CHAPTER 3

Basic Image Filtering Operations

3.1 Introduction

In this chapter the discussion is extended to noise suppression and enhancement in gray-scale images. Although these types of operations can for the most part be avoided in industrial applications of vision, it is useful to examine them in some depth because of their wide use in a variety of other image processing applications and because they set the scene for much of what follows. In addition, some fundamental issues come to light which are of vital importance.

It has already been seen that noise can arise in real images; hence it is necessary to have sound techniques for suppressing it. Commonly, in electrical engineering applications, noise is removed by means of low-pass or other filters that operate in the frequency domain (Rosie, 1966). Applying these filters to 1-D time-varying analog signals is straightforward, since it is necessary only to place them at suitable stages in the sequence of black boxes through which the signals pass. For digital signals, the situation is more complicated, since first the frequency transform of the signal must be computed, then the low-pass filter applied, and finally the signal obtained from the modified transform by converting back to the time domain. Thus, two Fourier transforms have to be computed, although modifying the signal while it is in the frequency domain is a straightforward task (Fig. 3.1). It is by now well known that the amount of processing involved in computing a discrete Fourier transform of a signal represented by N samples is of order N^2 (we shall write this as $O(N^2)$), and that the amount of computation can be cut down to $O(N \log_2 N)$ by employing the fast Fourier transform (FFT) (Gonzalez and Woods, 1992). This then becomes a practical approach for the elimination of noise.

When applying these ideas to images, we must first note that the signal is a spatial rather than a time-varying quantity and must be filtered in the spatial frequency domain. Mathematically, this makes no real difference, but there are nevertheless significant problems. First, there is no satisfactory analog shortcut,

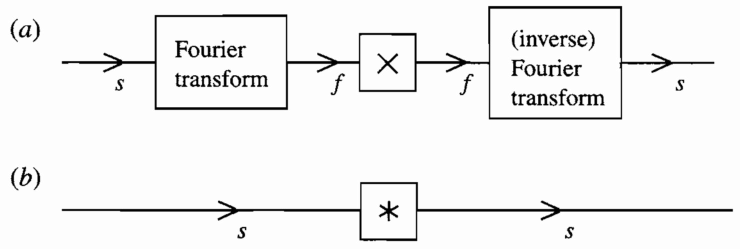


Figure 3.1 Low-pass filtering for noise suppression: s, spatial domain; f, spatial frequency domain; $\times$, multiplication by low-pass characteristic; $*$, convolution with Fourier transform of low-pass characteristics. (*a*) Low-pass filtering achieved most simply, by a process of multiplication in the (spatial) frequency domain; (*b*) low-pass filtering achieved by a process of convolution. Note that (*a*) may require more computation overall because of the two Fourier transforms that have to be performed.

and the whole process has to be carried out digitally. (Here we ignore optical processing methods despite their obvious power, speed, and high resolution, because they are by no means trivial to marry with digital computer technology; also, while there is much ongoing research in this area, it is difficult to anticipate the future position with any accuracy.) Second, for an $N \times N$ pixel image, the number of operations required to compute a Fourier transform is $O(N^3)$ and the FFT only reduces this to $O(N^2 \log_2 N)$, so the amount of computation is quite considerable. (Here it is assumed that the 2-D transforms are implemented by successive passes of 1-D transforms: see Gonzalez and Woods, 1992.) Note also that two Fourier transforms are required for the purpose of noise suppression (Fig. 3.1). Nevertheless, in many imaging applications it is worth proceeding in this way, not only so that noise can be removed but also so that TV scan lines and other artifacts can be filtered out. This situation particularly applies in remote sensing and space technology. However, in industrial applications the emphasis is always on real-time processing; in many cases, therefore, it is not practicable to remove noise by spatial frequency domain operations. A secondary problem is that low-pass filtering is suited to removing Gaussian noise but distorts the image if it is used to remove impulse noise.

Section 3.2 discusses Gaussian smoothing in both the spatial frequency and the spatial domains. The subsequent three sections introduce median filters, mode filters, and then general rank order filters, and contrast their main properties and uses. In Section 3.6 consideration is given to reducing computational load—with particular reference to the median filter. Section 3.7 introduces the sharp–unsharp masking technique, which provides a rather simple route to image enhancement. Then follow a number of sections that concentrate on the edge shifts produced by the various filters. In the case of the median filter, the discrete theory (Section 3.9) is much more exact than the continuum model (Section 3.8). All edge shifts are quite small, except for rank order filters

(Section 3.12); these are treated fairly fully because of their relevance to widely used morphological operators (Chapter 8) where the shifts are turned to advantage.

3.2 Noise Suppression by Gaussian Smoothing

Low-pass filtering is normally thought of as the elimination of signal components with high spatial frequencies; it is therefore natural to carry it out in the spatial frequency domain. Nevertheless, it is possible to implement it directly in the spatial domain. That this is possible is due to the well-known fact (Rosie, 1966) that multiplying a signal by a function in the spatial frequency domain is equivalent to convolving it with the Fourier transform of the function in the spatial domain (Fig. 3.1). If the final convolving function in the spatial domain is sufficiently narrow, then the amount of computation involved will not be excessive. In this way, a satisfactory implementation of the low-pass filter can be sought. It now remains to find a suitable convolving function.

If the low-pass filter is to have a sharp cutoff, then its transform in image space will be oscillatory. An extreme example is the sinc (sin x/x) function, which is the spatial transform of a low-pass filter of rectangular profile (Rosie, 1966). It turns out that oscillatory convolving functions are unsatisfactory since they can introduce halos around objects, hence distorting the image quite grossly. Marr and Hildreth (1980) suggested that the right types of filters to apply to images are those that are well behaved (nonoscillatory) both in the frequency and in the spatial domain. Gaussian filters are able to fulfill this criterion optimally: they have identical forms in the spatial and spatial frequency domains. In 1-D these forms are

$$f(x) = \left[1/(2\pi\sigma^2)^{1/2}\right]\exp\left[-x^2/(2\sigma^2)\right] \tag{3.1}$$

$$F(\omega) = \exp(-\sigma^2\omega^2/2) \tag{3.2}$$

Thus, the type of spatial convolving operator required for the purpose of noise suppression by low-pass filtering is one that approximates to a Gaussian profile. Many such approximations appear in the literature; these vary with the size of the neighborhood chosen and in the precise values of the convolution mask coefficients.

One of the most common masks is the following one, first introduced in Chapter 2, which is used more for simplicity of computation than for its fidelity to a

Gaussian profile:

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

Another commonly used mask, which approximates more closely to a Gaussian profile, is the following:

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

In both cases, the coefficients that precede the mask are used to weight all the mask coefficients. As mentioned in Section 2.2.3, these weights are chosen so that applying the convolution to an image does not affect the average image intensity. These two convolution masks probably account for over 80% of all discrete approximations to a Gaussian. Notice that as they operate within a 3×3 neighborhood, they are reasonably narrow and hence incur a relatively small computational load.

Let us next study the properties of this type of operator, deferring for now consideration of Gaussian operators in larger neighborhoods. First, imagine such an operator applied to a noisy image whose intensity is inherently uniform. Then clearly noise is suppressed, as it is now averaged over 9 pixels. This averaging model is obvious for the first of the two masks above but in fact applies equally to the second mask, once it is accepted that the averaging effect is differently distributed in accordance with the improved approximation to a Gaussian profile.

Although this example shows that noise is suppressed, it will be plain that the signal is also affected. This problem arises only where the signal is initially nonuniform. Indeed, if the image intensity is constant, or if the intensity map approximates to a plane, there is again no problem. However, if the signal is uniform over one part of a neighborhood and rises in another part of it, as is bound to occur adjacent to the edge of an object, then the object will make itself felt at the center of the neighborhood in the filtered image (see Fig. 3.2). As a result, the edges of objects become somewhat blurred. Looking at the operator as a "mixing operator" that forms a new picture by mixing together the intensities of pixels fairly close to each other, it is intuitively obvious why blurring occurs.

It is also apparent, from a spatial frequency viewpoint, why blurring should occur. Basically, we are aiming to give the signal a sharp cutoff in the spatial frequency domain, and as a result it will become slightly blurred in the spatial domain. Clearly, the blurring effect can be reduced by using the narrowest

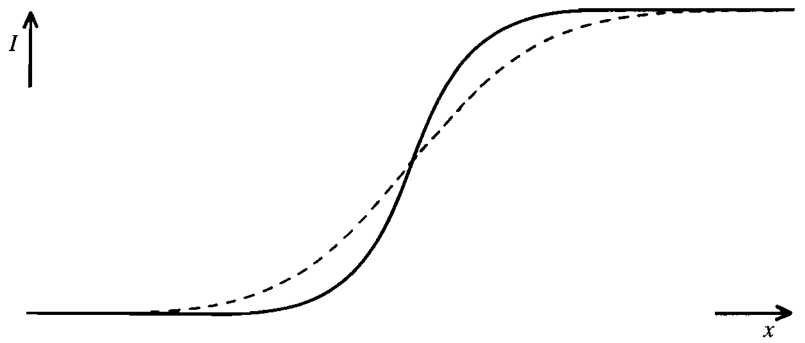


Figure 3.2 Blurring of object edges by simple Gaussian convolutions. The simple Gaussian convolution can be regarded as a gray-scale neighborhood "mixing" operator, hence explaining why blurring arises.

possible approximation to a Gaussian convolution filter, but at the same time the noise suppression properties of the filter are lessened. Assuming that the image was initially digitized at roughly the correct spatial resolution, it will not normally be appropriate to smooth it using convolution masks larger than 3×3 or at most 5×5 pixels. (Here we ignore methods of analyzing images that use a number of versions of the image with different spatial resolutions: see, for example, Babaud et al., 1986.)

Overall, low-pass filtering and Gaussian smoothing are not appropriate for the applications considered here because they introduce blurring effects. Notice also that where interference occurs, which can give rise to impulse or "spike" noise (corresponding to a number of individual pixels having totally the wrong intensities), merely averaging this noise over a larger neighborhood can make the situation worse, since the spikes will be smeared over a sizable number of pixels and will distort the intensity values of all of them. This consideration is important, leading naturally to the concepts of limit and median filtering.

3.3 Median Filters

The idea explored here is to locate those pixels in the image which have extreme and therefore highly improbable intensities and to ignore their actual intensities, replacing them with more suitable values. This is akin to drawing a graph through a set of plots and ignoring those plots that are evidently a long way from the best fit curve. An obvious way of using this technique is to apply a "limit" filter that prevents any pixel from having an intensity outside the intensity range of its neighbors:

```
LIMIT: [[ $\min P = \min(P1, P2, P3, P4, P5, P6, P7, P8)$;
          $\max P = \max(P1, P2, P3, P4, P5, P6, P7, P8)$;
          if $(P0 < \min P)$ $Q0 = \min P$;
          else if $(P0 > \max P)$ $Q0 = \max P$;
          else $Q0 = P0$;
       ]]                                                    (3.3)
```

To develop this technique, it is necessary to examine the local intensity distribution within a particular neighborhood. Points at the extremes of the distribution are quite likely to have arisen from impulse noise. Not only is it sensible to eliminate these points, as in the limit filter, but it is reasonable to try taking the process further, removing equal areas at either end of the distribution and ending with the median. Thus, we arrive at the median filter, which takes all the local intensity distributions and generates a new image corresponding to the set of median values. As the preceding argument indicates, the median filter is excellent at impulse noise suppression; this is amply confirmed in practice (see Fig. 3.3).

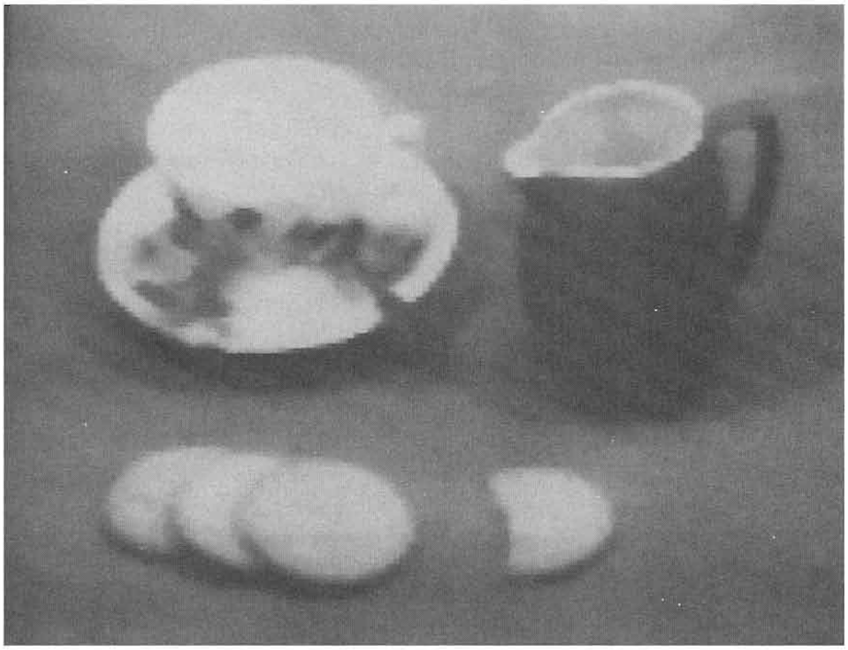


Figure 3.3 Effect of applying a 3×3 median filter to the image of Fig. 2.1*a*. Note the slight loss of fine detail and the rather "softened" appearance of the whole image.

In view of the blurring caused by Gaussian smoothing operators, it is pertinent to ask whether the median filter also induces blurring. Figure 3.3 shows that any blurring is only marginal, although some slight loss of fine detail takes place, which can give the resulting pictures a "softened" appearance. Theoretical discussion of this point is deferred for now. The lack of blur makes good the main deficiency of the Gaussian smoothing filter and results in the median filter being almost certainly the single most widely used filter in general image processing applications.

The median filter may be implemented in many ways: Figure 3.4 reproduces only an obvious algorithm that essentially implements the above description. The notation of Chapter 2 is used but is augmented suitably to permit the nine pixels in a 3×3 neighborhood to be accessed in turn with a running suffix (specifically, P0 to P8 are written as P[m], where m runs from 0 to 8).

The operation of the algorithm is as follows: first, the histogram array is cleared and the image is scanned, generating a new image in Q-space; next, for each neighborhood, the histogram of intensity values is constructed; then the median is found; and, finally, points in the histogram array that have been incremented are cleared. This last feature eliminates the need to clear the whole histogram and hence saves computation. Further savings could be made by finding the extreme intensity values and limiting the search for the median to this range. Unlike the general situation when the median of a distribution is being located, only one (half-) scan through the distribution is required, since the total area is known in advance (in this case it is 9).

As is clear from the above, methods of computing the median involve pixel intensity sorting operations. If a bubble sort (Gonnet, 1984) were used for this purpose, then up to $O(n^4)$ operations would be required for an $n \times n$ neighborhood, compared with some 256 operations for the histogram method we have described. Thus, sorting methods such as the bubble sort are faster for

```
MEDIAN: {
    for (i = 0; i <= 255; i++) hist[i] = 0;
    [[ for (m = 0; m <= 8; m++) hist[ P[m] ]++;
        i = 0; sum = 0;
        while (sum < 5) {
            sum = sum + hist[i];
            i = i + 1;
        }
        Q0 = i - 1;
        for (m = 0; m <= 8; m++) hist[ P[m] ] = 0;
    ]]
}
```

Figure 3.4 An implementation of the median filter.

small neighborhoods where n is 3 or 4 but not for neighborhoods where n is greater than about 5, or where pixel intensity values are more restricted.

