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

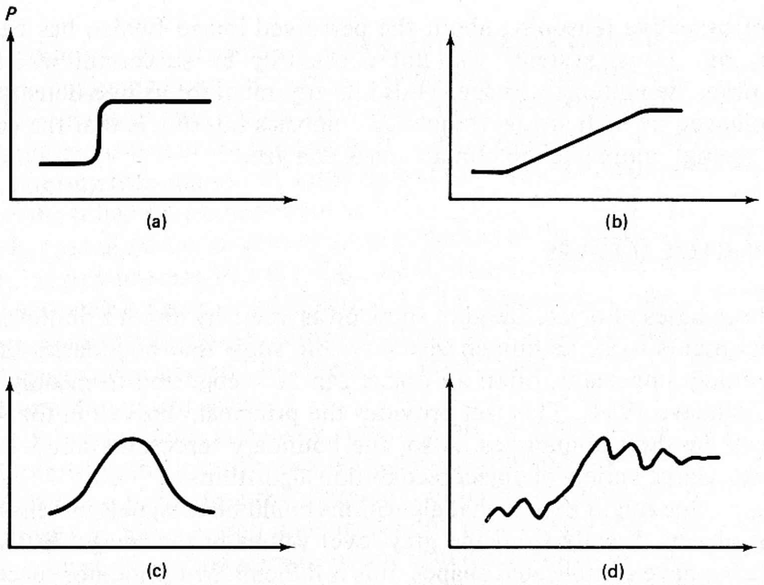


Fig. 3.9 Edge profiles.

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

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

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

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

Δ_1

0	1
−1	0

Δ_2

1	0
0	−1

(a)

−1	0	1
−1	0	1
−1	0	1

1	1	1
0	0	0
−1	−1	−1

(b)

−1	0	1
−2	0	2
−1	0	1

1	2	1
0	0	0
−1	−2	−1

(c)

Fig. 3.10 Gradient operators.

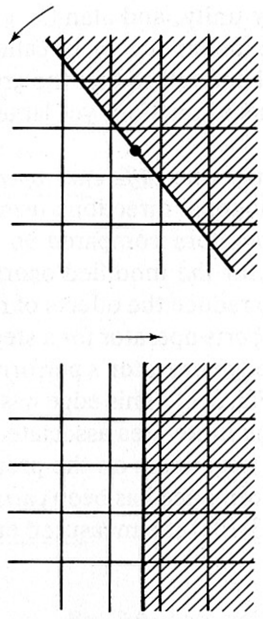


Fig. 3.11 Edge models for orientation and displacement sensitivity analyses.

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

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

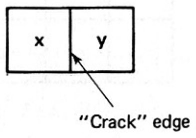


Fig. 3.12 "Crack" edge representation.

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

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.

$n = \frac{1}{2}$

$$\begin{array}{|cc|}\hline -1 & 1 \\ \hline\end{array} \qquad \begin{array}{|c|}\hline 1 \\ -1 \\ \hline\end{array} \qquad \begin{array}{cc} & 1 \\ -1 & \end{array} \qquad \begin{array}{cc} 1 & \\ & -1 \end{array}$$

$n = 1$

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 1 \\ -1 & 0 & 1 \\ -1 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & -1 \\ 0 & -1 & -1 \end{bmatrix}$$

$n = 2$

$$\begin{bmatrix} -1 & -1 & 0 & 1 & 1 \\ -1 & -1 & 0 & 1 & 1 \\ -1 & -1 & 0 & 1 & 1 \\ -1 & -1 & 0 & 1 & 1 \\ -1 & -1 & 0 & 1 & 1 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \end{bmatrix} \quad \begin{bmatrix} 0 & 1 & 1 & 1 & 1 \\ -1 & 0 & 1 & 1 & 1 \\ -1 & -1 & 0 & 1 & 1 \\ -1 & -1 & -1 & 0 & 1 \\ -1 & -1 & -1 & -1 & 0 \end{bmatrix} \quad \begin{bmatrix} 1 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & -1 \\ 1 & 1 & 0 & -1 & -1 \\ 1 & 0 & -1 & -1 & -1 \\ 0 & -1 & -1 & -1 & -1 \end{bmatrix}$$

