CHAPTER

The generalized Hough transform

11

In this chapter, we shall see how the Hough transform (HT) can be used to locate general shapes, and that it is broadly able to retain its robustness properties. On a slightly different tack, 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 standard approach for achieving this, but in some circumstances, we shall see that the generalized Hough transform (GHT) is able to outperform it. This chapter discusses these inference procedures and goes on to consider the various types of search that can be used with image data.

Look out for:

- The GHT technique
- its relation to spatial matched filtering
- how sensitivity is optimized by gradient rather than uniform weighting
- how the GHT may be used for ellipse detection
- how the computational loads of the various HT techniques can be estimated
- the match graph approach for identifying objects from their point features
- why subgraph−subgraph isomorphism leads to maximum robustness
- how symmetry can be used to simplify the matching task
- situations where the GHT can outperform the maximal clique paradigm.

This chapter describes the GHT and extends our view of the HT as a generic computer vision technique. It also makes order calculations of computational load for three HT-based methods of ellipse detection. It goes on to show how the presence of objects can be inferred from sets of point features and that in some situations there is considerable advantage in using the GHT for this purpose.

11.1 INTRODUCTION

In the previous few chapters, it has been seen that the HT is of great importance for the detection of features such as lines, circles and ellipses, and for finding relevant image parameters. This makes it worthwhile to see the extent to which the

Computer Vision. DOI: http://dx.doi.org/10.1016/B978-0-12-809284-2.00011-3

method can be generalized so that it can detect arbitrary shapes. The work of Merlin and Farber (1975) and Ballard (1981) was crucial historically and led to the development of the GHT. The GHT is studied in this chapter, showing first how it is implemented and then examining how it is optimized and adapted to particular types of image data. This requires us to go back to first principles, taking spatial matched filtering as a starting point. Having developed the relevant theory, the GHT is applied to the important case of ellipse detection, showing in particular how computational load may be minimized. The computational problems of the GHT and HT are then examined more generally.

For situations where objects have more complex shapes, it is common to locate them from salient features such as small holes, corners, straight, circular or elliptical segments, and indeed 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. Graph-matching methods have often been employed to achieve this, and this approach will be examined later in the chapter. However, certain graph-matching methods such as the paradigm maximal clique approach have distinct computational problems as they are known to be NP-complete. Interestingly, in some cases, it is possible to use the GHT to carry out the point pattern matching task (of identifying objects from their point features) and thus find objects efficiently in polynomial time: this procedure is described in Section 11.10. Attention is then drawn to the use of features that have additional attributes—such as the orientation and sharpness of corners—and how this can help to cut down computation and potential ambiguities in interpretation.

11.2 THE GENERALIZED HOUGH TRANSFORM

This section shows how the standard Hough technique is generalized so that it can detect arbitrary shapes. In principle, it is trivial to achieve this. First, we need to select a localization point L within a template of the idealized shape. Then, we need to arrange so that, instead of moving from an edge point a *fixed* distance R directly along the local edge normal to arrive at the center, as for circles, we move an appropriate *variable* distance R in a variable direction φ so as to arrive at L: R and φ are now functions of the local edge normal direction θ (Fig. 11.1). Under these circumstances, votes will peak at the preselected object localization point L. The functions $R(\theta)$ and $\phi(\theta)$ can be stored analytically in the computer algorithm, or for completely arbitrary shapes, they may be stored as lookup tables. In either case, the scheme is beautifully simple in principle but two complications arise in practice. The first arises because some shapes have features such as concavities and holes, so that several values of R and φ are required for certain values of θ (Fig. 11.2). The second arises because we are going from an

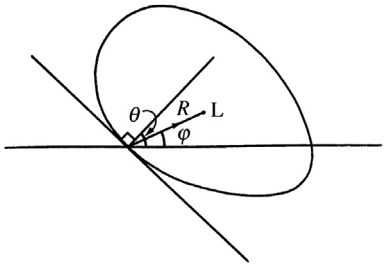


FIGURE 11.1

Computation of the generalized Hough transform.

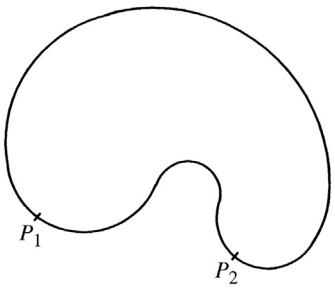


FIGURE 11.2

A shape exhibiting a concavity: certain values of θ correspond to several points on the boundary and hence require several values of R and φ—as for points P_1 and P_2.

isotropic shape (a circle) to an anisotropic shape which may be in a completely arbitrary orientation.

To cope with the first of these complications, the lookup table (usually called the "*R*-table") must contain a list of the positions **r**, relative to L, of all points on the boundary of the object for each possible value of edge orientation θ (or a similar effect must be achieved analytically): then, on encountering an edge fragment in the image whose orientation is θ, estimates of the position of L may be obtained by moving a distance (or distances) $\mathbf{R} = -\mathbf{r}$ from the given edge fragment. Clearly, if the *R*-table has multivalued entries (i.e., several values of **r** for certain values of θ), only one of these entries (for given θ) can give a correct estimate of the position of L. However, at least the method is guaranteed to give optimum sensitivity, as all relevant edge fragments contribute to the peak at L in parameter space. This property of optimal sensitivity reflects the fact that the GHT is a form of spatial matched filter: this property is analyzed in more detail below.

The second complication arises because any shape other than a circle is anisotropic. As in most applications (including industrial applications such as automated assembly), object orientations are initially unknown, the algorithm has to obtain its own information on object orientation. This means adding an extra

dimension in parameter space (Ballard, 1981). Then, each edge point contributes a vote in each plane in parameter space at a position given by that expected for an object of given shape and given orientation. Finally, the whole of parameter space is searched for peaks, the highest points indicating both the locations of objects and their orientations. Clearly, if object size is also a parameter, the problem becomes far worse, and this complication is ignored here (although the method of Section 10.7 is clearly relevant).

The changes made in proceeding to the GHT leave it just as robust as the HT circle detector described previously. This gives an incentive to improve the GHT so as to limit the computational problems in practical situations. In particular, the size of the parameter space must be cut down drastically both to save storage and to curtail the associated search task. Considerable ingenuity has been devoted to devising alternative schemes and adaptations to achieve this. Important special cases are those of ellipse detection and polygon detection, and in each of these, definite advances have been made: ellipse detection is covered in Chapter 10, Line, Circle, and Ellipse Detection for polygon detection, see Davies (1989a). Here, we proceed with some more basic studies of the GHT.

11.3 THE RELEVANCE OF SPATIAL MATCHED FILTERING

Many years ago, it was shown that the HT is equivalent to template matching (Stockman and Agrawala, 1977) and also to spatial matched filtering (Sklansky, 1978). Matched filtering dates from World War II when radar was being developed, and it was shown to be the ideal method for detecting signals: in particular, a filter that is "matched" to a given signal detects it with optimum signal-to-noise ratio (SNR) under white noise conditions (North, 1943; Turin, 1960). White noise is defined as noise that has equal power at all frequencies: in image science, white noise is understood to mean equal power at all *spatial* frequencies. The significance of this is that noise at different pixels is completely uncorrelated but is subject to the same gray-scale probability distribution—i.e., it has potentially the same range of amplitudes at all pixels.