Much of the discussion of the median filter in the literature is concerned with saving computation (Narendra, 1978; Huang et al., 1979; Danielsson, 1981). In particular, it has been observed that, on proceeding from one neighborhood to the next, relatively few new pixels are encountered. Thus, the new median value can be found by updating the old value rather than starting from scratch (Huang et al., 1979).

3.4 Mode Filters

Having considered the mean and the median of the local intensity distribution as candidate intensity values for noise smoothing filters, we also find it relevant to consider the mode of the distribution. Indeed, we might imagine that this is, if anything, more important than the mean or the median, since the mode represents the most probable value of any distribution.

A tedious problem arises, however, as soon as we attempt to apply this idea. The local intensity distribution is calculated from relatively few pixel intensity values (Fig. 3.5). This means that instead of a smooth intensity distribution whose mode is easily located, we are almost certain to have a multimodal distribution whose highest point does not indicate the position of the *underlying* mode. Clearly, the distribution needs to be smoothed out considerably before the mode is computed. Another tedious problem is that the width of the distribution varies widely from neighborhood to neighborhood (e.g., from close to zero to close to 256), so that it is difficult to know quite how much to smooth the distribution in any instance. For these reasons, it is likely to be better to choose an indirect measure of the position of the mode rather than to attempt to measure it directly.

The position of the mode can be estimated with reasonable accuracy once the median has been located (Davies, 1984a, 1988c). To understand the technique, it is necessary to consider how local intensity distributions of various sorts arise in practical situations. At most positions in an image, variations in pixel intensity are generated by steady changes in background illumination, or by steady variations in surface orientation, or else by noise. Thus, a symmetrical unimodal local intensity distribution is to be expected. It is well known that the mean, median, and mode are coincident in such cases. More problematic is what happens to the intensity variation near the edge of an object in the image. Here the local intensity distribution is unlikely to be symmetrical, and, more important, it may not even be unimodal. In fact, near an edge the distribution is in general inherently *bimodal*, since the neighborhood contains pixels with intensities

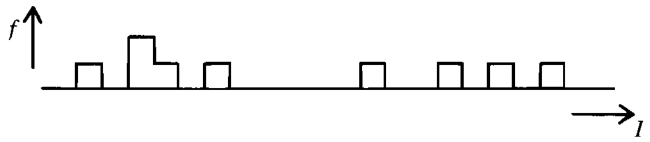


Figure 3.5 The sparse nature of the local intensity histogram for a small neighborhood. This situation clearly causes significant problems for estimation of the mode. It also has definite implications for rigorous estimation of the underlying median, assuming that the observed intensities are only noisy samples of the ideal intensity pattern (see Section 3.8.4).

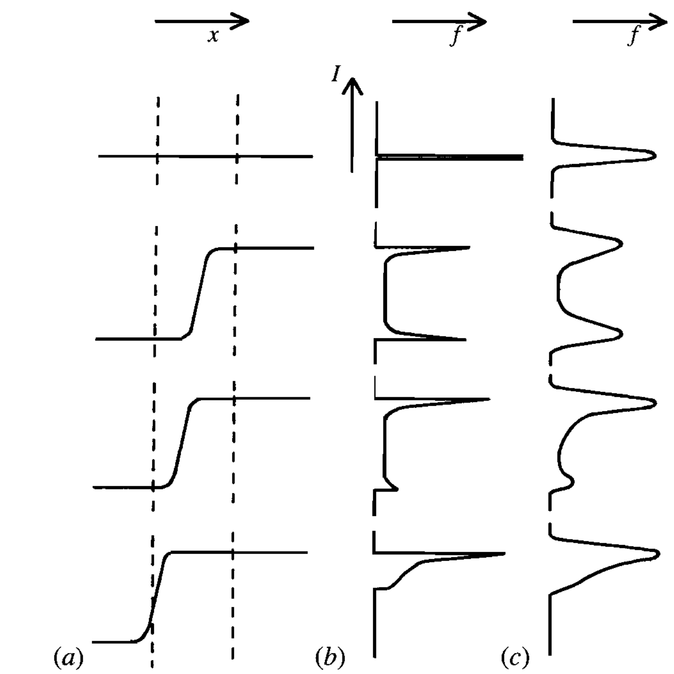


Figure 3.6 Local models of image data near the edge of an object: (*a*) cross sections of an edge falling in the vicinity of a filter neighborhood; (*b*) corresponding local intensity distributions when very little image noise is present; (*c*) situation when the noise level is increased.

corresponding to the values they would have on either side of the edge (Fig. 3.6). Considering the image as a whole, this will be the most likely alternative to a symmetrical unimodal distribution; any further possibilities such as trimodal distributions are rare and of varied causes (e.g., odd glints on the edges of metal objects) and are outside the scope of the present discussion.[1]

1 Here we are ignoring the effects of noise and just considering the underlying image signal.

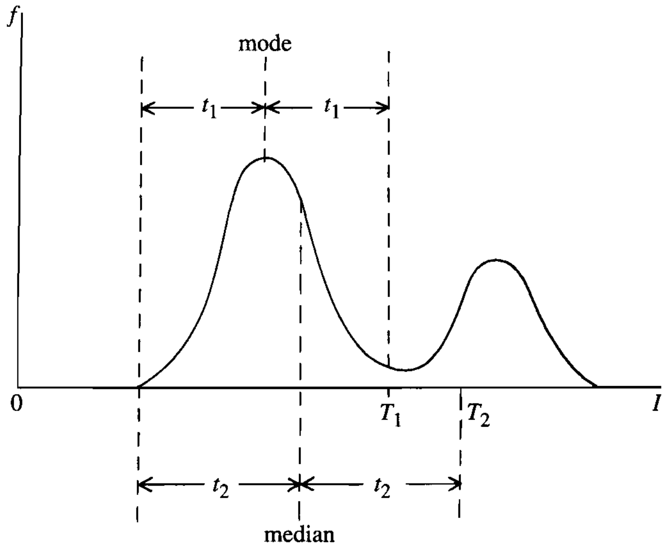


Figure 3.7 Rationale for the method of truncation. The obvious position at which to truncate the distribution is T_1. Since the position of the mode is not initially known, it is suboptimal but safe to truncate instead at T_2.

If the neighborhood straddles an edge and the local intensity distribution is bimodal, the larger peak position should clearly be selected as the most probable intensity value. A good strategy for finding the larger peak is to eliminate the smaller peak. If we knew the position of the mode, we could find where to truncate the smaller peak by first finding which extreme of the distribution was closer to the mode, and then moving an equal distance to the opposite side of the mode (Fig. 3.7). Since we start off *not* knowing the position of the mode, one option is to use the position of the median as an estimator of the position of the mode, and then to use that position to find where to truncate the distribution. Since the three means invariably take the order mean, median, and mode (see Fig. 3.8), except when distributions are badly behaved or multimodal, it turns out that this method is cautious in the sense that it truncates less of the distribution than the required amount: this makes it a safe method to use. When we now find the median of the truncated distribution, the position is much closer to the mode than the original median was, a good proportion of the second peak having been removed (Fig. 3.9). Iteration could be used to find an even closer approximation to the position of the mode. However, the results show that the method gives a marked enhancement in the image even when this is not done (Fig. 3.10).

What has been achieved by the above procedure? When we compare the results of applying a median filter and the new filter (which we call a truncated

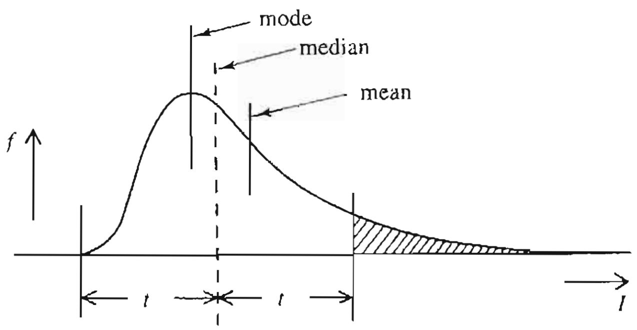


Figure 3.8 Relative positions of the mode, median, and mean for a typical unimodal distribution. This ordering is unchanged for a bimodal distribution, as long as it can be approximated by two Gaussian distributions of similar width.

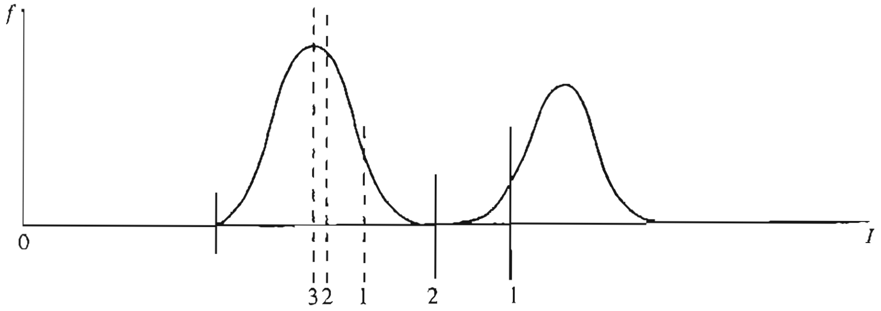


Figure 3.9 Iterative truncation of the local intensity distribution. Here the median converges on the mode within three iterations of the truncation procedure. This is possible since at each stage the mode of the new truncated distribution remains the same as that of the previous distribution.

median filter, for it is not a true mode filter—see below), we find that whereas the median filter is highly successful at removing noise, the new filter not only removes noise but also enhances the image so that edges are made sharper. By reference to Fig. 3.11 it is quite easy to see why this should happen. Basically, at a location even very slightly to one side of an edge, a majority of the pixel intensities contribute to the larger peak and the filter ignores the pixel intensities contributing to the smaller peak. Thus, the filter makes an informed binary choice about which side of the edge it is on. At first this seems to mean that it pushes a nearby edge further away. However, it must be remembered that it actually "pushes the edge away" from both sides, and the result is that its sides are made sharper and object outlines are crispened up. Particularly striking is the effect of applying this filter to an image a number of times, when objects start to

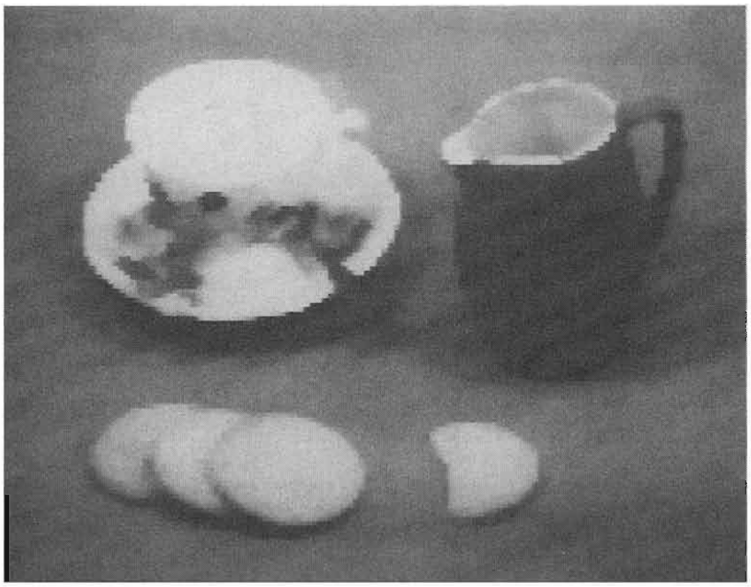


Figure 3.10 Effect of a single application of 3 × 3 truncated median filter to the image of Fig. 2.1*a*.

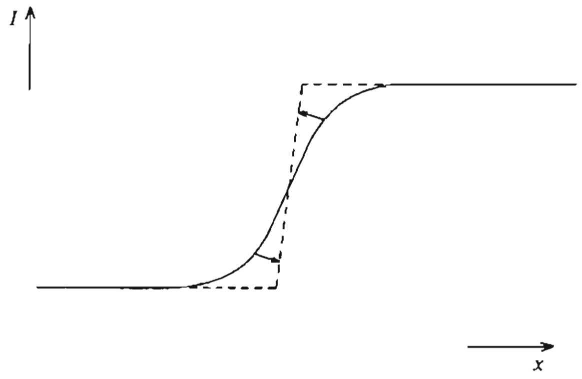


Figure 3.11 Image enhancement performed by the mode filter. Here the onset of the edge is pushed laterally by the action of the mode filter within one neighborhood. Since the same happens from the other side within an adjacent neighborhood, the actual position of the edge is unchanged in first order. The overall effect is to sharpen the edge.

become segmented into regions of fairly uniform intensity (Fig. 3.12). The complete algorithm for achieving this is outlined in Fig. 3.13.

We have dealt with this problem at some length for several reasons. First, it seems that the mode filter has not hitherto received the attention it deserves. Second, the median filter seems to be used fairly universally, often without very

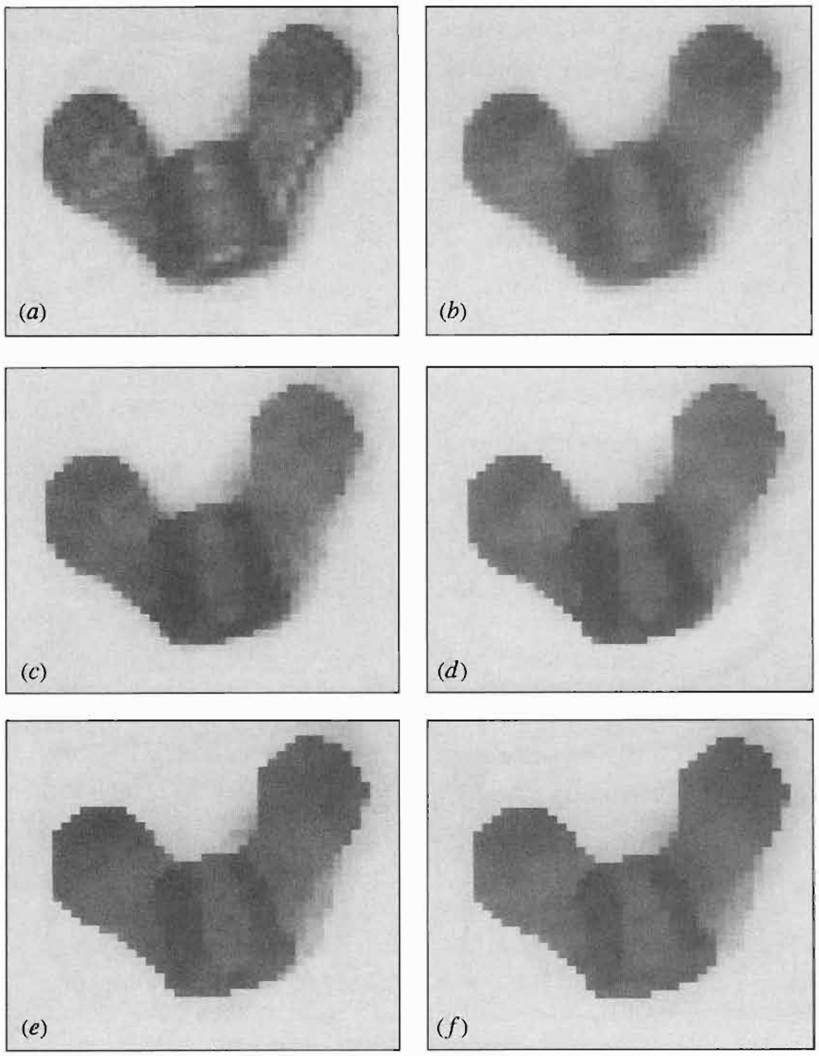


Figure 3.12 Results of repeated action of the truncated median filter: (*a*) the original, moderately noisy picture; (*b*) effect of a 3 × 3 median filter; (*c*)–(*f*) effect of 1–4 passes of the basic truncated median filter, respectively.

much justification or thought. Third, all these filters show what markedly different characteristics are available merely by analyzing the contents of the local intensity distribution and ignoring totally where in the neighborhood the different intensities appear. It is perhaps remarkable that there is sufficient information in the local intensity distribution for this to be possible. All this shows the danger of applying operators that have been derived in an ad hoc manner without first

```
do { // as many passes over image as necessary
   do { // for each pixel
      compute local intensity distribution;
      do { // iterate to improve estimate of mode
         find minimum, median, and maximum intensity values;
         decide from which end local intensity distribution should be truncated;
         deduce where local intensity distribution should be truncated;
         truncate local intensity distribution;
         find median of truncated local intensity distribution;
      } until median sufficiently close to mode of local distribution;
      transfer estimate of mode to output image space;
   } until all pixels processed;
} until sufficient enhancement of image;
```

Figure 3.13 Outline of algorithm for implementing the truncated median filter. In this algorithm (i) The outermost and innnermost loops can normally be omitted (i.e., they need to be executed once only). (ii) The *final* estimate of the position of the mode can be performed by simple averaging instead of computing the median: this has been found to save computation with negligible loss of accuracy. (iii) Instead of the minimum and maximum intensity values, the positions of the outermost octiles (for example) may be used to give more stable estimates of the extremes of the local intensity distribution.

making a specification of what is required and then designing an operator with the required characteristics. In fact, it appears that if we want a filter that has maximum impulse noise suppression capability, then we should use a median filter; and if we want a filter that enhances images by sharpening edges, then we should use a mode or truncated median filter. (Note that the truncated median filter should in some ways be an improvement on the mode filter in that it is more cautious when very close to an edge transition, where noise prevents an exact judgement from being made as to which side of the edge a pixel is on: see Davies, 1984a, 1988c.)