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

$$\begin{array}{|ccc|}\hline 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ \hline\end{array}$$

$$\begin{array}{|ccc|}\hline -1 & \sqrt{2} & -1 \\ & & \\ 1 & \sqrt{2} & 1 \\ \hline\end{array} \quad \begin{array}{|ccc|}\hline & 1 & \\ -1 & & 1 \\ & -1 & \\ \hline\end{array} \quad \begin{array}{|ccc|}\hline & 1 & \\ -1 & & 1 \\ & -1 & \\ \hline\end{array} \quad \begin{array}{|ccc|}\hline 1 & -2 & 1 \\ -2 & 4 & -2 \\ 1 & -2 & 1 \\ \hline\end{array}$$

$$\begin{array}{|ccc|}\hline -1 & & 1 \\ \sqrt{2} & & \sqrt{2} \\ -1 & & 1 \\ \hline\end{array} \quad \begin{array}{|ccc|}\hline \sqrt{2} & -1 & \\ -1 & & 1 \\ & 1 & \sqrt{2} \\ \hline\end{array} \quad \begin{array}{|ccc|}\hline -1 & & 1 \\ & & \\ 1 & & -1 \\ \hline\end{array} \quad \begin{array}{|ccc|}\hline -2 & 1 & -2 \\ 1 & 4 & 1 \\ -2 & 1 & -2 \\ \hline\end{array}$$

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.

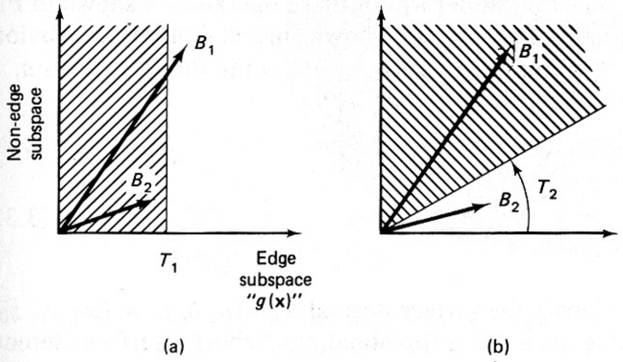


Fig. 3.15 Comparison of thresholding techniques.

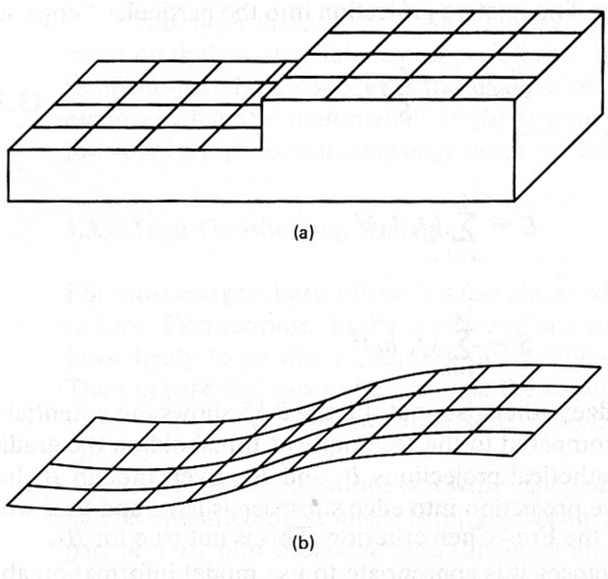


Fig. 3.16 Model-directed edge detection.

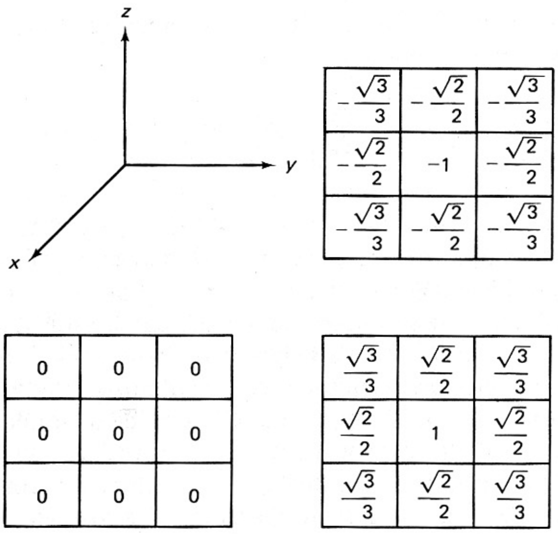
In three-dimensional data, boundaries of objects are surfaces. Edge elements in two dimensions become surface elements in three dimensions. The two-dimensional image gradient, when generalized to three dimensions, is the local surface normal. Just as in the two-dimensional case, many different basis 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

$$g_1(x, y, z) = \frac{x}{r} \qquad (3.31)$$
$$g_2(x, y, z) = \frac{y}{r}$$
$$g_3(x, y, z) = \frac{z}{r}$$