Mathematically, using a matched filter is identical to correlation with a signal (or "template") of the same temporal or spatial profile as the one to be detected (Rosie, 1966). Unfortunately, when applying correlation in image analysis, changes in background illumination cause large changes in signal from one image to another and from one part of an image to another. These problems have been tackled in two ways, which are as follows:

1. Adjusting templates so that they have a mean value of zero, to suppress the effects of varying levels of illumination;
2. Breaking up templates into a number of smaller templates each having zero mean: then as the sizes of the subtemplates approach zero, the effects of varying levels of illumination will tend to zero.

The first of these is widely used in image analysis and has been seen as the obvious one to apply when detecting edges, line segments, corners, or other small features. Indeed, when objects are detected by small features such as these, the second method is also tacitly being used. However, there is a problem in that if objects are detected from a set of small templates, they are not actually being detected in their entirety, so ways of inferring the presence of whole objects are required. And if this is not done sufficiently rigorously, the objects might be missed, or false alarms might be detected. Thus, deviation from the matched filter paradigm brings dangers with it.

Meanwhile, we are left with the idea that the GHT is a form of matched filter and incorporates both of the above ways of coping with uncontrolled variations in background illumination. It can also be said that these ways constitute a rather crude method for applying a noise-whitening filter, thereby bringing the matched filter closer to its ideal form.

Interestingly, employing zero-mean templates results in the absolute signal level being reduced to zero and only local relative signal levels being retained, so the GHT suppresses the signal from the bulk of the object, retaining only that near its boundary. As a result, the GHT is highly sensitive to object position but is not optimized for object detection. Overall, the GHT can therefore be viewed as a type of perimeter template around the outside of an object (Fig. 11.3)—though any internal high-contrast edges should also be included in the analysis.

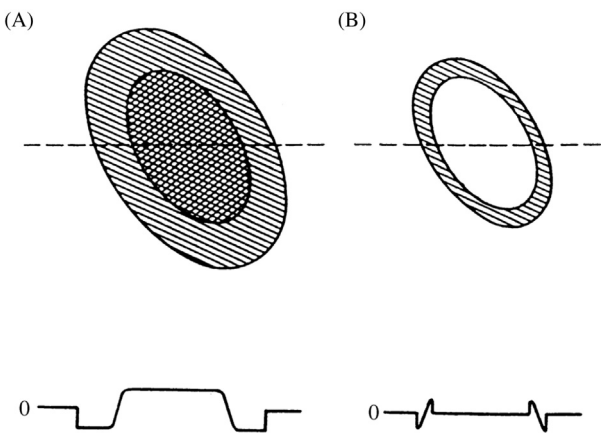


FIGURE 11.3

The idea of a perimeter template: both the original spatial matched filter template (A) and the corresponding "perimeter template" (B) have a zero mean (see text). The lower illustrations show the cross sections along the dotted lines.

11.4 GRADIENT WEIGHTING VERSUS UNIFORM WEIGHTING

Another long-standing problem regarding the GHT is how best to weight votes in parameter space in relation to the respective edge gradient magnitudes. To find an answer to this problem, we appeal to the spatial matched filter scenario for the ideal solution and then determine the corresponding solution for the GHT. First, note that the responses to the subtemplates (or to the perimeter template) are proportional to edge gradient magnitude. Next, note that with a spatial matched filter, signals are detected optimally by templates of the same shape. Each contribution to the spatial matched filter response therefore has to be proportional to the local magnitude of the signal and to that of the template. In view of the correspondence between (1) using a spatial matched filter to locate objects by looking for peaks in convolution space and (2) using a GHT to locate objects by looking for peaks in parameter space, we should use weights proportional to the gradients of the edge points *and* proportional to the a priori edge gradients.

There are two ways in which the choice of weighting is important. First, the use of uniform weighting implies that all edge pixels whose gradient magnitudes are above threshold will effectively have them reduced to the threshold value, so the signal will be curtailed: this can mean that the SNR of high-contrast objects will be reduced significantly. Second, the widths of edges of high-contrast objects will be broadened in a crude way by uniform weighting (see Fig. 11.4) but under gradient weighting this broadening will be controlled, giving a roughly Gaussian

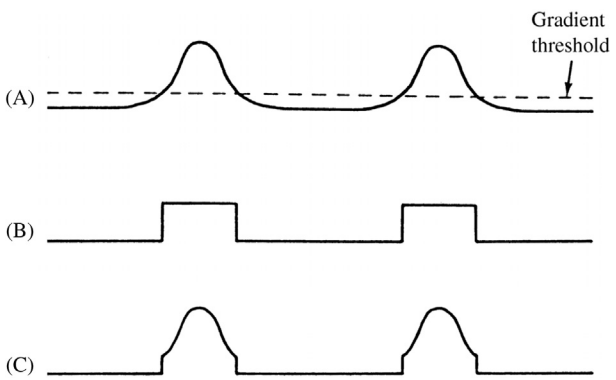


FIGURE 11.4

Effective gradient magnitude as a function of position within a section across an object of moderate contrast, thresholded at a fairly low level: (A) gradient magnitude for original image data and gradient thresholding level; (B) uniform weighting: the effective widths of edges are broadened rather crudely, adding significantly to the difficulty of locating the peak in parameter space; (C) gradient weighting: the position of the peak in parameter space can be estimated in a manner that is basically limited by the shape of the gradient profile for the original image data.

edge profile: thus, the peak in parameter space will be narrower and more rounded, and the object reference point L can be located more easily and with greater accuracy. This effect is visible in Fig. 11.5, which also shows the relatively increased noise level that results from uniform weighting.

Note also that low gradient magnitudes correspond to edges of poorly known location, whereas high values correspond to sharply defined edges: thus, the accuracy of the information relevant to object location is proportional to the magnitude of the gradient at each of the edge pixels, and appropriate weighting should therefore be used.