While considering enhancement, attention has been restricted to filters based on the local intensity distribution; many filters enhance images without the aid of the local intensity distribution (Lev et al., 1977; Nagao and Matsuyama, 1979), but they are not within the scope of this chapter. Note that the method of "sharp–unsharp masking" (Section 3.7) performs an enhancement function, although its main purpose is to restore images that have inadvertently become blurred, for example, by a hazy atmosphere or defocussed camera.

Finally, although this section has concentrated on the gray-scale properties of mode filters, Charles and Davies (2003a, 2004) have recently shown how to devise versions of the mode filter that operate on color images. Typical results are shown in Fig. 2.8. In addition, Fig. 2.9 shows that the mode filter has the hitherto unexpected property of being able to eliminate very large amounts of impulse noise from images—significantly more than a median filter—in spite of being designated as an image enhancement filter.

3.5 Rank Order Filters

The principle employed in rank order filters is to take all the intensity values in a given neighborhood; place these in order of increasing value; and finally select the rth of the n values and return this value as the filter local output value. Clearly, n rank order filters can be specified in terms of the value r that is used, but these filters are all intrinsically nonlinear; that is, the output intensity cannot be expressed as a linear sum of the component intensities within the neighborhood. In particular, the median filter (for which $r = (n+1)/2$, and which is only defined if n is odd[2]) does not normally give the same output image as a mean filter. Indeed, it is well known that the mean and median of a distribution are in general only coincident for symmetrical distributions. Note that minimum and maximum filters (corresponding to $r = 1$ and $r = n$, respectively) are also often classed as morphological filters (see Chapter 8).

3.6 Reducing Computational Load

Significant efforts have been made to speed the operation of the Gaussian filter since implementations in large neighborhoods require considerable amounts of computation (Wiejak et al., 1985). For example, smoothing images using a 30×30 Gaussian convolution mask in an image of 256×256 pixels involves 64 million basic operations. For such a basic operation as smoothing, this is unacceptable. However, the amount of computation can be cut down drastically, since a 2-D Gaussian convolution can be factorized into two 1-D Gaussian convolutions which can be applied in turn:

$$\exp(-r^2/2\sigma^2) = \exp(-x^2/2\sigma^2)\exp(-y^2/2\sigma^2) \tag{3.4}$$

The decomposition is rigorously provable and is not an approximation; we shall refer to this below in the context of the median filter. Meanwhile, the decompositions for the two 3×3 Gaussian filters we met earlier are:

$$\frac{1}{9}\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} = \frac{1}{3}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix}\frac{1}{3}\begin{bmatrix} 1 & 1 & 1 \end{bmatrix} \tag{3.5}$$

2 If n is even, the mean of the central two values in the distribution is usually taken as representing the median.

CHAPTER 4

Thresholding Techniques

4.1 Introduction

One of the first tasks to be undertaken in vision applications is to segment objects from their background. When objects are large and do not possess very much surface detail, segmentation is often imagined as splitting the image into a number of regions, each having a high level of uniformity in some parameter such as brightness, color, texture, or even motion. Hence, it should be straightforward to separate objects from one another and from their background, as well as to discern the different facets of solid objects such as cubes.

Unfortunately, this concept of segmentation is an idealization that is sometimes reasonably accurate, but more often in the real world it is an invention of the human mind, generalized inaccurately from certain simple cases. This problem arises because of the ability of the eye to understand real scenes at a glance, and hence to segment and perceive objects within images in the form they are known to have. Introspection is not a good way of devising vision algorithms, and it must not be overlooked that segmentation is actually one of the central and most difficult practical problems of machine vision.

Thus, the common view of segmentation as looking for regions possessing some degree of uniformity is to a large extent invalid. There are many examples of this in the world of 3-D objects. One example is a sphere lit from one direction, the brightness in this case changing continuously over the surface so that there is no distinct region of uniformity. Another is a cube where the direction of the lighting may lead to several of the facets having equal brightness values, so that it is impossible from intensity data alone to segment the image completely as desired.

Nevertheless, there is sufficient correctness in the concept of segmentation by uniformity measures for it to be worth pursuing for practical applications. The reason is that in many (especially industrial) applications only a very restricted range and number of objects are involved. In addition, it is possible to have almost complete control over the lighting and the general environment. The fact that a particular method may not be completely general need not be problematic, since by employing tools that are appropriate for the task at hand, a cost-effective solution will have been achieved in that case at least. However, in practical situations a tension exists between the simple cost-effective solution and the general-purpose but more computationally expensive solution. This tension must always be kept in mind in severely practical subjects such as machine vision.

4.2 Region-growing Methods

The segmentation idea as we have outlined it leads naturally to the region-growing technique (Zucker, 1976b). Here pixels of like intensity (or other suitable property) are successively grouped together to form larger and larger regions until the whole image has been segmented. Clearly, there have to be rules about not combining adjacent pixels that differ too much in intensity, while permitting combinations for which intensity changes gradually because of variations in background illumination over the field of view. However, this is not enough to make a viable strategy, and in practice the technique has to include the facility not only to merge regions together but also to split them if they become too large and inhomogeneous (Horowitz and Pavlidis, 1974). Particular problems are noise and sharp edges and lines that form disconnected boundaries, and for which it is difficult to formulate simple criteria to decide whether they form true region boundaries. In remote sensing applications, for example, it is often difficult to separate fields rigorously when edges are broken and do not give continuous lines. In such applications segmentation may have to be performed interactively, with a human operator helping the computer. Hall (1979) found that, in practice, regions tend to grow too far,[1] so that to make the technique work well it is necessary to limit their growth with the aid of edge detection schemes.

Thus, the region-growing approach to segmentation turns out to be quite complex to apply in practice. In addition, region-growing schemes usually operate iteratively, gradually refining hypotheses about which pixels belong to which regions. The technique is complicated because, carried out properly, it involves global as well as local image operations. Thus, each pixel intensity will in principle have to be examined many times, and as a result the process tends to be quite computationally intensive. For this reason it is not considered further here, since we are particularly interested in methods involving low computational load which are amenable to real-time implementation.

1 The danger is clearly that even one small break will in principle join two regions into a single larger one.

4.3 Thresholding

If background lighting is arranged so as to be fairly uniform, and we are looking for rather flat objects that can be silhouetted against a contrasting background, segmentation can be achieved simply by thresholding the image at a particular intensity level. This possibility was apparent from Fig. 2.3. In such cases, the complexities of the region-growing approach are bypassed. The process of thresholding has already been covered in Chapter 2, the basic result being that the initial gray-scale image is converted into a binary image in which objects appear as black figures on a white background or as white figures on a black background. Further analysis of the image then devolves into analysis of the shapes and dimensions of the figures. At this stage, object identification should be straightforward. Chapter 6 concentrates on such tasks. Meanwhile there is one outstanding problem—how to devise an automatic procedure for determining the optimum thresholding level.

4.3.1 *Finding a Suitable Threshold*

One simple technique for finding a suitable threshold arises in situations such as optical character recognition (OCR) where the proportion of the background that is occupied by objects (i.e., print) is substantially constant in a variety of conditions. A preliminary analysis of relevant picture statistics then permits subsequent thresholds to be set by insisting on a fixed proportion of dark and light in a sequence of images (Doyle, 1962). In practice, a series of experiments is performed in which the thresholded image is examined as the threshold is adjusted, and the best result is ascertained by eye. At that stage, the proportions of dark and light in the image are measured. Unfortunately, any changes in noise level following the original measurement will upset such a scheme, since they will affect the relative amounts of dark and light in the image. However, this technique is frequently useful in industrial applications, especially when particular details within an object are to be examined. Typical examples are holes in mechanical components such as brackets. (Notice that the mark–space ratio for objects may well vary substantially on a production line, but the proportion of hole area *within* the object outline would not be expected to vary.)

The technique that is most frequently employed for determining thresholds involves analyzing the histogram of intensity levels in the digitized image (see Fig. 4.1). If a significant minimum is found, it is interpreted as the required threshold value (Weska, 1978). The assumption being made here is that the peak on the left of the histogram corresponds to dark objects, and the peak on the

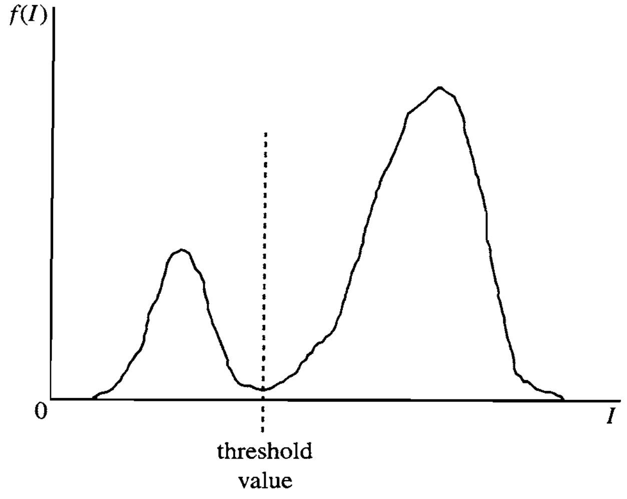


Figure 4.1 Idealized histogram of pixel intensity levels in an image. The large peak on the right results from the light background; the smaller peak on the left is due to dark foreground objects. The minimum of the distribution provides a convenient intensity value to use as a threshold.

right corresponds to light background (i.e., it is assumed that, as in many industrial applications, objects appear dark on a light background).

This method is subject to the following major difficulties:

1. The valley may be so broad that it is difficult to locate a significant minimum.
2. There may be a number of minima because of the type of detail in the image, and selecting the most significant one will be difficult.
3. Noise within the valley may inhibit location of the optimum position.
4. There may be no clearly visible valley in the distribution because noise may be excessive or because the background lighting may vary appreciably over the image.
5. Either of the major peaks in the histogram (usually that due to the background) may be much larger than the other, and this will then bias the position of the minimum.
6. The histogram may be inherently multimodal, making it difficult to determine which is the relevant thresholding level.

Perhaps the worst of these problems is the last: if the histogram is inherently multimodal, and we are trying to employ a single threshold, we are applying what is essentially an ad hoc technique to obtain a meaningful result. In general, such efforts are unlikely to succeed, and this is clearly a case where full image interpretation must be performed before we can be satisfied that the results are valid. Ideally, thresholding rules have to be formed after many images have been analyzed. In what follows, such problems of meaningfulness are eschewed. Instead attention is concentrated on how best to find a genuine single threshold when its position is obscured, as suggested by problems (1)–(6) above (which can be ascribed to image "clutter," noise, and lighting variations).

4.3.2 *Tackling the Problem of Bias in Threshold Selection*

This subsection considers problem (5) above—that of eliminating the bias in the selection of thresholds that arises when one peak in the histogram is larger than the other. First, note that if the relative heights of the peaks are known, this effectively eliminates the problem, since the "fixed proportion" method of threshold selection outlined above can be used. However, this is not normally possible. A more useful approach is to prevent bias by weighting down the extreme values of the intensity distribution and weighting up the intermediate values in some way. To achieve this effect, note that the intermediate values are special in that they correspond to object edges. Hence, a good basic strategy is to find positions in the image where there are significant intensity gradients—corresponding to pixels in the regions of edges—and to analyze the intensity values of these locations while ignoring other points in the image.

To develop this strategy, consider the "scattergram" of Fig. 4.2. Here pixel properties are plotted on a 2-D map with intensity variation along one axis and intensity gradient magnitude variation along the other. Broadly speaking, three main populated regions are seen on the map: (1) a low-intensity, low-gradient region corresponding to the dark objects; (2) a high-intensity, low-gradient region corresponding to the background; and (3) a medium-intensity, high-gradient region corresponding to object edges (Panda and Rosenfeld, 1978). These three regions merge into each other, with the high-gradient region that corresponds to edges forming a path from the low-intensity to the high-intensity region. The fact that the three regions merge in this way gives a choice of methods for selecting intensity thresholds. The first choice is to consider the intensity distribution along the zero gradient axis, but this alternative has the same disadvantage as the original histogram method—namely, introducing a substantial bias. The second choice is that at high gradient, which means looking for a peak rather than a valley in the intensity distribution. The final choice is that at moderate values of gradient, which was suggested earlier as a possible strategy for avoiding bias.

It is now apparent that there is likely to be quite a narrow range of values of gradient for which the third method will be a genuine improvement. In addition, it may be necessary to construct a scattergram rather than a simple intensity

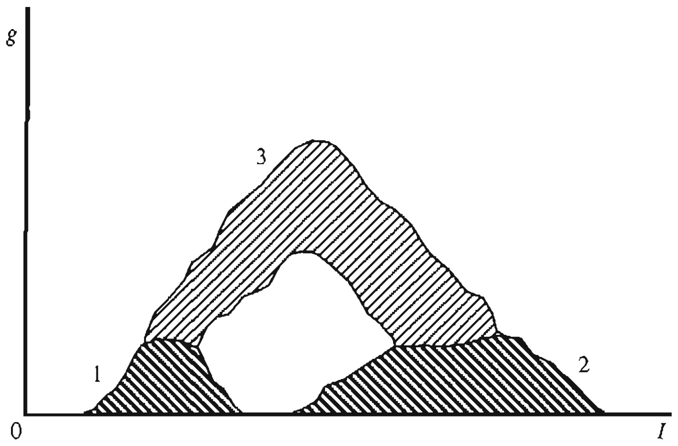


Figure 4.2 Scattergram showing the frequency of occurrence of various combinations of pixel intensity I and intensity gradient magnitude g in an idealized image. There are three main populated regions of interest: (1) a low I, low g region; (2) a high I, low g region; and (3) a medium I, high g region. Analysis of the scattergram provides useful information on how to segment the image.

histogram in order to locate it. If, instead, we try to weight the plots on a histogram with the magnitudes of the intensity gradient values, we are likely to end with a peaked intensity distribution (i.e., method 2) rather than one with a valley (Weska and Rosenfeld, 1979). A mathematical model of the situation is presented in Section 4.3.3 in order to explore the situation more fully.

