the strength of the gradient

exceeds some threshold, increment all points in the accumulator array along the appropriate line, i.e.,

$$A(c, m) := A(c, m) + 1$$

for m and c satisfying $c = -mx + y$ within the limits of the digitization.

4. Local maxima in the accumulator array now correspond to collinear points in the image array. The values of the accumulator array provide a measure of the number of points on the line.

This technique is generally known as the Hough technique [Hough 1962].

Since m may be infinite in the slope-intercept equation, a better parameterization of the line is $x \sin\theta + y \cos\theta = r$. This produces a sinusoidal curve in (r, θ) space for fixed x, y, but otherwise the procedure is unchanged.

The generalization of this technique to other curves is straightforward and this method works for any curve $f(\mathbf{x}, \mathbf{a}) = 0$, where a is a parameter vector. (In this chapter we often use the symbol f as various general functions unrelated to the image gray-level function.) In the case of a circle parameterized by

$$(x-a)^2 + (y-b)^2 = r^2 \tag{4.1}$$

for fixed $\mathbf{x}$, the modified algorithm 4.1 increments values of a, b, r lying on the surface of a cone. Unfortunately, the computation and the size of the accumulator array increase exponentially as the number of parameters, making this technique practical only for curves with a small number of parameters.

The Hough method is an efficient implementation of a generalized matched filtering strategy (i.e., a template-matching paradigm). For instance, in the case of a circle, imagine a template composed of a circle of 1's (at a fixed radius R) and 0's everywhere else. If this template is convolved with the gradient image, the result is the portion of the accumulator array $A(a, b, R)$.

In its usual form, the technique yields a set of parameters for a curve that best explains the data. The parameters may specify an infinite curve (e.g., a line or parabola). Thus, if a finite curve segment is desired, some further processing is necessary to establish end points.

4.3.1 Use of the Gradient

Dramatic reductions in the amount of computation can be achieved if the gradient direction is integrated into the algorithm [Kimme et al. 1975]. For example, consider the problem of detecting a circle of fixed radius R.

Without gradient information, all values a, b lying on the circle given by (4.1) are incremented. With the gradient direction, only the points near (a,b) in Fig. 4.6 need be incremented. From geometrical considerations, the point (a,b) is given by

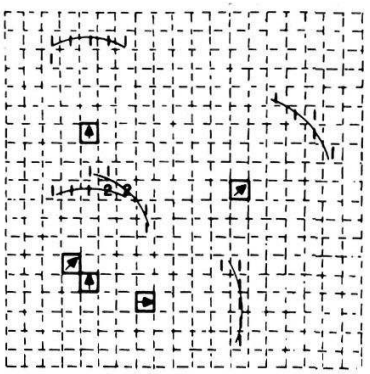


Fig 4.6 Reduction in computation with gradient information

$$a = x - r\sin\phi \tag{4.2}$$
$$b = y + r\cos\phi$$

where $\phi(x)$ is the gradient angle returned by an edge operator. Implicit in these equations is the assumption that the circle is the boundary of a disk that has gray levels greater than its surroundings. These equations may also be derived by differentiating (4.2), recognizing that $dy/dx = \tan\phi$, and solving for a and b between the resultant equation and (4.2). Similar methods can be applied to other conics. In each case, the use of the gradient saves one dimension in the accumulator array.

The gradient magnitude can also be used as a heuristic in the incrementing procedure. Instead of incrementing by unity, the accumulator array location may be incremented by a function of the gradient magnitude. This heuristic can balance the magnitude of brightness change across a boundary with the boundary length, but it can lead to detection of phantom lines indicated by a few bright points, or to missing dim but coherent boundaries.

4.3.2 Some Examples

The Hough technique has been used successfully in a variety of domains. Some examples include the detection of human hemoglobin fingerprints [Ballard et al. 1975], the detection of tumors in chest films [Kimme et al. 1975], the detection of storage tanks in aerial images [Lantz et al. 1978], and the detection of ribs in chest radiographs [Wechsler and Sklansky 1977]. Figure 4.7 shows the tumor-detection application. A section of the chest film (Fig. 4.7b) is searched for disks of radius 3 units. In Fig. 4.7c, the resultant accumulator array $A[a, b, 3]$ is shown in a pictoral fashion, by interpreting the array values as gray levels. This process is repeated for various radii and then a set of likely circles is chosen by setting a radius-dependent threshold for the accumulator array contents. This result is shown in Fig. 4.7d. The