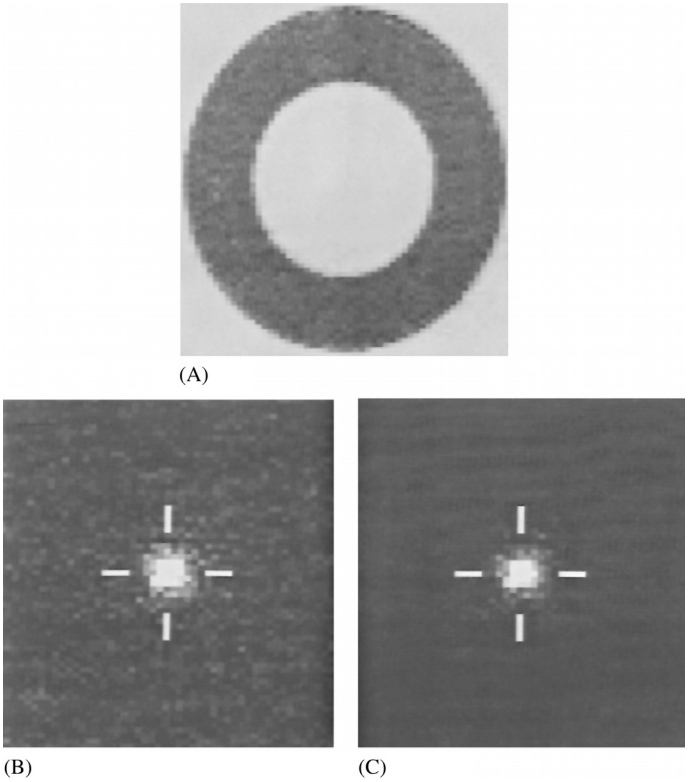


FIGURE 11.5

Results of applying the two types of weighting to a real image: (A) original image; (B) results in parameter space for uniform weighting; (C) results for gradient weighting. The peaks (which arise from the outer edges of the washer) are normalized to the same levels in the two cases: the increased level of noise in (B) is readily apparent. In this example, the gradient threshold is set at a low level (around 10% of the maximum level) so that low-contrast objects can also be detected.

11.4.1 CALCULATION OF SENSITIVITY AND COMPUTATIONAL LOAD

The aim of this subsection is to underline the above ideas by working out formulae for sensitivity and computational load. It is assumed that p objects of size around $n \times n$ are being sought in an image of size $N \times N$.

Correlation requires N^2n^2 operations to compute the convolutions for all possible positions of the object in the image. Using the perimeter template, the number of basic operations is reduced to $\sim N^2n$, corresponding to the reduced number of pixels in the template. The GHT requires $\sim N^2$ operations to locate the edge pixels, plus a further $\sim pn$ operations to accumulate the points in parameter space.

The situation for sensitivity is rather different. With correlation, the results for n^2 pixels are summed, giving a signal proportional to n^2, although the noise (assumed to be independent at every pixel) is proportional to n: this is because of the well-known result that the noise powers of various independent noise components are additive (Rosie, 1966). Overall, this results in the SNR being proportional to n. The perimeter template possesses only $\sim n$ pixels, and here, the overall result is that the SNR is proportional to $\sqrt{n}$. The situation for the GHT is inherently identical to that for the perimeter template method, so long as plots in parameter space are weighted in proportion to edge gradient g multiplied by a priori edge gradient G. It is now necessary to compute the constant of proportionality α. Take s as the average signal, equal to the intensity (assumed to be roughly uniform) over the body of the object, and S as the magnitude of a full matched filter template. In the same units, g (and G) is the magnitude of the signal within the perimeter template. Then, $\alpha = 1/sS$. This means that the perimeter template method and the GHT method lose sensitivity in two ways—first because they look at less of the available signal, and second because they look where the signal is low. For a high value of gradient magnitude, which occurs for a step edge (where most of the variation in intensity takes place within the range of 1 pixel), the values of g and G saturate out, so that they are nearly equal to s and S (see Fig. 11.6). Under these conditions, the perimeter template method and the GHT have sensitivities that depend only on the value of n.

Table 11.1 summarizes the situation discussed above. The oft-quoted statement that the computational load of the GHT is proportional to the number of

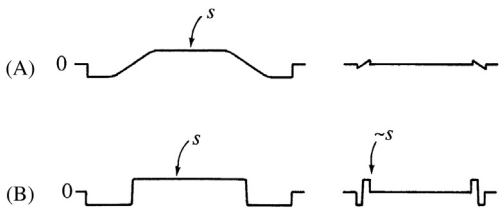


FIGURE 11.6

Effect of edge gradient on perimeter template signal: (A) low edge gradient: signal is proportional to gradient; (B) high edge gradient: signal saturates at value of s.

Table 11.1 Formulae for Computational Load and Sensitivity[a]

	Template Matching	Perimeter Template Matching	Generalized Hough Transform
Number of operations	$O(N^2n^2)$	$O(N^2n)$	$O(N^2) + O(pn)$
Sensitivity	$O(n)$	$O(\sqrt{n}gG/sS)$	$O(\sqrt{n}gG/sS)$
Maximum sensitivity[b]	$O(n)$	$O(\sqrt{n})$	$O(\sqrt{n})$

[a] *This table gives formulae for computational load and sensitivity when* p *objects of size* n × n *are sought in an image of size* N × N. *The intensity of the image within the whole object template is taken as* s *and the value for the ideal template is taken as* S*: corresponding values for intensity gradient within the perimeter template are* g *and* G.
[b] *Maximum sensitivity refers to the case of a step edge, for which* $g \approx s$ and $G \approx S$ *(see Fig. 11.6).*

perimeter pixels, rather than to the much greater number of pixels within the body of an object, is only an approximation. In addition, this saving is not obtained without cost: in particular, the sensitivity (SNR) is reduced (at best) as the square root of object area/perimeter (note that area and perimeter are measured in the same units, so it is valid to find their ratio).

Finally, the absolute sensitivity for the GHT varies as gG. As contrast changes so that $g \rightarrow g'$, we see that $gG \rightarrow g'G$: i.e., sensitivity changes by a factor g'/g. Hence, theory predicts that sensitivity is proportional to contrast. Although this result might have been anticipated, we now see that it is valid only under conditions of gradient weighting.

11.4.2 SUMMARY

The above sections examined the GHT and found that the following factors are involved in optimizing it:

1. Each point in parameter space should be weighted in proportion to the intensity gradient at the edge pixel giving rise to it, *and* in proportion to the a priori gradient, if sensitivity is to be optimized.
2. The computational load of the GHT can be minimized by ignoring pixels having low magnitudes of intensity gradient. If the threshold of gradient magnitude is set too high, fewer objects are likely to be detected; if it is set too low, computational savings will be diminished. Suitable means are required for setting the threshold, but little reduction in computation will be possible if the highest sensitivity in a low-contrast image is to be retained.
3. The GHT is inherently optimized for object location rather than object detection. This means that it may miss low-contrast objects which are detectable by other methods that take the whole area of an object into account. However, this consideration is often unimportant in applications where SNR is less of a problem than finding objects quickly in an uncluttered environment.

Overall, it is clear that the GHT is a spatial matched filter only in a particular sense, and as a result may not always achieve the highest possible sensitivity. The main advantage of the technique is that it is highly efficient, overall computational load in principle being proportional to the relatively few pixels on the perimeters of objects rather than to the much greater numbers of pixels within them. In addition, by concentrating on the boundaries of objects, the GHT retains its power to locate objects accurately. It is thus important to distinguish clearly between sensitivity in *detecting* objects and sensitivity in *locating* them.