4.3.2.1 *Methods Based on Finding a Valley in the Intensity Distribution*

This section considers how to weight the intensity distribution using a parameter other than the intensity gradient, in order to locate accurately the valley in the intensity distribution. A simple strategy is first to locate all pixels that have a significant intensity gradient and then to find the intensity histogram not only of these pixels but also of nearby pixels. This means that the two main modes in the intensity distribution are still attenuated very markedly and hence the bias in the valley position is significantly reduced. Indeed, the numbers of background and foreground pixels that are now being examined are very similar, so the bias from the relatively large numbers of background pixels is virtually eliminated. (Note that if the modes are modeled as two Gaussian distributions of equal widths and they also have equal heights, then the minimum lies exactly halfway between them.)

CHAPTER 5

Edge Detection

5.1 Introduction

In Chapter 4 segmentation was tackled by the general approach of finding regions of uniformity in images—on the basis that the areas found in this way would have a fair likelihood of coinciding with the surfaces and facets of objects. The most computationally efficient means of following this approach was that of thresholding, but this turns out for real images to be failure-prone or else quite difficult to implement satisfactorily. Indeed, making it work well seems to require a multiresolution or hierarchical approach, coupled with sensitive measures for obtaining suitable local thresholds. Such measures have to take account of local intensity gradients as well as pixel intensities, and the possibility of proceeding more simply—by taking account of intensity gradients alone—was suggested.

Edge detection has long been an alternative path to image segmentation, and it is the method pursued in this chapter. Whichever way is inherently the better approach, edge detection has a considerable additional advantage in that it immediately reduces by a large factor (typically around 100) the redundancy inherent in the image data. This is useful because it immensely reduces both the space needed to store the information and the amount of processing subsequently required to analyze it.

Edge detection has gone through an evolution spanning more than 30 years. Two main methods of edge detection have been apparent over this period, the first of these being template matching (TM) and the second being the differential gradient (DG) approach. In either case, the aim is to find where the intensity gradient magnitude g is sufficiently large to be taken as a reliable indicator of the edge of an object. Then g can be thresholded in a similar way to that in which intensity was thresholded in Chapter 4. (It is possible to look for local maxima of g instead of thresholding it, but this possibility and its particular complexities are ignored until a later section.) The TM and DG methods differ mainly in how they proceed to estimate g locally. However, important differences exist in how they determine local edge orientation, which is an important variable in certain object detection schemes.

5.2 Basic Theory of Edge Detection

Both DG and TM operators estimate local intensity gradients with the aid of suitable convolution masks. In the case of the DG type of operator, only two such masks are required—for the x and y directions. In the TM case, it is usual to employ up to 12 convolution masks capable of estimating local components of gradient in the different directions (Prewitt, 1970; Kirsch, 1971; Robinson, 1977; Abdou and Pratt, 1979).

In the TM approach, the local edge gradient magnitude (for short, the edge "magnitude") is approximated by taking the maximum of the responses for the component masks:

$$g = \max(g_i\colon\ i = 1, \ldots, n) \tag{5.1}$$

where n is usually 8 or 12.

In the DG approach, the local edge magnitude may be computed vectorially using the nonlinear transformation:

$$g = \left(g_x^2 + g_y^2\right)^{1/2} \tag{5.2}$$

In order to save computational effort, it is common practice (Abdou and Pratt, 1979) to approximate this formula by one of the simpler forms:

$$g = |g_x| + |g_y| \tag{5.3}$$

or

$$g = \max\left(|g_x|, |g_y|\right) \tag{5.4}$$

which are, on average, equally accurate (Föglein, 1983).

In the TM approach, edge orientation is estimated simply as that of the mask giving rise to the largest value of gradient in equation (5.1). In the DG approach, it is estimated vectorially by the more complex equation:

$$\theta = \arctan(g_y/g_x) \tag{5.5}$$

(*a*) Masks for the Roberts 2 × 2 operator:

$$R_{x'} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \qquad R_{y'} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$$

(*b*) Masks for the Sobel 3 × 3 operator:

$$S_x = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad S_y = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix}$$

(*c*) Masks for the Prewitt 3 × 3 "smoothed gradient" operator:

$$P_x = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad P_y = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Figure 5.1 Masks of well-known differential edge operators. In this figure masks are presented in an intuitive format (viz. coefficients increasing in the positive x and y directions) by rotating the normal convolution format through 180°. This convention is employed throughout this chapter. The Roberts 2 × 2 operator masks (*a*) can be taken as being referred to axes x', y' at 45° to the usual x, y.

DG equations (5.2) and (5.5) are considerably more computationally intensive than TM equation (5.1), although they are also more accurate. In some situations, however, orientation information is not required. In addition, image contrast may vary widely, so there may be little to be gained from thresholding a more accurate estimate of g. This may explain why so many workers have employed the TM instead of the DG approach. Since both approaches essentially involve estimation of local intensity gradients, it is not surprising that TM masks often turn out to be identical to DG masks—see Figs. 5.1 and 5.2.

5.3 The Template Matching Approach

Figure 5.2 shows four sets of well-known TM masks for edge detection. These masks were originally (Prewitt, 1970; Kirsch, 1971; Robinson, 1977) introduced on an intuitive basis, starting in two cases from the DG masks shown in Fig. 5.1. In all cases, the eight masks of each set are obtained from a given mask by permuting the mask coefficients cyclically. By symmetry, this is a good strategy for even permutations, but symmetry alone does not justify it for odd permutations. The situation is explored in more detail below.

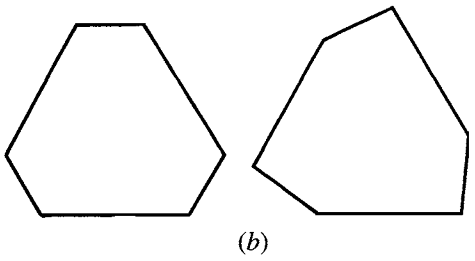


Figure 14.8 (*a*) Semiregular hexagon with threefold rotation symmetry and having all its exterior angles equal: two planes are required in parameter space in this case; (*b*) a similar hexagon with two slightly different exterior angles. Here three planes are needed in parameter space.

This method is wasteful in computation, however, since it does not make use of the fact that the main peak in parameter space has been found to lie in a *particular* plane K. Hence, we can take the set of (averaged) polygon side orientations θ from the histogram already constructed, identify which sides had which orientations, deduce from each θ a value for the polygon orientation $\eta=\alpha+\theta-\varphi_i$ (see Section 14.3.3), and enter this value into a histogram of estimated η values. The peak in this histogram will eventually give an accurate value for the polygon orientation η.

Strong similarities, but also significant differences, are evident between the procedure described above and that described in Chapter 13 for the determination of ellipse parameters. In particular, both procedures are two-stage ones, in which it is computationally more efficient to find object location before object orientation.

14.5 Why Corner Detection?

Our analysis has shown the possibilities for direct detection of polygons, but it has also revealed the limitations of the approach: these arise specifically with more complex polygons—such as those having concavities or many sides. In such cases, it may be easier to revert to using line or corner detection schemes coupled with the abstract pattern matching approaches discussed in Chapter 15.

In previous chapters we noted that objects are generally located most efficiently from their features. Prominent features have been seen to include straight lines, arcs, holes, and corners. Corners are particularly important since they may be used to orient objects and to provide measures of their dimensions—a knowledge of orientation being vital on some occasions, for example if a robot is to find the best way of picking up the object, and measurement being necessary in most inspection applications. Thus, accurate, efficient corner detectors are of great relevance in machine vision. In the remainder of the chapter we shall focus on corner detection procedures.

Workers in this field have devoted considerable ingenuity to devising corner detectors, sometimes with surprising results (see, for example, the median-based detector described later in this chapter). We start by considering what is perhaps the most obvious detection scheme—that of template matching.

14.6 Template Matching

Following our experience with template matching methods for edge detection (Chapter 5), it would appear to be straightforward to devise suitable templates for corner detection. These would have the general appearance of corners, and in a 3×3 neighborhood they would take forms such as the following:

$$\begin{bmatrix} -4 & 5 & 5 \\ -4 & 5 & 5 \\ -4 & -4 & -4 \end{bmatrix} \quad \begin{bmatrix} 5 & 5 & 5 \\ -4 & 5 & -4 \\ -4 & -4 & -4 \end{bmatrix}$$

the complete set of eight templates being generated by successive 90° rotations of the first two shown. Bretschi (1981) proposed an alternative set of templates. As for edge detection templates, the mask coefficients are made to sum to zero so that corner detection is insensitive to absolute changes in light intensity. Ideally, this set of templates should be able to locate all corners and to estimate their orientation to within 22.5°.

Unfortunately, corners vary considerably in a number of their characteristics, including in particular their degree of pointedness,[1] internal angle, and the intensity gradient at the boundary. Hence, it is quite difficult to design optimal corner detectors. In addition, generally corners are insufficiently pointed for good results to be obtained with the 3×3 template masks shown above. Another problem is that in larger neighborhoods, not only do the masks become larger but also more of them are needed to obtain optimal corner responses, and it rapidly becomes clear that the template matching approach is likely to involve excessive computation for practical corner detection. The alternative is to approach the problem analytically, somehow deducing the ideal response for a corner at any arbitrary orientation, and thereby bypass the problem of calculating

1 The term *pointedness* is used as the opposite to *bluntness*, the term *sharpness* being reserved for the total angle η through which the boundary turns in the corner region, that is, π minus the internal angle.

many individual responses to find which one gives the maximum signal. The methods described in the remainder of this chapter embody this alternative philosophy.

14.7 Second-order Derivative Schemes

Second-order differential operator approaches have been used widely for corner detection and mimic the first-order operators used for edge detection. Indeed, the relationship lies deeper than this. By definition, corners in gray-scale images occur in regions of rapidly changing intensity levels. By this token, they are detected by the same operators that detect edges in images. However, corner pixels are much rarer[2] than edge pixels. By one definition, they arise where two relatively straight-edged fragments intersect. Thus, it is useful to have operators that detect corners *directly*, that is, *without unnecessarily locating edges*. To achieve this sort of discriminability, it is necessary to consider local variations in image intensity up to at least second order. Hence, the local intensity variation is expanded as follows:

$$I(x,y) = I(0,0) + I_x x + I_y y + I_{xx}x^2/2 + I_{xy}xy + I_{yy}y^2/2 + \cdots \tag{14.6}$$

where the suffices indicate partial differentiation with respect to x and y and the expansion is performed about the origin X_0 $(0,0)$. The symmetrical matrix of second derivatives is:

$$\mathscr{I}_{(2)} = \begin{bmatrix} I_{xx} & I_{xy} \\ I_{yx} & I_{yy} \end{bmatrix} \quad \text{where } I_{xy} = I_{yx} \tag{14.7}$$

This gives information on the local curvature at X_0. A suitable rotation of the coordinate system transforms $\mathscr{I}_{(2)}$ into diagonal form:

$$\tilde{\mathscr{I}}_{(2)} = \begin{bmatrix} I_{\tilde{x}\tilde{x}} & 0 \\ 0 & I_{\tilde{y}\tilde{y}} \end{bmatrix} = \begin{bmatrix} \kappa_1 & 0 \\ 0 & \kappa_2 \end{bmatrix} \tag{14.8}$$

where appropriate derivatives have been reinterpreted as principal curvatures at X_0.

2 We might imagine a 256×256 image of 64K pixels, of which 1000 ($\sim$2%) lie on edges and a mere 30 ($\sim$0.05%) are situated at corner points.

We are particularly interested in rotationally invariant operators, and it is significant that the trace and determinant of a matrix such as $\mathscr{I}_{(2)}$ are invariant under rotation. Thus we obtain the Beaudet (1978) operators:

$$\text{Laplacian} = I_{xx} + I_{yy} = \kappa_1 + \kappa_2 \tag{14.9}$$

and

$$\text{Hessian} = \det(\mathscr{I}_{(2)}) = I_{xx}I_{yy} - I_{xy}^2 = \kappa_1\kappa_2 \tag{14.10}$$

It is well known that the Laplacian operator gives significant responses along lines and edges and hence is not particularly suitable as a corner detector. On the other hand, Beaudet's "DET" (determinant) operator does not respond to lines and edges but gives significant signals near corners. It should therefore form a useful corner detector. However, DET responds with one sign on one side of a corner and with the opposite sign on the other side of the corner. At the point of real interest—on the corner—it gives a null response. Hence, rather more complicated analysis is required to deduce the presence and exact position of each corner (Dreschler and Nagel, 1981; Nagel, 1983). The problem is clarified by Fig. 14.9. Here the dotted line shows the path of maximum horizontal curvature for various intensity values up the slope. The DET operator gives maximum response at positions P and Q on this line, and the parts of the line between P and Q must be explored to find the "ideal" corner point C where DET is zero.

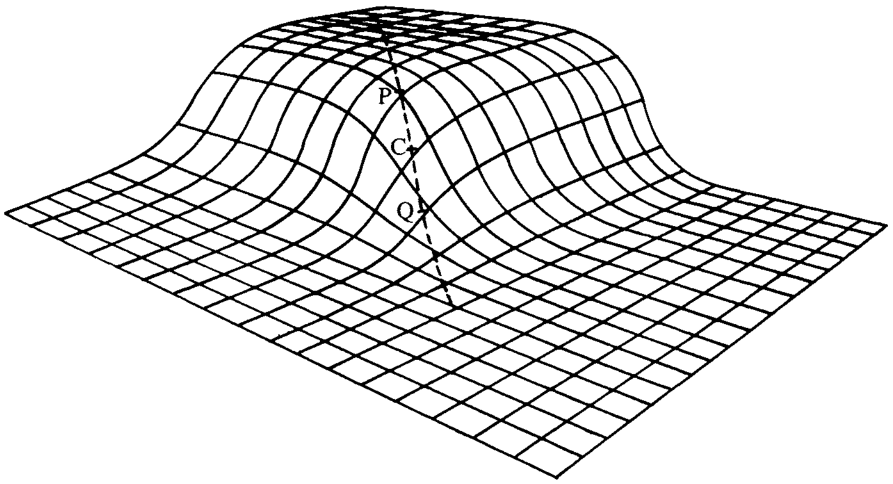


Figure 14.9 Sketch of an idealized corner, taken to give a smoothly varying intensity function. The dotted line shows the path of maximum horizontal curvature for various intensity values up the slope. The DET operator gives maximum responses at P and Q, and it is required to find the ideal corner position C where DET gives a null response.

Perhaps to avoid rather complicated procedures of this sort, Kitchen and Rosenfeld (1982) examined a variety of strategies for locating corners, starting from the consideration of local variation in the directions of edges. They found a highly effective operator that estimates the projection of the local rate of change of gradient direction vector along the horizontal edge tangent direction, and showed that it is mathematically identical to calculating the horizontal curvature κ of the intensity function I. To obtain a realistic indication of the strength of a corner, they multiplied κ by the magnitude of the local intensity gradient g:

$$\begin{aligned} C = \kappa g &= \kappa\left(I_x^2 + I_y^2\right)^{1/2} \\ &= \frac{I_{xx}I_y^2 - 2I_{xy}I_xI_y + I_{yy}I_x^2}{I_x^2 + I_y^2} \end{aligned} \tag{14.11}$$

Finally, they used the heuristic of nonmaximum suppression along the edge normal direction to localize the corner positions further.