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

$$a = (g_1, f) = \sum_D g_1(\mathbf{x}) f(\mathbf{x}-\mathbf{x}_0)$$
$$b = (g_2, f) \qquad (3.32)$$
$$c = (g_3, f)$$

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

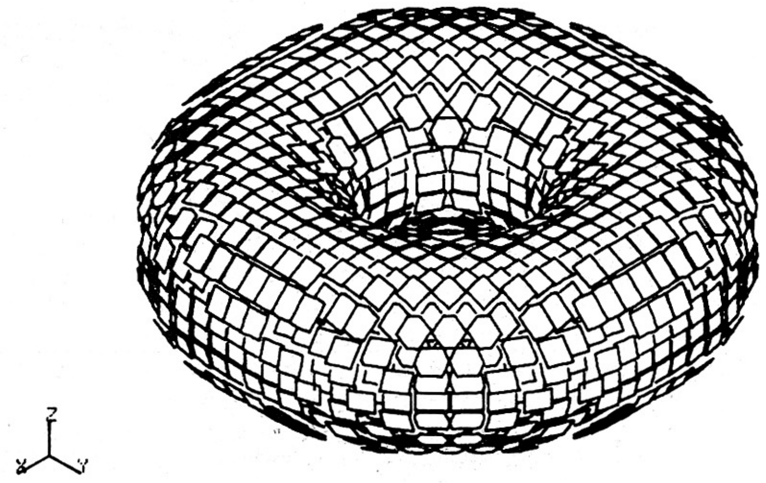


Fig. 3.18 Results of applying the Zucker-Hummel 3-D operator to synthetic image data in the shape of a torus.

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.

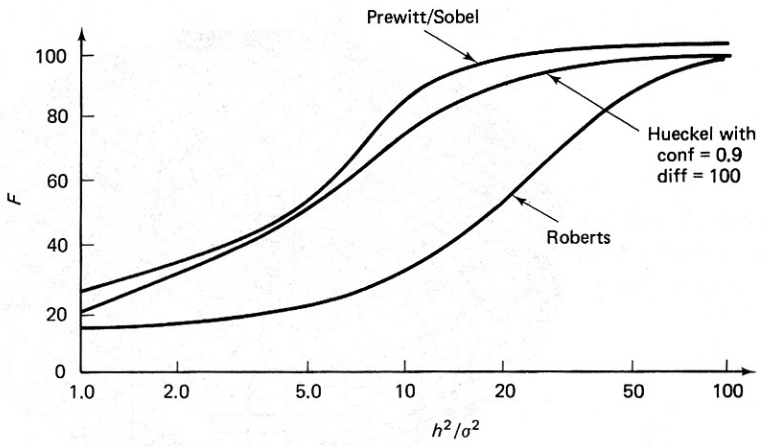


Fig. 3.19 Edge operator performance using Pratt's measure (Eq. 3.33).

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.

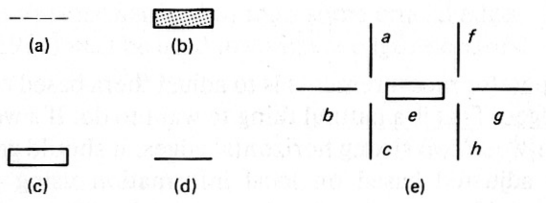


Fig. 3.20 Edge notation. (a) Edge position with no edge. (b) Edge position with edge. (c) Edge to be updated. (d) Edge of unknown strength. (e) Configuration of edges around a central edge e.

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 $i \geqslant j$.)

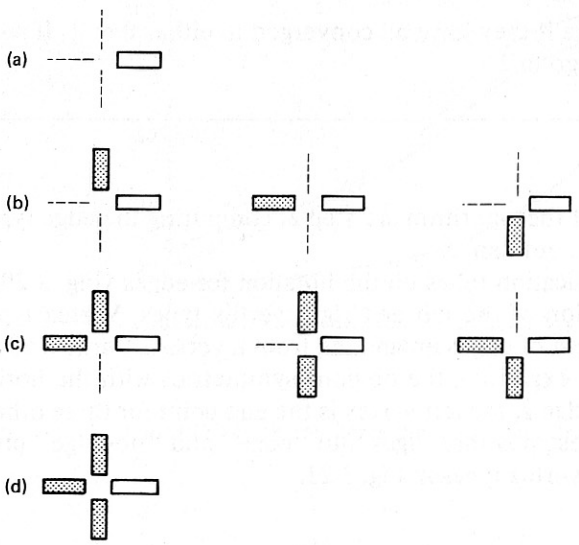


Fig. 3.21 Classification of vertex type of left-hand endpoint of edge e, Fig. 3.20.