11.5 USE OF THE GHT FOR ELLIPSE DETECTION

It has already been seen that when the GHT is used to detect anisotropic objects, there is an intrinsic need to employ a large number of planes in parameter space. However, it is shown below that by accumulating the votes for all possible orientations in a *single* plane in parameter space, significant savings in computation can sometimes be made. Basically, the idea is largely to reduce the considerable storage requirements of the GHT by using only one instead of (typically) 360 planes in parameter space, whereas at the same time significantly reducing the computation involved in the final search for peaks. Such a scheme could have concomitant disadvantages such as the production of spurious peaks, and this aspect will have to be examined carefully.

To achieve these aims, it is necessary to analyze the shape of the point spread function (PSF) to be accumulated for each edge pixel. To demonstrate this, we take the case of ellipses of unknown orientation. We start by taking a general edge fragment at a position defined by ellipse parameter ψ and deducing the bearing of the center of the ellipse relative to the local edge normal (Fig. 11.7). Working first in an ellipse-based axes system, for an ellipse with semimajor and semiminor axes a and b, respectively, it is clear that:

$$x = a \cos \psi \tag{11.1}$$

$$y = b \sin \psi \tag{11.2}$$

Hence,

$$\mathrm{d}x/\mathrm{d}\psi = -a \sin \psi \tag{11.3}$$

$$\mathrm{d}y/\mathrm{d}\psi = b \cos \psi \tag{11.4}$$

giving

$$\mathrm{d}y/\mathrm{d}x = -(b/a) \cot \psi \tag{11.5}$$

Hence, the orientation of the edge normal is given by:

$$\tan \theta = (a/b) \tan \psi \tag{11.6}$$

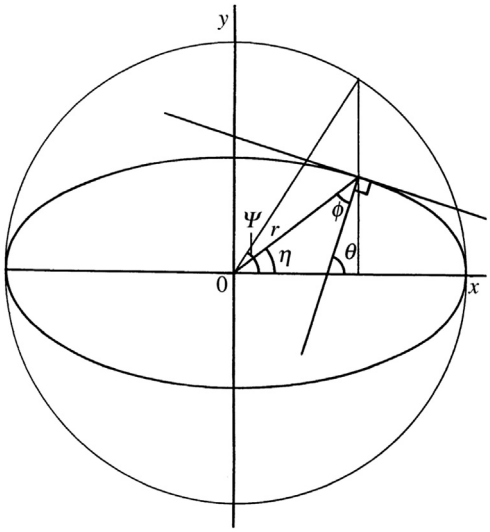


FIGURE 11.7

Geometry of an ellipse and its edge normal.

At this point, we wish to deduce the bearing of the center of the ellipse relative to the local edge normal. From Fig. 11.7:

$$\phi = \theta - \eta \tag{11.7}$$

where

$$\tan \eta = y/x = (b/a) \tan \psi \tag{11.8}$$

and

$$\begin{aligned} \tan \phi &= \tan (\theta - \eta) \\ &= \frac{\tan \theta - \tan \eta}{1 + \tan \theta \tan \eta} \end{aligned} \tag{11.9}$$

Substituting for $\tan\theta$ and $\tan\eta$, and then rearranging, gives:

$$\tan \phi = \frac{(a^2 - b^2)}{2ab} \sin 2\psi \tag{11.10}$$

In addition:

$$r^2 = a^2 \cos^2 \psi + b^2 \sin^2 \psi \tag{11.11}$$

To obtain the PSF for an ellipse of unknown orientation, we now simplify matters by taking the current edge fragment to be at the origin and orientated with its normal along the u-axis (Fig. 11.8). The PSF is then the locus of all

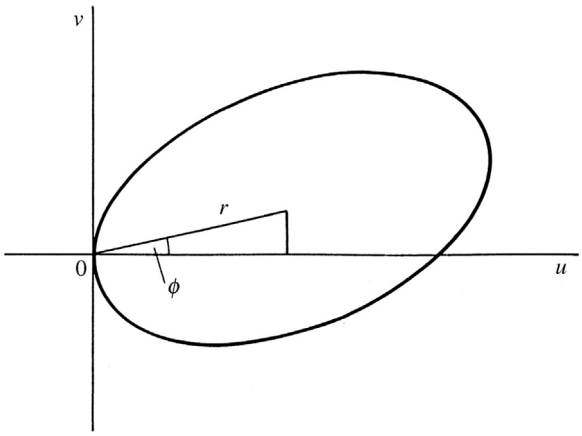


FIGURE 11.8

Geometry for finding the PSF for ellipse detection by forming the locus of the centers of ellipses touching a given edge fragment.

possible positions of the center of the ellipse. To find its form, it is merely required to eliminate ψ between Eqs. (11.10) and (11.11). This is facilitated by reexpressing r^2 in double angles (the significance of double angles lies in the 180° rotation symmetry of an ellipse):

$$r^2 = \frac{a^2 + b^2}{2} + \frac{a^2 - b^2}{2} \cos 2\psi \tag{11.12}$$

After some manipulations, the locus is obtained as:

$$r^4 - r^2(a^2 + b^2) + a^2 b^2 \sec^2 \phi = 0 \tag{11.13}$$

which can, in the edge-based coordinate system, also be expressed in the form:

$$v^2 = (a^2 + b^2) - u^2 - a^2 b^2 / u^2 \tag{11.14}$$

In fact, this is a complex and variable shape, as shown in Fig. 11.9, though for ellipses of low eccentricity, the PSF also approximates to an ellipse. However, it is normally better to implement it accurately using a specially constructed lookup table.

11.5.1 PRACTICAL DETAILS

Having constructed a lookup table for ellipse detection, the detection algorithm has to scale it, position it, and rotate it so that points can be accumulated in parameter space. Fig. 11.10 shows the result of applying the above scheme to an image of some O-rings lying on a slope, whereas Fig. 11.11 shows the result obtained for an elliptical object; the respective PSFs contain 50 and 100 votes.

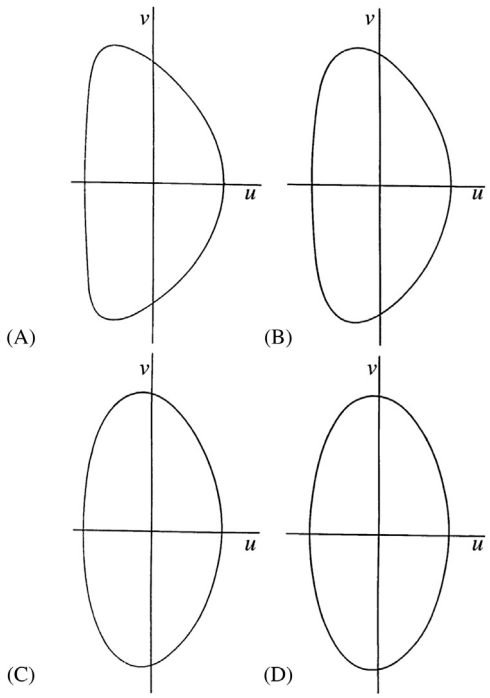


FIGURE 11.9

Typical PSF shapes for detection of ellipses with various eccentricities: (A) ellipse with $a/b = 21.0$; (B) ellipse with $a/b = 5.0$; (C) ellipse with $a/b = 2.0$; (D) ellipse with $a/b = 1.4$. Notice how the PSF shape approaches a small ellipse of aspect ratio 2.00 as eccentricity tends to zero.