In 1983, Nagel was able to show that mathematically the Kitchen and Rosenfeld (KR) corner detector using nonmaximum suppression is virtually identical to the Dreschler and Nagel (DN) corner detector. A year later, Shah and Jain (1984) studied the Zuniga and Haralick (ZH) corner detector (1983) based on a bicubic polynomial model of the intensity function. They showed that this is essentially equivalent to the KR corner detector. However, the ZH corner detector operates rather differently in that it thresholds the intensity gradient and then works with the subset of edge points in the image, only at that stage applying the curvature function as a corner strength criterion. By making edge detection explicit in the operator, the ZH detector eliminates a number of false corners that would otherwise be induced by noise.

The inherent near-equivalence of these three corner detectors need not be overly surprising, since in the end the different methods would be expected to reflect the same underlying physical phenomena (Davies, 1988d). However, it is gratifying that the ultimate result of these rather mathematical formulations is interpretable by something as easy to visualize as horizontal curvature multiplied by intensity gradient.

14.8 A Median-Filter-Based Corner Detector

Paler et al. (1984) devised an entirely different strategy for detecting corners. This strategy adopts an initially surprising and rather nonmathematical approach based on the properties of the median filter. The technique involves applying a median filter to the input image and then forming another image that is the difference between the input and the filtered images. This difference image contains a set of signals that are interpreted as local measures of corner strength.

Applying such a technique seems risky since its origins suggest that, far from giving a correct indication of corners, it may instead unearth all the noise in the original image and present this as a set of "corner" signals. Fortunately, analysis shows that these worries may not be too serious. First, in the absence of noise, strong signals are not expected in areas of background. Nor are they expected near straight edges, since median filters do not shift or modify such edges significantly (see Chapter 3). However, if a window is moved gradually from a background region until its central pixel is just over a convex object corner, there is no change in the output of the median filter. Hence there is a strong difference signal indicating a corner.

Paler et al. (1984) analyzed the operator in some depth and concluded that the signal strength obtained from it is proportional to (1) the local contrast and (2) the "sharpness" of the corner. The definition of sharpness they used was that of Wang et al. (1983), meaning the angle η through which the boundary turns. Since it is assumed here that the boundary turns through a significant angle (perhaps the whole angle η) within the filter neighborhood, the difference from the second-order intensity variation approach is a major one. Indeed, it is an implicit assumption in the latter approach that first- and second-order coefficients describe the local intensity characteristics with reasonable rigor, the intensity function being inherently continuous and differentiable. Thus, the second-order methods may give unpredictable results with pointed corners where directions change within the range of a few pixels. Although there is some truth in this, it is worth looking at the similarities between the two approaches to corner detection before considering the differences.

14.8.1 *Analyzing the Operation of the Median Detector*

This subsection considers the performance of the median corner detector under conditions where the gray-scale intensity varies by only a small amount within the median filter neighborhood region. This permits the performance of the corner detector to be related to low-order derivatives of the intensity variation, so that comparisons can be made with the second-order corner detectors mentioned earlier.

To proceed, we assume a continuous analog image and a median filter operating in an idealized circular neighborhood. For simplicity, since we are attempting to relate signal strengths and differential coefficients, we ignore noise. Next, recall (Chapter 3) that for an intensity function that increases monotonically with distance in some arbitrary direction $\tilde{x}$ but that does not vary in the perpendicular direction $\tilde{y}$, the median within the circular window is equal to the value at the center of the neighborhood. Thus, the median corner detector gives zero signal if the horizontal curvature is locally zero.

If there is a small horizontal curvature κ, the situation can be modeled by envisaging a set of constant-intensity contours of roughly circular shape and approximately equal curvature within the circular window that will be taken to have radius a (Fig. 14.10). Consider the contour having the median intensity value. The center of this contour does not pass through the center of the window but is displaced to one side along the negative $\tilde{x}$-axis. Furthermore, the

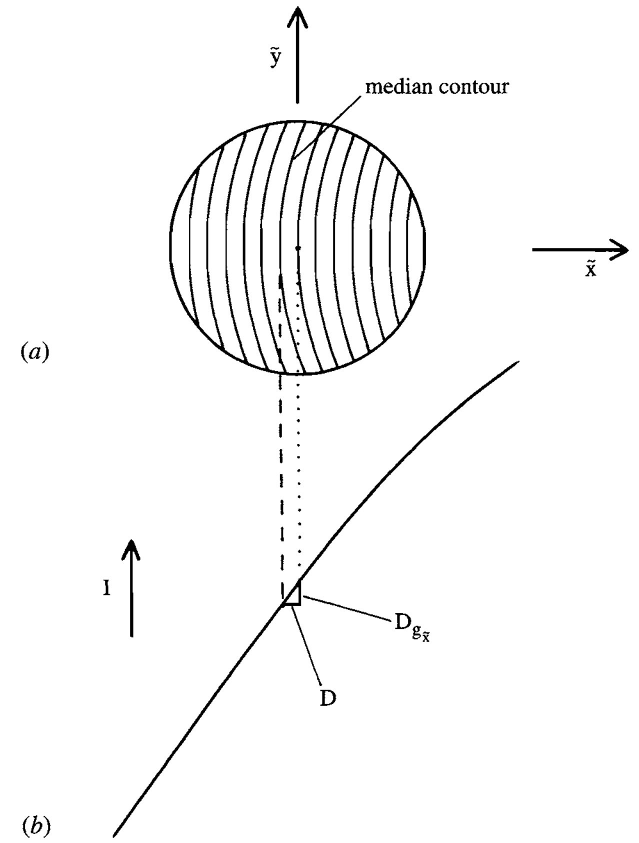


Figure 14.10 (*a*) Contours of constant intensity within a small neighborhood. Ideally, these are parallel, circular, and of approximately equal curvature (the contour of median intensity does not pass through the center of the neighborhood); (*b*) cross section of intensity variation, indicating how the displacement D of the median contour leads to an estimate of corner strength.

signal obtained from the corner detector depends on this displacement. If the displacement is D, it is easy to see that the corner signal is $Dg_{\bar{x}}$ since $g_{\bar{x}}$ allows the intensity change over the distance D to be estimated (Fig. 14.10). The remaining problem is to relate D to the horizontal curvature κ. A formula giving this relation has already been obtained in Chapter 3. The required result is:

$$D = \kappa a^2/6 \tag{14.12}$$

so the corner signal is

$$C = Dg_{\bar{x}} = \kappa g_{\bar{x}} a^2/6 \tag{14.13}$$

Note that C has the dimensions of intensity (contrast) and that the equation may be reexpressed in the form:

$$C = (g_{\bar{x}} a) \times (2a\kappa)/12 \tag{14.14}$$

so that, as in the formulation of Paler et al. (1984), corner strength is closely related to corner contrast and corner sharpness.

To summarize, the signal from the median-based corner detector is proportional to horizontal curvature and to intensity gradient. Thus, this corner detector gives an identical response to the three second-order intensity variation detectors discussed in Section 14.7, the closest practically being the KR detector. However, this comparison is valid only when second-order variations in intensity give a complete description of the situation. The situation might be significantly different where corners are so pointed that they turn through a large proportion of their total angle within the median neighborhood. In addition, the effects of noise might be expected to be rather different in the two cases, as the median filter is particularly good at suppressing impulse noise. Meanwhile, for small horizontal curvatures, there ought to be no difference in the positions at which median and second-order derivative methods locate corners, and accuracy of localization should be identical in the two cases.

14.8.2 *Practical Results*

Experimental tests with the median approach to corner detection have shown that it is a highly effective procedure (Paler et al., 1984; Davies, 1988d). Corners are detected reliably, and signal strength is indeed roughly proportional both to local image contrast and to corner sharpness (see Fig. 14.11). Noise is more apparent for 3×3 implementations, which makes it better to use 5×5 or larger

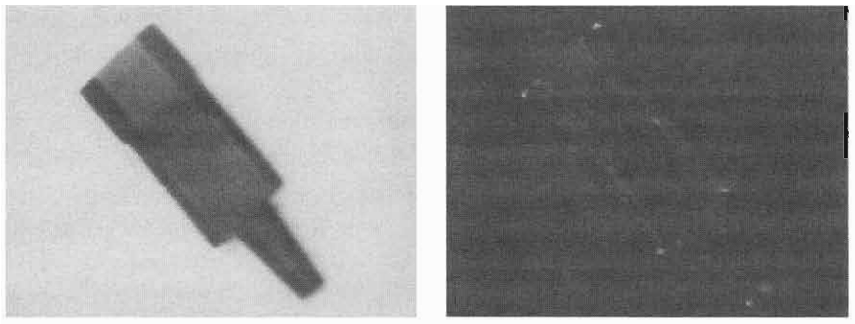


Figure 14.11 (*a*) Original off-camera 128×128 6-bit gray-scale image; (*b*) result of applying the median-based corner detector in a 5×5 neighborhood. Note that corner signal strength is roughly proportional both to corner contrast and to corner sharpness.

neighborhoods to give good corner discrimination. However, because median operations are slow in large neighborhoods and background noise is still evident even in 5×5 neighborhoods, the basic median-based approach gives poor performance by comparison with the second-order methods. However, both of these disadvantages are virtually eliminated by using a "skimming" procedure in which edge points are first located by thresholding the edge gradient, and the edge points are then examined with the median detector to locate the corner points (Davies, 1988d). With this improved method, performance is found to be generally superior to that for, say, the KR method in that corner signals are better localized and accuracy is enhanced. Indeed, the second-order methods appear to give rather fuzzy and blurred signals that contrast with the sharp signals obtained with the improved median approach (Fig. 14.12).

At this stage the reason for the more blurred corner signals obtained using the second-order operators is not clear. Basically, there is no valid rationale for applying second-order operators to pointed corners, since higher derivatives of the intensity function will become important and will at least in principle interfere with their operation. However, it is evident that the second-order methods will probably give strong corner signals when the tip of a pointed corner appears anywhere in their neighborhood, so there is likely to be a minimum blur region of radius a for any corner signal. This appears to explain the observed results adequately. The sharpness of signals obtained by the KR method may be improved by nonmaximum suppression (Kitchen and Rosenfeld, 1982; Nagel, 1983). However, this technique can also be applied to the output of median-based corner detectors. Hence, the fact remains that the median-based method gives inherently better localized signals than the second-order methods (although for low-curvature corners the signals have similar localization).

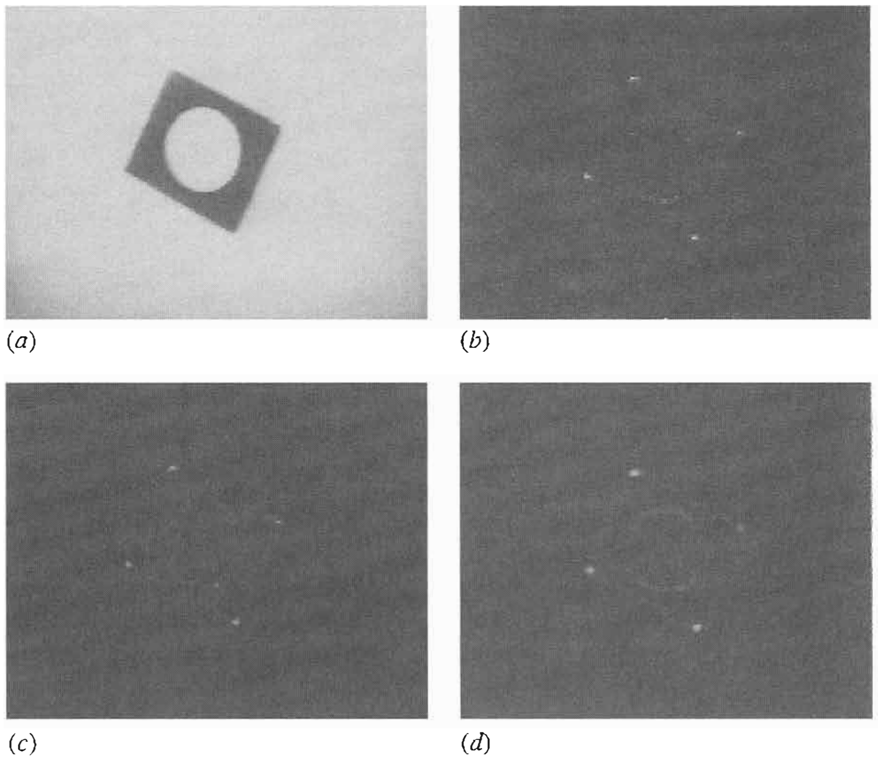


Figure 14.12 Comparison of the median and KR corner detectors: (*a*) original 128 × 128 gray-scale image; (*b*) result of applying a median detector; (*c*) result of including a suitable gradient threshold; (*d*) result of applying a KR detector. The considerable amount of background noise is saturated out in (*a*) but is evident from (*b*). To give a fair comparison between the median and KR detectors, 5 × 5 neighborhoods are employed in each case, and nonmaximum suppression operations are not applied. The same gradient threshold is used in (*c*) and (*d*).

Overall, the inherent deficiencies of the median-based corner detector can be overcome by incorporating a skimming procedure, and then the method becomes superior to the second-order approaches in giving better localization of corner signals. The underlying reason for the difference in localization properties appears to be that the median-based signal is ultimately sensitive only to the particular few pixels whose intensities fall near the median contour within the window, whereas the second-order operators use typical convolution masks that are in general sensitive to the intensity values of all the pixels within the window. Thus, the KR operator tends to give a strong signal when the tip of a pointed corner is present anywhere in the window.

14.9 The Hough Transform Approach to Corner Detection

In certain applications, corners might rarely appear in the idealized pointed form suggested earlier. Indeed, in food applications corners are commonly chipped, crumbly, or rounded, and are bounded by sides that are by no means straight lines (Fig. 14.13). Even mechanical components may suffer chipping or breakage. Hence, it is useful to have some means of detecting these types of nonideal corners. The generalized HT provides a suitable technique (Davies, 1986b, 1988a).

The method used is a variation on that described in Section 14.3 for locating objects such as squares. Instead of accumulating a line of candidate center points parellel to each side after moving laterally inward a distance equal to half the length of a side, only a small lateral displacement is required when locating corners. The lateral displacement that is chosen must be sufficient to offset any corner bluntness, so that the points that are located by the transform lines are consistently positioned relative to the idealized corner point (Fig. 14.14). In addition, the displacement must be sufficiently large that enough points on the sides of the object contribute to the corner transforms to ensure adequate sensitivity.

There are two relevant parameters for the transform: D, the lateral displacement, and T, the length of the transform line arising from each edge point. These parameters must be related to the following parameters for the object being detected: B, the width of the corner bluntness region (Fig. 14.14), and S, the length of the object side (for convenience the object is assumed to be

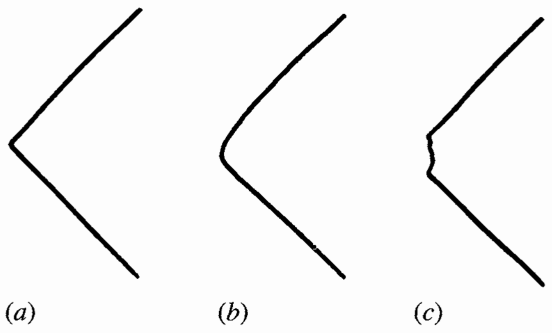


Figure 14.13 Types of corners: (*a*) pointed; (*b*) rounded; (*c*) chipped. Corners of type (*a*) are normal with metal components, those of type (*b*) are usual with biscuits and other food products, whereas those of type (*c*) are common with food products but rarer with metal parts.