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

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

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

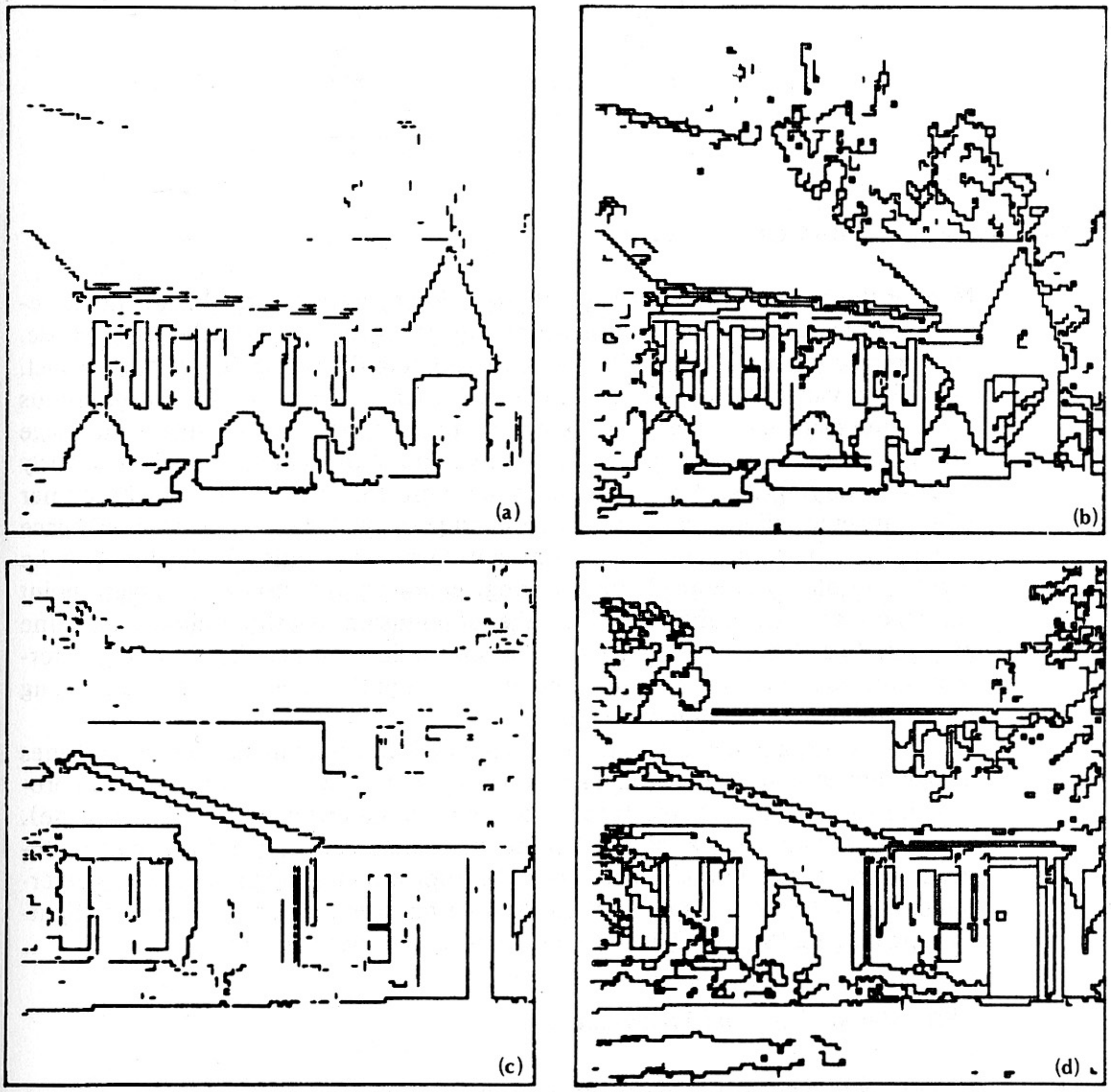


Fig. 3.22 Edge relaxation results. (a) Raw edge data. Edge strengths have been thresholded at 0.25 for display purposes only. (b) Results after five iterations of relaxation applied to (a). (c) Different version of (a). Edge strengths have been thresholded at 0.25 for display purposes only. (d) Results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