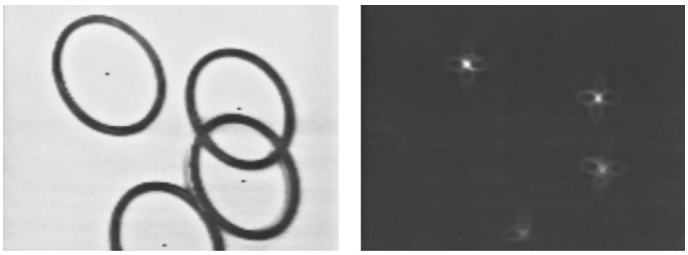


FIGURE 11.10

Applying the PSF to detection of tilted circles: (A) off-camera 128 × 128 image of a set of circular O-rings on a 45° slope of arbitrary direction; (B) transform in parameter space: notice the peculiar shape of the ellipse transform, which is close to a "four-leaf clover" pattern. (A) also indicates the positions of the centers of the O-rings as located from (B): accuracy is limited by the presence of noise, shadows, clutter, and available resolution, to an overall standard deviation of about 0.6 pixels.

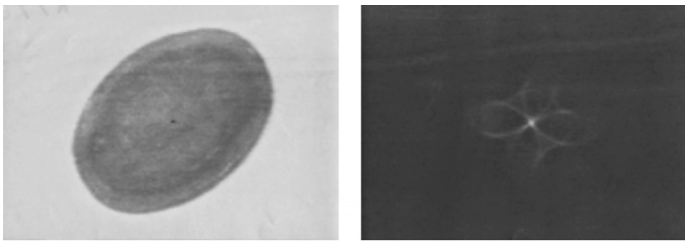


FIGURE 11.11

Applying the PSF to detection of elliptical objects: (A) off-camera 128 × 128 image of an elliptical bar of soap of arbitrary orientation; (B) transform in parameter space: in this case, the clover-leaf pattern is better resolved. Accuracy of location is limited partly by distortions in the shape of the object but the peak location procedure results in an overall standard deviation of the order of 0.5 pixels.

In Fig. 11.10, the O-rings are found accurately and with a fair degree of robustness, despite overlap and occlusion.

Fig. 11.10 also shows the arrangement of points in parameter space that results from applying the PSF to every edge point on the boundary of an ellipse: The pattern is somewhat clearer in Fig. 11.11. In either case, it is seen to contain a high degree of structure (curiously, the votes seem to form approximate "four-leaved clover" patterns). For an ideal transform, there would be no structure apart from the main peak, and all points on the PSF *not* falling on the peak at the center of the ellipse would be randomly distributed nearby. Nevertheless, the peak at the center is very well defined and confirms that this form of GHT is completely viable.

11.6 COMPARING THE VARIOUS METHODS FOR ELLIPSE DETECTION

This section briefly compares the computational loads for the methods of ellipse detection discussed above. To make fair comparisons, we concentrate on ellipse detection per se and ignore any additional procedures concerned with (1) finding other ellipse parameters, (2) distinguishing ellipses from other shapes, or (3) separating concentric ellipses. We start by examining the GHT method and the diameter bisection method.

First, suppose that an $N \times N$ pixel image contains p identical ellipses with semiaxes a, b, and two PDF-orientated parameters defined by $c = \frac{1}{2}(a + b)$, $d = \frac{1}{2}(a - b)$. By ignoring noise and general background clutter, we shall be favoring the diameter bisection method, as will be seen below. Next, the discussion is simplified by supposing that the computational load resides mainly

in the calculation of the positions at which votes should be accumulated in parameter space—the effort involved in locating edge pixels and in locating peaks in parameter space is much smaller.

Under these circumstances, the load for the GHT method may be approximated by the product of the number of edge pixels and the number of points per edge pixel that have to be accumulated in parameter space, the latter being equal to the number of points on the PSF. Hence, the load is proportional to:

$$\begin{aligned} L_{\mathrm{G}} &\approx p \times 2\pi c \times 2\pi(2d + d)/2 = 6\pi^2\, pcd \\ &\approx 60\, pcd \end{aligned} \tag{11.15}$$

where the ellipse has been taken to have relatively low eccentricity so that, as indicated in Section 11.5, the PSF itself approximates to an ellipse, and its semi-axes have the values $2d$ and d.

For the diameter bisection method, the actual voting is a minor part of the algorithm—as indeed it is in the GHT method (see the snippet of code listed in Table 11.1). In either case, most of the computational load concerns edge orientation calculations or comparisons. Assuming that these calculations and comparisons involve similar inherent effort, it is fair to assess the load for the diameter bisection method as:

$$\begin{aligned} L_{\mathrm{D}} &\approx {}^{p \times 2\pi c}C_2 \approx (2\pi pc)^2/2 \\ &\approx 20p^2c^2 \end{aligned} \tag{11.16}$$

Hence,

$$L_D/L_{\mathrm{G}} \approx pc/3d \tag{11.17}$$

When a is close to b, as for a circle, $L_{\mathrm{G}} \to 0$ and then the diameter bisection method becomes a poor option. However, in some cases, it is found that a is close to $2b$, so that c is close to $3d$. The ratio of the loads then becomes:

$$L_D/L_{\mathrm{G}} \approx p \tag{11.18}$$

It is possible that p will be as low as 1 in some cases: however, such cases are likely to be rare and are offset by applications where there is significant background image clutter and noise, or where all p ellipses have other edge detail giving irrelevant signals that can be considered a type of self-induced clutter (see the O-ring example of Fig. 11.10).

It is also possible that some of the pairs of edge points in the diameter bisection method can be excluded before they are considered, e.g., by giving every edge point a range of interaction related to the size of the ellipses. This would tend to reduce the computational load by a factor of the order of (but not as small as) p. However, the computational overhead required for this would not be negligible.

Overall, the GHT method should be significantly faster than the diameter bisection method in most real applications, the diameter bisection method being at a definite disadvantage when image clutter and noise are strong. By

comparison, the chord−tangent method always requires more computation than the diameter bisection method, as not only does it examine every pair of edge points but also it generates a *line* of votes in parameter space for each pair.

Against these, computational limitations must be noted the different characteristics of the methods. First, the diameter bisection method is not particularly discriminating, in that it locates many symmetrical shapes, as remarked earlier. The chord−tangent method is selective for ellipses but is not selective about their size or eccentricity. The GHT method is selective about all of these factors. These types of discriminability, or lack of it, can turn out to be advantageous or disadvantageous, depending on the application: hence, we do no more here than draw attention to the situation. It is also relevant that the diameter bisection method is rather less robust than the other methods: this is so as if one edge point of an antiparallel pair is not detected, then the other point of the pair cannot contribute to detection of the ellipse—a factor that does not apply for the other two methods as they take all edge information into account.