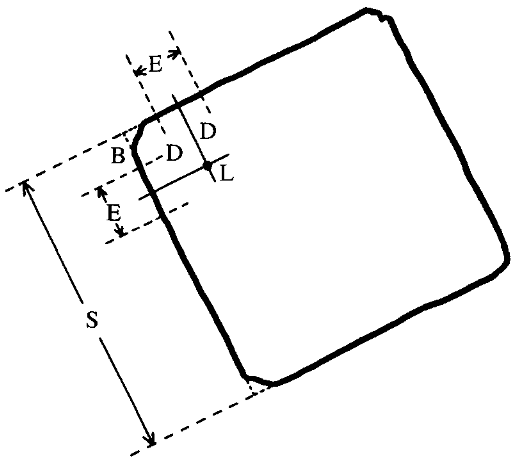


Figure 14.14 Geometry for locating blunt corners: B, region of bluntness; D, lateral displacement employed in edge pixel transform; E, distance over which position of corner is estimated; I, idealized corner point; L, localization point of a typical corner; S, length of a typical side. The degree of corner bluntness shown here is common among food products such as fishcakes.

a square). E, the portion of a side that is to contribute to a corner peak, is simply related to T:

$$T = E \tag{14.15}$$

Since we wish to make corner peaks appear as close to idealized corners as possible, we may also write:

$$D = B + E/2 \tag{14.16}$$

These definitions now allow the sensitivity of corner location to be optimized. Note first that the error in the measurement of each corner position is proportional to $1/\sqrt{E}$, since our measurement of the relevant part of each side is effectively being averaged over E independent signals. Next, note that the purpose of locating corners is ultimately to measure (1) the orientation of the object and (2) its linear dimensions. Hence, the fractional accuracy is proportional to:

$$A = (S - 2D)\sqrt{E} \tag{14.17}$$

Eliminating D now gives:

$$A = (S - 2B - E)\sqrt{E} \tag{14.18}$$

Differentiating, we find:

$$\mathrm{d}A/\mathrm{d}E = (S - 2B - 3E)/2\sqrt{E} \tag{14.19}$$

Accuracy can now be optimized by setting $\mathrm{d}A/\mathrm{d}E = 0$, so that:

$$E = S/3 - 2B/3 \tag{14.20}$$

making

$$D = S/6 + 2B/3 \tag{14.21}$$

Hence, there is a clear optimum in accuracy, and as $B \rightarrow 0$ it occurs for $D \approx S/6$, whereas for rather blunt or crumbly types of objects (typified by biscuits) $B \approx S/8$ and then $D \approx S/4$.

If optimal object detection and location had been the main aim, this would have been achieved by setting $E = S - 2B$ and detecting all the corners together at the center of the square. It is also possible that by optimizing object orientation, object detection is carried out with reduced sensitivity, since only a length $2E$ of the perimeter contributes to each signal, instead of a length $4S - 8B$.

The calculations presented here are important for a number of reasons: (1) they clarify the point that tasks such as corner location are carried out not in isolation but as parts of larger tasks that are optimizable; (2) they show that the ultimate purpose of corner detection may be either to locate objects accurately or to orient and measure them accurately; and (3) they indicate that there is an optimum region of interest E for any operator that is to locate corners. This last factor has relevance when selecting the optimum size of neighborhood for the other corner detectors described in this chapter.

Although the calculations above emphasized sensitivity and optimization, the underlying principle of the HT method is to find corners by interpolation rather than by extrapolation, as interpolation is generally a more accurate procedure. The fact that the corners are located away from the idealized corner positions is to some extent a disadvantage (although it is difficult to see how a corner can be located directly if it is missing). However, once the corner peaks have been found, for known objects the idealized corner positions may be deduced (Fig. 14.15). It is as well to note that this step is unlikely to be necessary if the real purpose of corner location is to orient the object and to measure its dimensions.

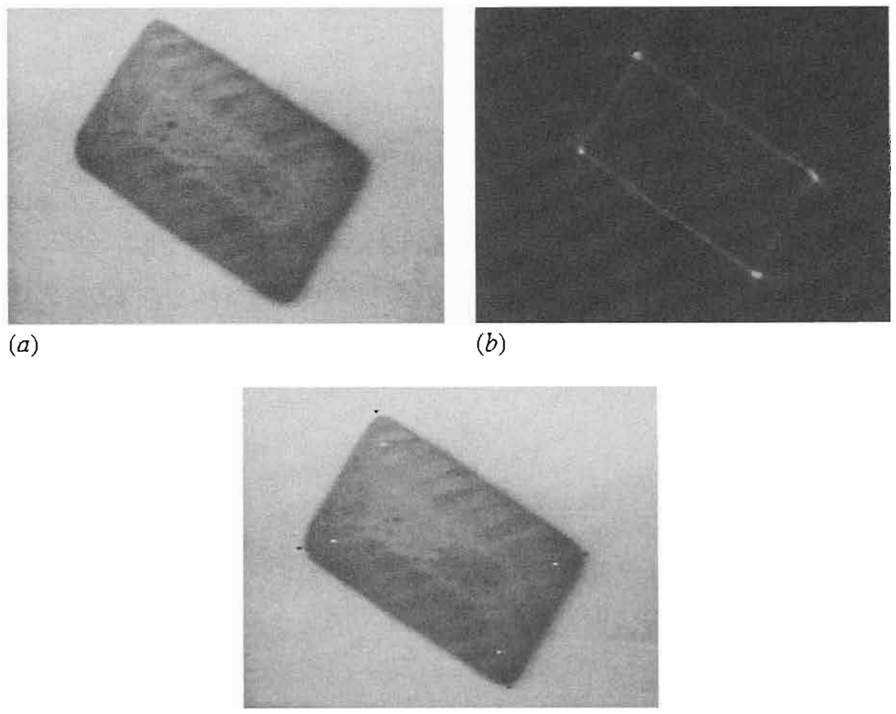

(*a*) (*b*) (*c*)

Figure 14.15 Example of the Hough transform approach: (*a*) original image of a biscuit (128×128 pixels, 64 gray levels); (*b*) transform with lateral displacement around 22% of the shorter side; (*c*) image with transform peaks located (white dots) and idealized corner positions deduced (black dots). The lateral displacement employed here is close to the optimum for this type of object.

14.10 The Plessey Corner Detector

At this point we consider whether other strategies for corner detection could be devised. The second-order derivative class of corner detector was devised on the basis that corners are ideal, smoothly varying differentiable intensity profiles, whereas the median-based detector was conceived totally independently on the basis of curved step edges whose profiles are quite likely *not* to be smoothly varying and differentiable. It is profitable to take the latter idea further and to consider corners as the intersection of two step edges.

First, imagine that a small region, which is the support region of such a corner detector, contains both a step edge aligned along the x direction and another aligned along the y direction. To detect these edges, we can use an operator that estimates $<I_x>$ and another that estimates $<I_y>$. Note that a corner detector would have to detect both types of edges simultaneously. To do so, we can use a combined estimator such as $<|I_x|><|I_y|>$ or $<I_x^2><I_y^2>$. Each of these operators leads to corner signals that can be thresholded conveniently using a single positive threshold value.

Next, note that this estimator requires the edge orientations to be known and indeed aligned along the x- and y-axes directions. To obtain more general properties, the operator needs to transform appropriately under rotation of axes. In addition, magnitudes are difficult to handle mathematically, so we start with the squared magnitude estimator and employ a symmetrical matrix of first derivatives in the transformable form:

$$\mathscr{G} = \begin{bmatrix} <I_x^2> & <I_xI_y> \\ \\ <I_xI_y> & <I_y^2> \end{bmatrix} \tag{14.22}$$

where

$$\det(\mathscr{G}) = <I_x^2><I_y^2> - <I_xI_y>^2 \tag{14.23}$$

This expression would clearly be identically zero if evaluated separately at each individual pixel. Nevertheless, our intuition is correct, and the base parameters $<I_x^2>$, $<I_y^2>$, and $<I_xI_y>$ must all be determined as *averages* over all pixels in the detector support region. If this is done, the expression need not average to zero, as the interpixel variations leading to the three base parameter values will introduce deviations. This situation is easily demonstrated, as rotation of axes to permit diagonalization of the matrix yields:

$$\tilde{\mathscr{G}} = \begin{bmatrix} <I_{\tilde{x}}^2> & 0 \\ \\ 0 & <I_{\tilde{y}}^2> \end{bmatrix} = \begin{bmatrix} \gamma_1 & 0 \\ \\ 0 & \gamma_2 \end{bmatrix} \tag{14.24}$$

with eigenvalues $<I_{\tilde{x}}^2> = \gamma_1$, $<I_{\tilde{y}}^2> = \gamma_2$. In this case, the Hessian is $\gamma_1\gamma_2$, and this is a sensible value to take for the corner strength. In particular, it is substantial only if both edges are present within the support region. The Plessey operator (Harris and Stephens, 1988; see also Noble, 1988) normalizes this measure by dividing by $\gamma_1 + \gamma_2$ and forming a square root:[3]

$$\mathscr{C} = [\gamma_1\gamma_2/(\gamma_1+\gamma_2)]^{1/2} = [< I_{\tilde{x}}^2 >< I_{\tilde{y}}^2 >/(< I_{\tilde{x}}^2 > + < I_{\tilde{y}}^2 >)]^{1/2} \tag{14.25}$$

Interestingly, this does not ruin the transformation properties of the operator because both the determinant and the trace of the relevant symmetrical matrix are invariant under orthogonal transformations.

This operator has some useful properties. In particular, it is better to call it an "interest" operator rather than a corner detector because it also responds to situations where the two constituent edges cross over, so that a double corner results (as happens in many places on a checkerboard). It is really responding to and averaging the effects of all the locations where two edges appear within the operator support region. We can easily show that it has this property, as the fact that the operator embodies $< I_{\tilde{x}}^2 >$ and $< I_{\tilde{y}}^2 >$ means that it is insensitive to the sign of the local edge gradients (Fig. 14.16*a*).

The Plessey operator also has the undesirable property[4] that it can give a bias at T-junctions, to an extent dependent on the relative intensities of the three adjacent regions (Fig. 14.16*b*). This is the effect of averaging the three base parameters over the whole support region. It is difficult to be absolutely sure where the peak response will be as the computation is quite complex. In such cases, symmetry can act as a useful constraining factor—as in the case of a double corner on a checkerboard, where the peak signal necessarily coincides with the double corner location. (This applies to all the edge crossover points in Fig. 14.16*a*.) Clearly, noise will have some effect on the peak locations in off-camera images.

14.11 Corner Orientation

This chapter has so far considered the problem of corner detection as relating merely to corner location. However, corners differ from holes in that they are not isotropic, and hence they are able to provide orientation information. Such information can be used by procedures that collate the information from various features in order to deduce the presence and positions of objects

3 As originally devised, the Plessey operator is inverted relative to this definition, and a square root is not computed. Here we include a square root so that the corner signal is proportional to image contrast, as for the other corner detectors described in this chapter.

4 See, for example, the images in Shen and Wang (2002).

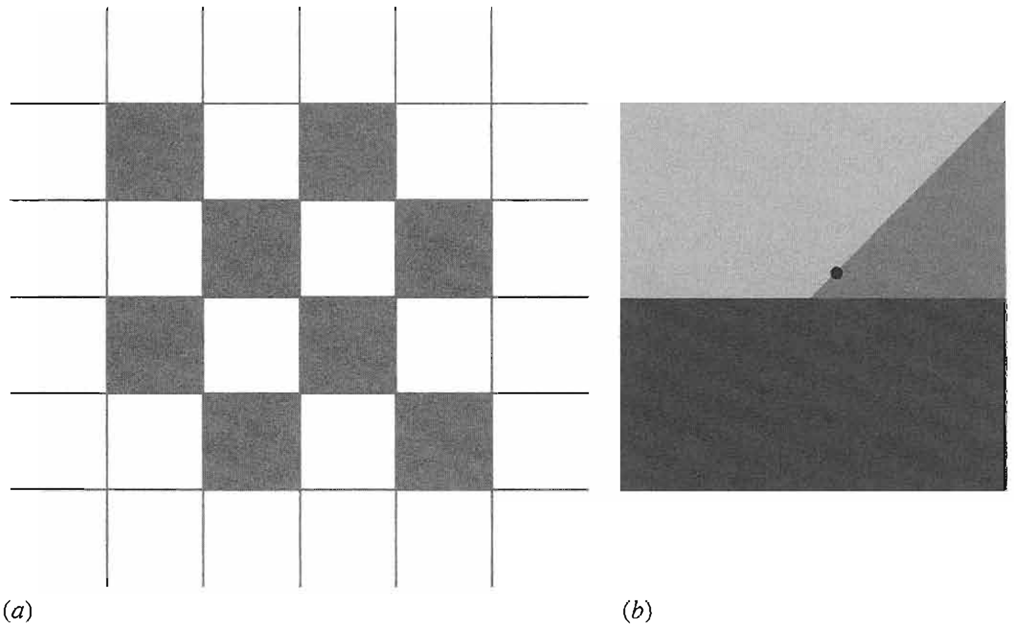

(*a*) (*b*)

Figure 14.16 Effect of the Plessey corner detector. (*a*) shows a checkerboard pattern that gives high responses at each of the edge crossover points. Because of symmetry, the location of the peaks is exactly at the crossover locations. (*b*) Example of a T-junction. The black dot shows a typical peak location. In this case there is no symmetry to dictate that the peak must occur exactly at the T-junction.

containing them. In Chapter 15 we will see that orientation information is valuable in immediately eliminating a large number of possible interpretations of an image, and hence of quickly narrowing down the search problem and saving computation.

Clearly, when corners are not particularly pointed, or are detected within rather small neighborhoods, the accuracy of orientation will be somewhat restricted. However, orientation errors will seldom be worse than 90° and will often be less than 20°. Although these accuracies are far worse than those (around 1°) for edge orientation (see Chapter 5), they provide valuable constraints on possible interpretations of an image.

Here we consider only simple means of estimating corner orientation. Basically, once a corner has been located accurately, it is a rather trivial matter to estimate the orientation of the intensity gradient at that location. The main problem that arises is that a slight error in locating the corner gives a bias in estimating its orientation toward that of one or another of the bounding sides—and for a sharp pointed corner, this can mean an error approaching 90°. The simplest solution to this problem is to find the mean orientation over a small region surrounding the estimated corner position. Then, without excessive computation, the orientation accuracy will usually be within the 20° limit suggested above.

Polygonal objects may be recognized by their sides or by their corners. This chapter has studied both approaches, with corner detection probably being more generic—especially if corners are considered as general salient features. Some corner detectors are better classed as interest point detectors, for they may not restrict their attention to true corners.

14.12 Concluding Remarks

When using the GHT for the detection of polygons, it is useful to try to minimize storage and computation. In some cases only one parameter plane is needed to detect a polygonal object: commonly, a number of parameter planes of the order of the number of sides of the polygon is required, but this still represents a significant saving compared with the numbers needed to detect general shapes. It has been demonstrated how optimal savings can be achieved by taking into account the symmetry possessed by many polygon shapes such as squares and rectangles.

The analysis has shown the possibilities for direct detection of polygons, but it has also revealed the limitations of this approach. These limitations arise specifically with more complex polygons—such as those having concavities or many sides. In such cases, it may be easier to revert to using line or corner detection schemes coupled with the abstract pattern matching approaches discussed in Chapter 15. Indeed, the latter approach offers the advantage of greater resistance to shape distortions if suitable algorithms to infer the presence of objects can be found.

Hence, the chapter also appropriately examined corner detection schemes. Apart from the obvious template matching procedure, which is strictly limited in its application, and its derivative, the lateral histogram method (see Problems 14.5 and 14.6), three main approaches have been described. The first was the second-order derivative approach, which includes the KR, DN, and ZH methods—all of which embody the same basic schema. The second was the median-based method, which turned out to be equivalent to the second-order methods—but only in situations where corners have smoothly varying intensity functions; the third was a Hough-based scheme that is particularly useful when corners are liable to be blunt, chipped, or damaged. The analysis of this last scheme revealed interesting general points about the specification and design of corner detection algorithms.

Although the median-based approach has obvious inherent disadvantages—its slowness and susceptibility to noise—these negative aspects can largely be overcome by a skimming (two-stage template matching) procedure. Then the method becomes essentially equivalent to the second-order derivative methods for low-curvature intensity functions, as stated above. However, for pointed corners, higher-order derivatives of the intensity function are bound to be important. As a result the second-order methods are based on an unjustifiable rationale and are found to blur corner signals, hence reducing the accuracy of corner localization. This problem does not arise for the median-based method, which seems to offer rather superior performance in practical situations. Clearly, however, it should not be used when image noise is excessive, as then the skimming procedure will either (with a high threshold) remove a proportion of the corners with the noise, or else (with a low threshold) not provide adequate speedup to counteract the computational load of the median filtering routine.

Overall, the Plessey corner detector described in Section 14.10 and more recently the SUSAN detector (Smith and Brady, 1997) have been among the most widely used of the available corner detectors, though whether this is the result of conclusive tests against carefully defined criteria is somewhat doubtful. For more details and comparisons, see the next section.

14.13 Bibliographical and Historical Notes

The initial sections of this chapter arose from the author's own work which was aimed at finding optimally sensitive means of locating shapes, starting with spatial matched filters. Early work in this area included that of Sklansky (1978), Ballard (1981), Brown (1983), and Davies (1987c), which compared the HT approach with spatial matched filters. These general investigations led to more detailed studies of how straight edges are optimally detected (Davies, 1987d,j), and then to polygon detection (Davies, 1986d, 1989c). Although optimal sensitivity is not the only criterion for effectiveness of object detection, nonetheless it is of great importance, since it governs both effectiveness per se (especially in the presence of "clutter" and occlusion) and the accuracy with which objects may be located.

The emphasis on spatial matched filters seemed to lead inevitably to use of the GHT, with its accompanying computational problems. The present chapter should make it plain that although these problems are formidable, systematic analysis can make a significant impact on them.

Corner detection is a subject that has been developing for well over two decades. Beaudet's (1978) work on rotationally invariant image operators set the scene for the development of parallel corner detection algorithms. This was soon followed by Dreschler and Nagel's (1981) more sophisticated second-order corner detector. The motivation for their research was mapping the motion of cars in traffic scenes, with corners providing the key to unambiguous interpretation of image sequences. One year later, Kitchen and Rosenfeld (1982) had completed their study of corner detectors based mainly on edge orientation and had developed the second-order KR method described in Section 14.7. The years 1983 and 1984 saw the development of the lateral histogram corner detector, the second-order ZH detector, and the median-based detector (Wu and Rosenfeld, 1983; Zuniga and Haralick, 1983; Paler et al., 1984). Subsequently, Davies' work on the detection of blunt corners and on analyzing and improving the median-based detector appeared (1986b, 1988a,d, 1992a). These papers form the basis of Sections 14.8 and 14.9. Meanwhile, other methods had been developed, such as the Plessey algorithm (Harris and Stephens, 1988; see also Noble, 1988), which is covered in Section 14.10. The much later algorithm of Seeger and Seeger (1994) is also of interest because it appears to work well on real (outdoor) scenes in real time without requiring the adjustment of any parameters.

The Smith and Brady (1997) SUSAN algorithm marked a further turning point, needing no assumptions on the corner geometry, for it works by making simple comparisons of local gray levels. This is one of the most often cited of all corner detection algorithms. Rosin (1999) adopted the opposite approach of modeling corners in terms of subtended angle, orientation, contrast, bluntness, and boundary curvature with excellent results. However, parametric approaches such as this tend to be computation intensive. Note also that parametric models sometimes run into problems when the assumptions made in the model do not match the data of real images. It is only much later, when many workers have tested the available operators, that such deficiencies come to light. The late 1990s also saw the arrival of a new boundary-based corner detector with good detection and localization (Tsai et al., 1999). A development of the Plessey detector called a gradient direction detector (Zheng et al., 1999)—which was optimized for good localization—also has reduced complexity.

In the 2000s, further corner detectors have been developed. The Ando (2000) detector, similar in concept to the Tsai et al. (1999) detector, has been found to have good stability, localization, and resolution. Lüdtke et al. (2002) designed a detector based on a mixture model of edge orientation. In addition to being effective in comparison with the Plessey and SUSAN operators, particularly at large opening angles, the method provides accurate angles and strengths for the corners. Olague and Hernández (2002) worked on a unit step-edge function (USEF) concept, which is able to model complex corners well. This innovation resulted in adaptable detectors that are able to detect corners with subpixel accuracy. Shen and Wang (2002) described a Hough-transform-based detector; as this works in a 1-D parameter space, it is fast enough for real-time operation. A useful feature of the Shen and Wang paper is the comparison with, and hence between, the Wang and Brady detector, the Plessey detector, and the SUSAN detector. Several example images show that it is difficult to know exactly what one is looking for in a corner detector (i.e., corner detection is an ill-posed problem), and that even the well-known detectors sometimes inexplicably fail to find corners in some very obvious places. Golightly and Jones (2003) present a practical problem in outdoor country scenery. They discuss not only the incidence of false positives and false negatives but also the probability of correct association in corner matching (e.g., during motion).

Rocket (2003) presents a performance assessment of three corner detection algorithms—the Kitchen–Rosenfeld detector, the median-based detector, and the Plessey detector. The results are complex, with the three detectors all having very different characteristics. Rocket's paper is valuable because it shows how to optimize the three methods (not least showing that a particular parameter k for the Plessey detector should be $\sim$0.04) and also because it concentrates on careful research, with no "competition" from detectors invented by its author. Finally, Tissainayagam and Suter (2004) assess the performance of corner detectors, relating this study specifically to point feature (motion) tracking applications. It would appear that whatever the potential value of other reviews, this one has particular validity within its frame of reference. Interestingly, it finds that, in image sequence analysis, the Plessey detector is more robust to noise than the SUSAN detector. A possible explanation is that it "has a built-in smoothing function as part of its formulation."

This review of the literature covers many old and recent developments on corner detection, but it is not the whole story. In many applications, it is not specific corner detectors that are needed but salient point detectors. As indicated in Chapter 13, salient point features can be small holes or corners. However, they may also be characteristic patterns of intensity. Recent literature has referred to such patterns as interest points. Often interest point detectors are essentially corner detectors, but this does not have to be the case. For example, the Plessey detector is sometimes called an interest detector—with good reason, as indicated in Section 14.10. Moravec (1977) was among the first to refer to interest points. He was followed, for example, by Schmid et al. (2000), and later by Sebe and Lew (2003) and Sebe et al. (2003), who revert to calling them salient points but with no apparent change of meaning (except that the points should also be visually interesting). Overall, it is probably safest to use the Haralick and Shapiro (1993) definition: a point being "interesting" if it is both distinctive and invariant—that is, it stands out and is invariant to geometric distortions such as might result from moderate changes in scale or viewpoint. (Note that Haralick and Shapiro also list other desirable properties—stability, uniqueness, and interpretability.) The invariance aspect is taken up by Kenney et al. (2003) who show how to remove from consideration ill-conditioned points in order to make matching more reliable.

The considerable ongoing interest in corner detection and interest point detection underlines the point, made in Section 14.5, that good point feature detectors are vital to efficient image interpretation, as will be seen in more detail in subsequent chapters.

14.14 **Problems**

1. How would a quadrant of a circle be detected optimally? How many planes would be required in parameter space?
2. Find formulas for the number of planes required in parameter space for optimal detection of (a) a parallelogram and (b) a rhombus.
3. Find how far the signal-to-noise ratio is below the optimum for an isosceles triangle of sides 1, 5, and 5 cm, which is detected in a single-plane parameter space.
4. An N-sided polygon is to be located using the Hough transform. Show that the number of planes required in parameter space can, in general, be reduced from $N+1$ to $N-1$ by a suitable choice of localization point. Show that this rule breaks down for a semicircle, where the circular part of the boundary can be regarded as being composed of an infinite number of short sides. Determine why the rule breaks down and find the minimum number of parameter planes.
5. By examining suitable binary images of corners, show that the median corner detector gives a maximal response within the corner boundary rather than halfway down the edge outside the corner. Show how the situation is modified for gray-scale images. How will this affect the value of the gradient noise-skimming threshold to be used in the improved median detector?
6. Sketch the lateral histograms (defined as in Section 13.3) for an image containing a single convex quadrilateral shape. Show that it should be possible to find the corners of the quadrilateral from the lateral histograms and hence to locate it in the image. How would the situation change for a quadrilateral with a concavity? Why should the lateral histogram method be particularly well suited to the detection of objects with blunt corners?

7. Sketch the response patterns for a [−1 2 −1] type of 1-D Laplacian operator applied to corner detection in lateral histograms for an image containing a triangle, (a) if the triangle is randomly oriented, and (b) if one or more sides are parallel to the x- or y-axis. In case (b), how is the situation resolved? In case (a), how would sensitivity depend on orientation?

8. Prove equation (14.11), starting with the following formula for curvature:

$$\kappa = (\mathrm{d}^2y/\mathrm{d}x^2)/\left[1 + (\mathrm{d}y/\mathrm{d}x)^2\right]^{3/2}$$

Hint: First express $\mathrm{d}y/\mathrm{d}x$ in terms of the components of intensity gradient, remembering that the intensity gradient vector (I_x, I_y) is oriented along the edge normal; then replace the x, y variation by I_x, I_y variation in the formula for κ.

Abstract Pattern Matching Techniques

Abstract pattern matching involves stepping back from the image itself and working at a higher level, grouping features in an abstract way to infer the presence of objects. Graph matching has long been a conventional method for performing this task, but in the right circumstances a suitable adaptation of the generalized Hough transform can actually outperform it. This chapter discusses inference procedures and considers relational descriptions of scenes and the various types of searches that can be used with image data.

Look out for:

- the match graph approach for identifying objects from their point features.
- how the need for robustness against noise, clutter, and occlusion translates into the requirement for subgraph–subgraph isomorphism.
- the maximal clique paradigm.
- how symmetry can be used to simplify the matching task.
- how the generalized Hough transform can be used for point pattern matching.
- how order calculations can be used to compare the speeds of matching algorithms.
- how relational descriptors may be used for logical analysis of scenes.
- the different types of search algorithms that may be used in scene analysis.

This chapter completes the work of Part 2 by showing how the presence of objects can be inferred from point features as an alternative to edge features. Even with point features it is found that the Hough transform may sometimes be used with advantage. However, all inference techniques need to be analyzed for computational complexity and suitable optimizations made. The need for complexity analysis carries on with even more force in subsequent work—not least the more complex algorithms used for processing 3-D images in Part 3 of this volume.

CHAPTER 15

Abstract Pattern Matching Techniques

15.1 Introduction

We have seen how objects having quite simple shapes may be located in digital images via the Hough transform. For more complex shapes this approach tends to require excessive computation. In general, this problem may be overcome by locating objects from their features. Suitable salient features include small holes, corners, straight, circular, or elliptical segments, and any readily localizable subpatterns. Earlier chapters have shown how such features may be located. However, at some stage it becomes necessary to find methods for collating the information from the various features in order to recognize and locate the objects containing them. This task is studied in the present chapter.

It is perhaps easiest to envisage the feature collation problem when the features themselves are unstructured points carrying no directional information—nor indeed any attributes other than their x, y coordinates in the image. Then the object recognition task is often called *point pattern matching*.[1] The features that are closest to unstructured points are small holes, such as the "docker" holes in many types of biscuits. Corners can also be considered as points if their other attributes—including sharpness, and orientation—are ignored. In the following section we start with point features, and then we see how the attributes of more complex types of features can be included in recognition schemes.

Overall, we should remember that it is highly efficient to use small high-contrast features for object detection, since the computation involved in searching

1 Note that this term is sometimes used not just for object recognition but for initial matching of two stereo views of the same scene.

an image decreases as the template becomes smaller. As will be clear from the preceding chapters, the main disadvantage of such an approach to object detection is the loss in sensitivity (in a signal-to-noise sense) owing to the greatly impoverished information content of the point feature image. However, even the task of identifying objects from a rather small number of point features is far from trivial and frequently involves considerable computation, as we will see in this chapter. We start by studying a graph-theoretic approach to point pattern matching, which involves the "maximal clique" concept.

15.2 A Graph-theoretic Approach to Object Location

This section considers a commonly occurring situation that involves considerable constraints—objects appearing on a horizontal worktable or conveyor at a known distance from the camera. It is also assumed (1) that objects are flat or can appear in only a restricted number of stances in three dimensions, (2) that objects are viewed from directly overhead, and (3) that perspective distortions are small. In such situations, the objects may in principle be identified and located with the aid of very few point features. Because such features are taken to have no structure of their own, it will be impossible to locate an object uniquely from a single feature, although positive identification and location would be possible using two features if these were distinguishable and if their distance apart were known. For truly indistinguishable point features, an ambiguity remains for all objects not possessing 180° rotation symmetry. Hence, at least three point features are in general required to locate and identify objects at a known range. Clearly, noise and other artifacts such as occlusions modify this conclusion. When matching a template of the points in an idealized object with the points present in a real image, we may find the following:

1. A great many feature points may be present because of multiple instances of the chosen type of object in the image.
2. Additional points may be present because of noise or clutter from irrelevant objects and structure in the background.
3. Certain points that should be present are missing because of noise or occlusion, or because of defects in the object being sought.

These problems mean that we should in general be attempting to match a subset of the points in the idealized template to various subsets of the points in the image. If the point sets are considered to constitute *graphs* with the point features as *nodes*, the task devolves into the mathematical problem of subgraph—subgraph isomorphism, that is, finding which subgraphs in the image graph are isomorphic[2] to subgraphs of the idealized template graph. Of course, there may be a large number of matches involving rather few points. These would arise from sets of features that *happen* (see, for example, item (2) above) to lie at valid distances apart in the original image. The most significant matches will involve a fair number of features and will lead to correct object identification and location. A point feature matching scheme will be most successful if it finds the most likely interpretation by searching for solutions with the greatest internal consistency—that is, with the greatest number of point matches per object.

Unfortunately, the scheme of things presented here is still too simplistic in many applications, for it is insufficiently robust against distortions. In particular, optical (e.g., perspective) distortions may arise, or the objects themselves may be distorted, or by resting partly on other objects they may not be quite in the assumed stance. Hence, distances between features may not be exactly as expected. These factors mean that some tolerance has to be accepted in the distances between pairs of features, and it is common to employ a threshold such that interfeature distances have to agree within this tolerance before matches are accepted as potentially valid. Distortions produce more strain on the point matching technique and make it all the more necessary to seek solutions with the greatest possible internal consistency. Thus, as many features as possible should be taken into account in locating and identifying objects. The maximal clique approach is intended to achieve this.