11.7 A GRAPH-THEORETIC APPROACH TO OBJECT LOCATION

This section considers a commonly occurring situation involving 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 from very few point features. As 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 known range. Clearly, noise and other artifacts such as occlusions modify this conclusion. In fact, when matching a template of the points in an idealized object with the points present in a real image, we may find that:

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, i.e., finding which subgraphs in the image graph are isomorphic to subgraphs of the idealized template graph. [Isomorphic means having the same basic shape and structure.] Of course, there may be a large number of matches involving rather few points: these would arise from sets of features that *happen* (see e.g., 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. Clearly, 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—i.e., with the greatest number of point matches per object.

Unfortunately, the schema presented above is still too simplistic in many applications as 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. Clearly, distortions lay 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 of achieving this is by constructing 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 then necessary to find regions of the match graph where the cross-linkages are maximized. To achieve this, *cliques* are sought within the match graph. A clique is a *complete subgraph*—i.e., 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, it is likely that the larger clique 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.

Fig. 11.12A 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 example is shown in Fig. 11.12B: there are nine possible feature assignments, six valid compatibilities, and four maximal cliques, only the largest corresponding to an exact match.

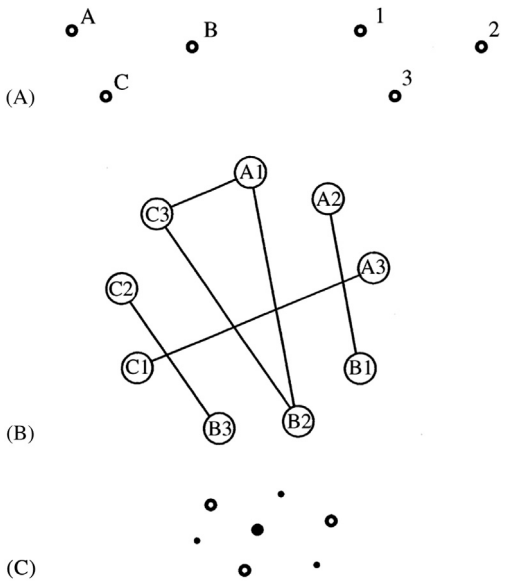


FIGURE 11.12

A simple matching problem—a general triangle: (A) basic labeling of model (*left*) and image (*right*); (B) match graph; (C) placement of votes in parameter space. In (B) the maximal cliques are: (1) A1, B2, C3; (2) A2, B1; (3) B3, C2; and (4) C1, A3. In (C), the following notation is used: ◦, positions of observed features; •, positions of votes; ●, position of main voting peak.

Fig. 11.13A shows the situation for the less trivial case of a quadrilateral, the match graph being shown in Fig. 11.13B. In this case, there are 16 possible feature assignments, 12 valid compatibilities, and seven maximal cliques. If occlusion of a feature occurs, this will (taken on its own) reduce the number of possible feature assignments and also the number of valid compatibilities: in addition, the number of maximal cliques and the size of the largest maximal clique will be reduced. On the other hand, noise or clutter can add erroneous features. If the latter are at arbitrary distances from existing features, then the number of possible feature assignments will be increased but there will not be any more compatibility in the match graph, so the latter will have only trivial additional complexity. However, if the extra features appear at *allowed* distances from existing features, this will introduce extra compatibilities into the match graph and make it more tedious to analyze. In the case shown in Fig. 11.14, both types of complication—an occlusion and an additional feature—arise: there are now eight pairwise assignments and six maximal cliques, rather fewer overall than in the original case of Fig. 11.13. However, the important factor is that the largest

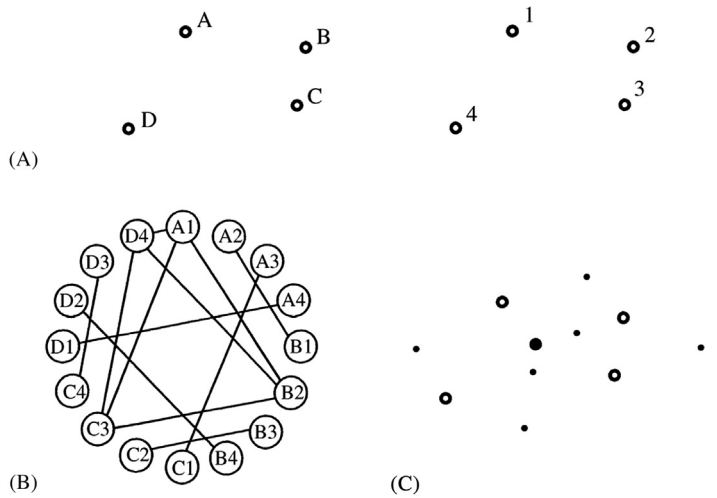


FIGURE 11.13

Another matching problem—a general quadrilateral: (A) basic labeling of model (*left*) and image (*right*); (B) match graph; (C) placement of votes in parameter space (notation as in Fig. 11.12).

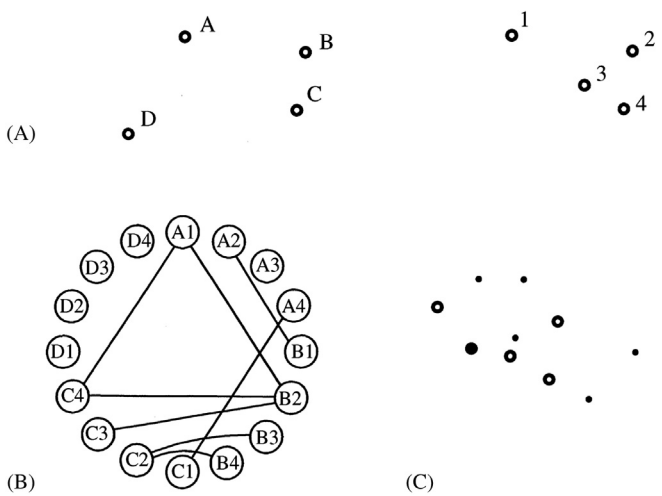


FIGURE 11.14

Matching when one feature is occluded and another is added: (A) basic labeling of model (*left*) and image (*right*); (B) match graph; (C) placement of votes in parameter space (notation as in Fig. 11.12).

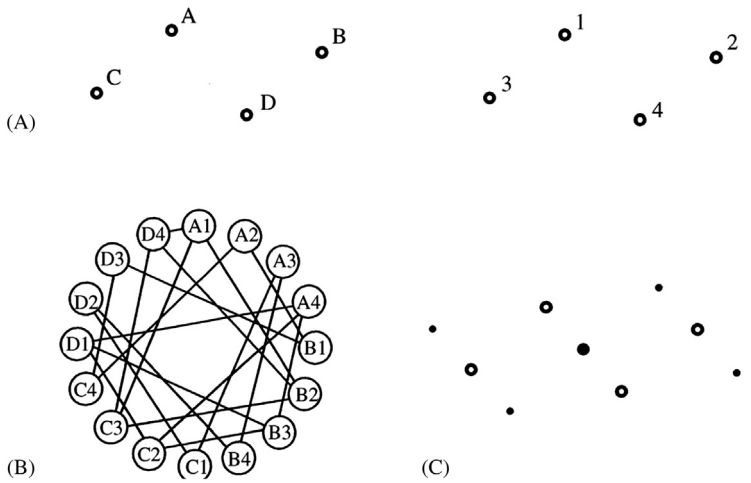


FIGURE 11.15

Matching a figure possessing some symmetry: (A) basic labeling of model (*left*) and image (*right*); (B) match graph; (C) placement of votes in parameter space (notation as in Fig. 11.12).

maximal clique still indicates the most likely interpretation of the image and that the technique is inherently highly robust.