As a start, as many features as possible are identified in the original image, and these are numbered in some convenient order such as the order of appearance in a normal TV raster scan. The numbers then have to be matched against the letters corresponding to the features on the idealized object. A systematic way to do this is to construct a *match graph* (or *association graph*) in which the nodes represent feature assignments and arcs joining nodes represent pairwise compatibilities between assignments. To find the best match, it is necessary to find regions of the match graph where the cross linkages are maximized. To do so, *cliques* are sought within the match graph. A clique is a *complete subgraph*—that is, one for which all pairs of nodes are connected by arcs. However, the previous arguments indicate that if one clique is completely included within another clique, the larger clique likely represents a better match—and indeed *maximal cliques* can be taken as leading to the most reliable matches between the observed image and the object model.

Figure 15.1*a* illustrates the situation for a general triangle: for simplicity, the figure takes the observed image to contain only one triangle and assumes that lengths match exactly and that no occlusions occur. The match graph in this

2 That is, of the same basic shape and structure.

CHAPTER 26

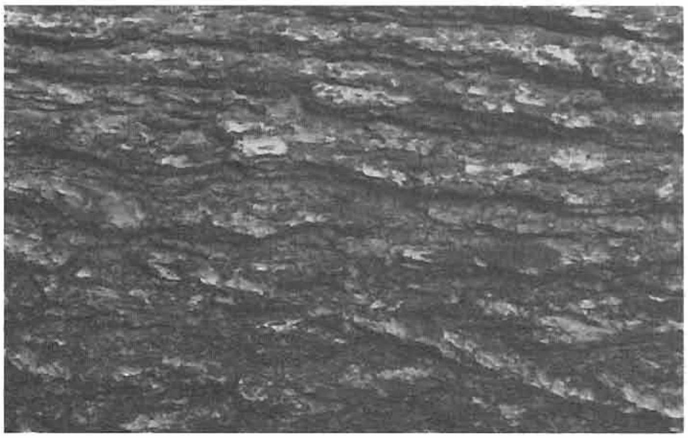
Texture

26.1 Introduction

In the foregoing chapters many aspects of image analysis and recognition have been studied. At the core of these matters has been the concept of segmentation, which involves the splitting of images into regions that have some degree of uniformity, whether in intensity, color, texture, depth, motion, or other relevant attributes. Care was taken in Chapter 4 to emphasize that such a process will be largely ad hoc, since the boundaries produced will not necessarily correspond to those of real objects. Nevertheless, it is important to make the attempt, either as a preliminary to more accurate or iterative demarcation of objects and their facets, or else as an end in itself—for example, to judge the quality of surfaces.

In this chapter we move on to the study of texture and its measurement. Texture is a difficult property to define. Indeed, in 1979 Haralick reported that no satisfactory definition of it had yet been produced. Perhaps we should not be surprised by this fact, as the concept has rather separate meanings in the contexts of vision, touch, and taste, the particular nuances being understood by different people also being highly individual and subjective. Nevertheless, we require a working definition of texture, and in vision the particular aspect we focus on is the variation in intensity[1] of a particular surface or region of an image. Even with this statement, we are being indecisive about whether we are describing the physical object being observed or the image derived from it. This reflects the fact that it is the roughness of the surface or the structure or composition of the material that gives rise to its visual properties. However, in this chapter we are interested mainly in the interpretation of images, and so we define texture as the characteristic variation in intensity of a region of an image, which should allow us to recognize and describe it and to outline its boundaries (Fig. 26.1).

1 We could at this point generalize the definition to cover variation in color, but this would complicate matters unnecessarily and would not add substantially to coverage of the subject.


(*a*)

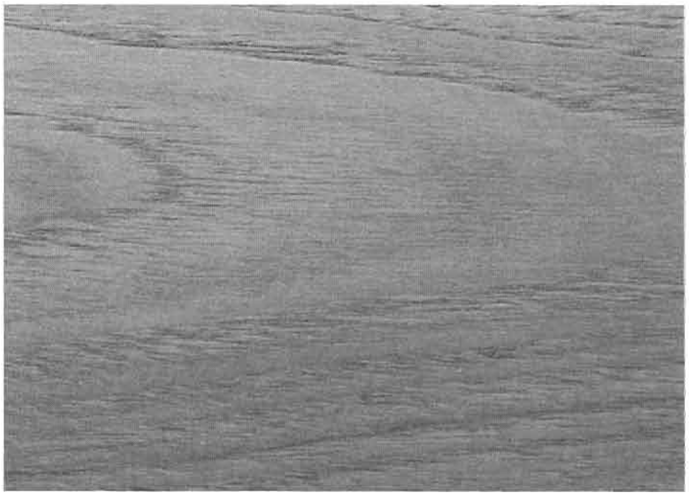
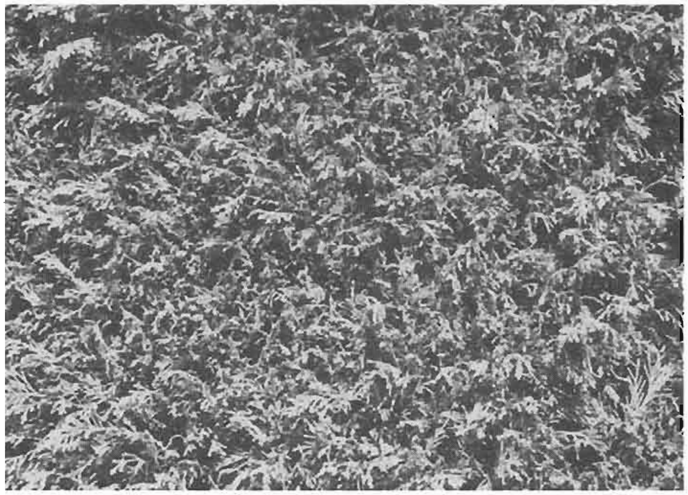

(*b*)

Figure 26.1 A variety of textures. These textures demonstrate the wide variety of familiar textures that are easily recognized from their characteristic intensity patterns.

This definition of texture implies that texture is nonexistent in a surface of uniform intensity, and it does not say anything about how the intensity might be expected to vary or how we might recognize and describe it. Intensity might vary in numerous ways, but if the variation does not have sufficient uniformity, the texture may not be characterized sufficiently closely to permit recognition or segmentation.

We next consider ways in which intensity might vary. It can vary rapidly or slowly, markedly or with low contrast, with a high or low degree of directionality,


(c)

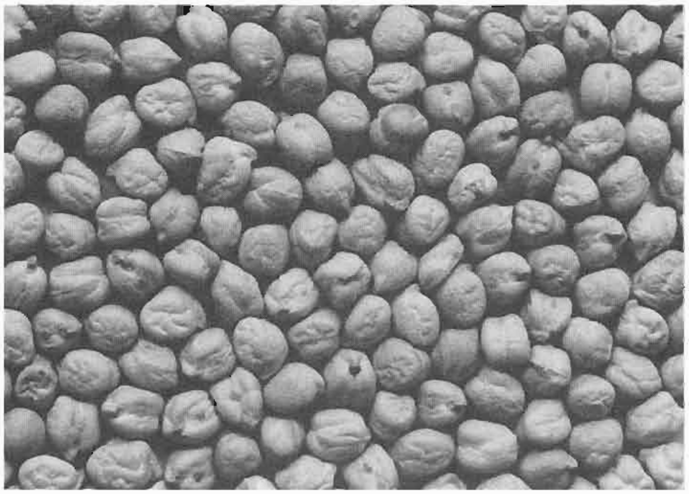
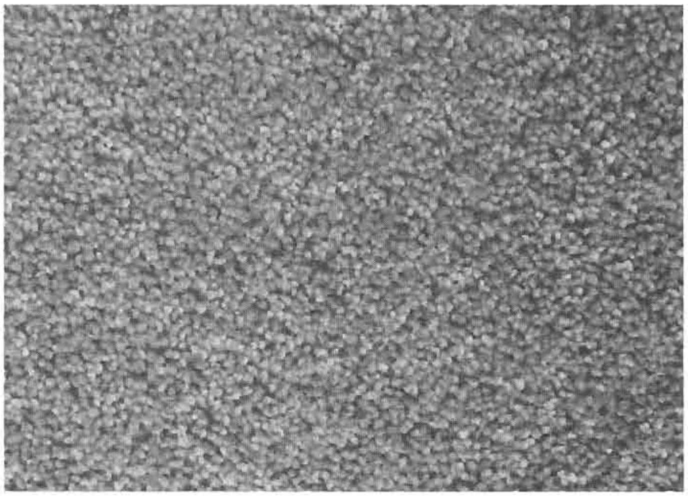

(d)

Figure 26.1 Continued.

and with greater or lesser degrees of regularity. This last characteristic is often taken as key: either the textural pattern is regular as for a piece of cloth, or it is random as for a sandy beach or a pile of grass cuttings. However, this ignores the fact that a regular textural pattern is often not wholly regular (again, as for a piece of cloth), or not wholly random (as for a mound of potatoes of similar size). Thus, the degrees of randomness and of regularity will have to be measured and compared when characterizing a texture.


(*e*)

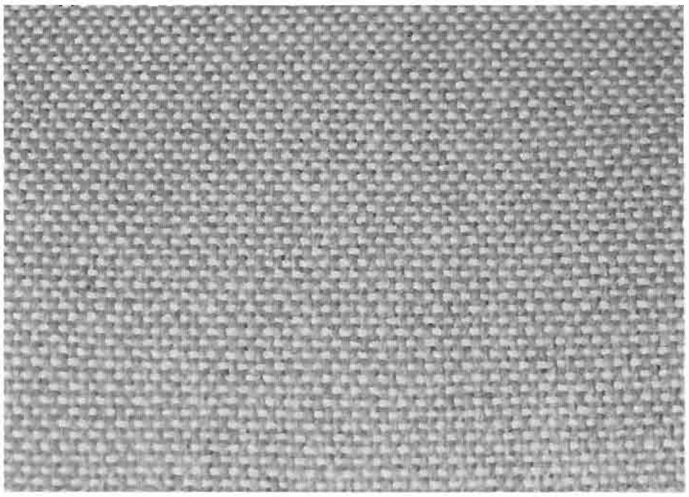
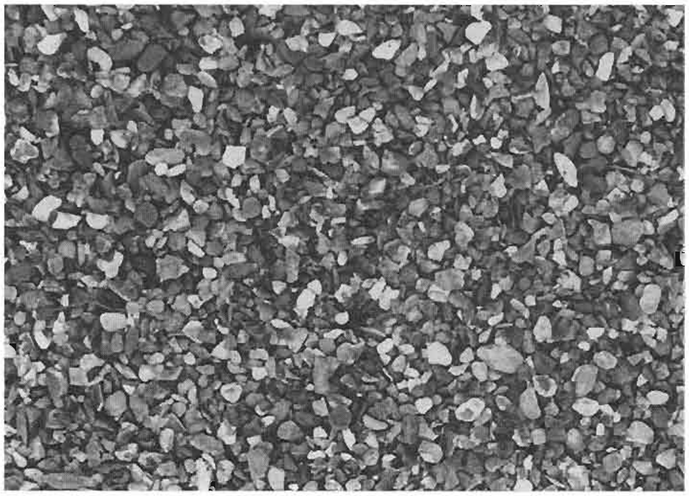

(*f*)

Figure 26.1 Continued.

There are more profound things to say about the textures. Often they are derived from tiny objects or components that are themselves similar, but that are placed together in ways ranging from purely random to purely regular—be they bricks in a wall, grains of sand, blades of grass, strands of material, stripes on a shirt, wickerwork on a basket, or a host of other items. In texture analysis it is useful to have a name for the similar textural elements


(g)

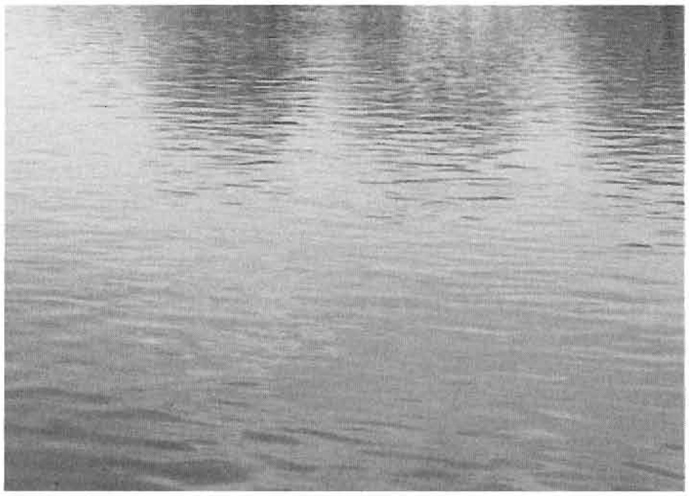

(h)

Figure 26.1 Continued.

that are replicated over a region of the image: such textural elements are called *texels*. These considerations lead us to characterize textures in the following ways:

1. The texels will have various sizes and degrees of uniformity.
2. The texels will be oriented in various directions.
3. The texels will be spaced at varying distances in different directions.

4. The contrast will have various magnitudes and variations.
5. Various amounts of background may be visible between texels.
6. The variations composing the texture may each have varying degrees of regularity vis-à-vis randomness.

It is quite clear from this discussion that a texture is a complicated entity to measure. The reason is primarily that many parameters are likely to be required to characterize it. In addition, when so many parameters are involved, it is difficult to disentangle the available data and measure the individual values or decide the ones that are most relevant for recognition. And, of course, the statistical nature of many of the parameters is by no means helpful. However, we have so far only attempted to show how complex the situation can be. In the following paragraphs, we attempt to show that quite simple measures can be used to recognize and segment textures in practical situations.

Before proceeding, it is useful to recall that in the analysis of shape a dichotomy exists between available analysis methods. We could, for example, use a set of measures such as circularity, aspect ratio, and so on which would permit a description of the shape, but which would not allow it to be reconstructed. Or we could use descriptors such as skeletons with distance function values, or moments, which would permit full and accurate reconstruction—though the set of descriptors might have been curtailed so that only limited but predictable accuracy was available. In principle, such a reconstruction criterion should be possible with texture. However, in practice there are two levels of reconstruction. In the first, we could reproduce a pattern that, to human eyes, would be indistinguishable from the off-camera texture until one compared the two on a pixel-by-pixel basis. In the second, we could reproduce a textured pattern exactly. The point is that normally textures are partially statistical in nature, so it will be difficult to obtain a pixel-by-pixel match in intensities. Neither, in general, will it be worth aiming to do so. Thus, texture analysis generally only aims at obtaining accurate statistical descriptions of textures, from which *apparently* identical textures can be reproduced if desired.

At this point it ought to be stated that many workers have contributed to, and used, a wide range of approaches for texture analysis over a period of well over 40 years. The sheer weight of the available material and the statistical nature of it can be daunting for many. It is therefore recommended that those interested in obtaining a quick working view of the subject start by reading Sections 26.2 and 26.4, looking over Sections 26.5 and 26.6, and then proceeding to the end of the chapter. In this way they will bypass much of the literature review material that is part and parcel of a full study of the subject. (Section 26.4 is particularly relevant to practitioners, as it describes the Laws' texture energy approach which is intuitive, straightforward to apply in both software and hardware, and highly effective in many application areas.)