When using methods such as the maximal clique approach which involve repetitive operations, it is useful to look for means of saving computation. In fact, when the objects being sought possess some symmetry, economies can be made. Consider the case of a parallelogram (Fig. 11.15). Here, the match graph has 20 valid compatibilities, and there are 10 maximal cliques. Of these, the largest two have equal numbers of nodes, and *both* identify the parallelogram within a symmetry operation. This means that the maximal clique approach is doing more computation than absolutely necessary: this can be avoided by producing a new "symmetry-reduced" match graph after relabeling the model template in accordance with the symmetry operations (see Fig. 11.16). This gives a much smaller match graph with half the number of pairwise compatibilities and half the number of maximal cliques. In particular, there is only one nontrivial maximal clique: note, however, that its size is not reduced by the application of symmetry.

11.7.1 A PRACTICAL EXAMPLE—LOCATING CREAM BISCUITS

Fig. 11.17A shows one of a pair of cream biscuits which are to be located from their "docker" holes—this strategy being advantageous as it has the potential for highly accurate product location prior to detailed inspection (in this case, the purpose is to locate the biscuits accurately from the holes, and then to check the alignment of the biscuit wafers and detect any excess cream around the sides of

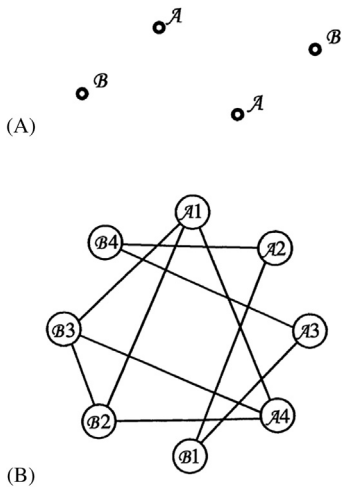


FIGURE 11.16

Using a symmetry-reduced match graph: (A) relabeled model template; (B) symmetry-reduced match graph.

the product). The holes found by a simple template-matching routine are indicated in Fig. 11.17B: the template used is rather small and, as a result, the routine is fairly fast but fails to locate all holes; in addition, it can give false alarms. Hence, an "intelligent" algorithm must be used to analyze the hole location data.

Clearly, this is a highly symmetrical type of object and so it should be beneficial to employ the symmetry-reduced match graph described above. To proceed, it is helpful to tabulate the distances between all pairs of holes in the object model (Fig. 11.18B). Then, this table can be regrouped to take account of symmetry operations (Fig. 11.18D). This will help when we come to build the match graph for a particular image. Analysis of the data in the above example shows that there are two nontrivial maximal cliques, each corresponding correctly to one of the two biscuits in the image. Note, however, that the reduced match graph does not give a *complete* interpretation of the image: it locates the two objects, but it does not confirm uniquely which hole is which. In particular, for a given starting hole of type A, it is not known which is which of the two holes of type B. This can be ascertained by applying simple geometry to the coordinates in order to determine (say) which hole of type B is reached by moving around the center hole E in a clockwise sense.

11.8 POSSIBILITIES FOR SAVING COMPUTATION

In these examples, the checking of which subgraphs are maximal cliques is a simple problem. However, in real matching tasks, it can quickly become

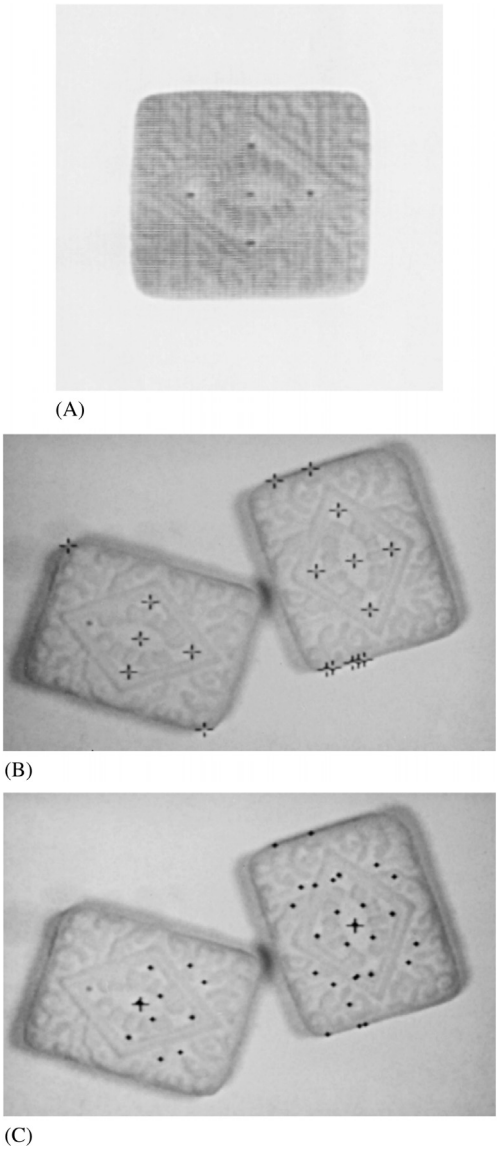
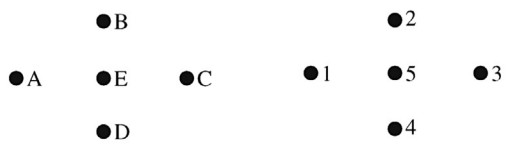


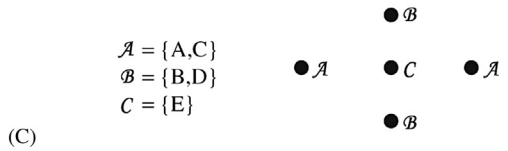
FIGURE 11.17

(A) A typical cream sandwich biscuit; (B) a pair of cream sandwich biscuits with crosses indicating the result of applying a simple hole detection routine; (C) the two biscuits reliably located by the GHT from the hole data in (B): the isolated small crosses indicate the positions of single votes.



	A	B	C	D	E
A	0	d	2a	d	a
B	d	0	d	2b	b
C	2a	d	0	d	a
D	d	2b	d	0	b
E	a	b	a	b	0

(B)



(C)

	$\mathcal{A}$	$\mathcal{B}$	$\mathcal{C}$
$\mathcal{A}$	0,2a	d	a
$\mathcal{B}$	d	0,2b	b
$\mathcal{C}$	a	b	0

(D)

FIGURE 11.18

Interfeature distances for holes on cream biscuits: (A) basic labeling of model (*left*) and image (*right*); (B) allowed distance values; (C) revised labeling of model using symmetric set notation; (D) allowed distance values. The cases of zero interfeature distance in the final table can be ignored as they do not lead to useful matches.

Table 11.2 A Simple Maximal Clique Algorithm

```
set clique size to 2;
// this is the size already included by the match graph
while (newcliques = true) { // new cliques still being found
  increment clique size;
  set newcliques = false;
  for all cliques of previous size {
    set all cliques of previous size to status maxclique;
    for all possible extra nodes
      if extra node is joined to all existing nodes in clique {
        store as a clique of current size;
        set newcliques = true;
      }
  }
  // the larger cliques have now been found
  for all cliques of current size
    for all cliques of previous size
      if all nodes of smaller clique are included in current clique
        set smaller clique to status not maxclique;
  // the subcliques have now been relabelled
}
```

unmanageable (the reader is encouraged to draw the match graph for an image containing two objects of seven points!).

Table 11.2 shows what is perhaps the most obvious type of algorithm for finding maximal cliques. It operates by examining in turn all cliques of a given number of nodes and finding what cliques can be constructed from them by adding additional nodes (bearing in mind that any additional nodes must be compatible

with all existing nodes in the clique). This permits all cliques in the match graph to be identified. However, an additional step is needed to eliminate (or relabel) all cliques that are included as subgraphs of a new larger clique before it is known which cliques are maximal.

In view of the evident importance of finding maximal cliques, many algorithms have been devised for the purpose. It is probable that the best of these is now close to the fastest possible speed of operation. Unfortunately, the optimum execution time is known to be bounded not by a polynomial in M (for a match graph containing maximal cliques of up to M nodes) but by a much faster varying function. Specifically, the task of finding maximal cliques is akin to the well-known traveling salesman problem and is known to be "NP-complete," implying that it runs in exponential time (see Section 11.9.1). Thus, whatever the run-time may be for values of M up to about 6, it will typically be 100 times slower for values of M up to about 10, and 100 times slower again for M greater than ~ 14. In practical situations, there are several ways of tackling this problem, which are as follows:

1. use the symmetry-reduced match graph wherever possible
2. choose the fastest available maximal clique algorithm
3. write critical loops of the maximal clique algorithm in machine code
4. build special hardware or multiprocessor systems to implement the algorithm
5. use the local−feature−focus (LFF) method (see below: this means searching for cliques of small M and then working with an alternative method)
6. use an alternative sequential strategy, which may however not be guaranteed to find all the objects in the image
7. use the GHT approach (see Section 11.9).

Of these methods, the first should be used wherever applicable. Methods 2−4 amount to improving the implementation and are subject to diminishing returns: note that the execution time varies so rapidly with M that even the best software implementations are unlikely to give a practical increase in M of more than 2 (i.e., $M \rightarrow M + 2$). Likewise, dedicated hardware implementations may only give increases in M of the order of 4 to 6. Method 5 is a "short-cut" approach which proves highly effective in practice. The idea is to search for specific subsets of the features of an object and then to hypothesize that the object exists and go back to the original image to check that it is actually present. Bolles and Cain (1982) devised this method when looking for hinges in quite complex images. In principle, the method has the disadvantage that the particular subset of an object that is chosen as a cue may be missing because of occlusion or some other artifact. Hence, it may be necessary to look for several such cues on each object. This is an example of further deviation from the matched filter paradigm, which reduces detection sensitivity yet again. The method is called the LFF method because objects are sought by cues or local foci.

The maximal clique approach is a type of exhaustive search procedure and is effectively a parallel algorithm. This has the effect of making it highly robust but

is also part of the reason for its slow speed. An alternative is to perform some sort of sequential search for objects, stopping when sufficient confidence is attained in the interpretation or partial interpretation of the image. For example, the search process may be terminated when a match has been obtained for a certain minimum number of features on a given number of objects. Such an approach may be useful in some applications and will generally be considerably faster than the full maximal clique procedure when M is greater than about 6. An analysis of several tree-search algorithms for subgraph isomorphism was carried out by Ullmann (1976): the paper tests algorithms using artificially generated data, and it is not clear how they relate to real images. The success or otherwise of all nonexhaustive search algorithms must, however, depend critically on the particular types of image data being analyzed: hence, it is difficult to give further general guidance on this matter (but see Section 11.11 for additional comments on search procedures).

The final method listed above is based on the GHT. In many ways, this provides an ideal solution to the problem as it presents an exhaustive search technique that is essentially equivalent to the maximal clique approach, while not falling into the NP-complete category. This may seem contradictory, as any approach to a well-defined mathematical problem should be subject to the mathematical constraints known to be involved in its solution. However, although the abstract maximal clique problem is known to be NP-complete, the *subset* of maximal clique problems that arises from 2-D image-based data may well be solved with less computation by other means, and in particular, by a 2-D technique. This special circumstance does appear to be valid but it naturally offers no possibility of solving general NP-complete problems by reference to the specific solutions found using the GHT approach! The GHT approach is described in the following section.

11.9 USING THE GHT FOR FEATURE COLLATION

This section describes how the GHT can be used as an alternative to the maximal clique approach to collate information from point features in order to find objects. Initially, we consider situations where objects have no symmetries—as for the cases of Figs. 11.12–11.14.

To apply the GHT, we first list all features and then accumulate votes in parameter space at every possible position of a localization point L consistent with each *pair* of features (Fig. 11.19). This strategy is particularly suitable in the present context, as it corresponds to the pairwise assignments used in the maximal clique method. To proceed, it is necessary merely to use the interfeature distance as a lookup parameter in the GHT R-table. For indistinguishable point features, this means that there must be two entries for the position of L for each value of the interfeature distance. Note that we have assumed that no symmetries exist and

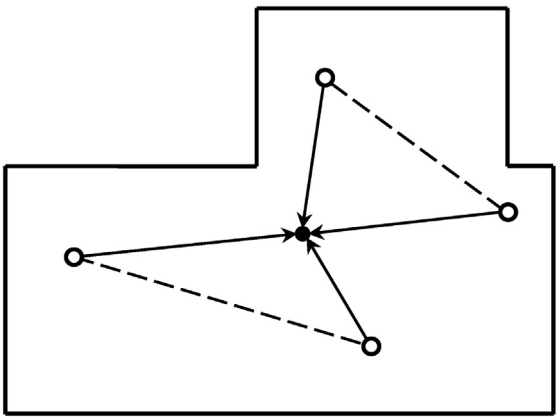


FIGURE 11.19

Method for locating L from pairs of feature positions: each pair of feature points gives two possible voting positions in parameter space, when objects have no symmetries. When symmetries are present, certain pairs of features may give rise to up to four voting positions: this is confirmed on careful examination of Fig. 11.17C.

that all pairs of features have different interfeature distances. If this were not so, then more than two vectors would have to be stored in the *R*-table per interfeature distance value.

To illustrate the procedure, it is applied first to the triangle example of Fig. 11.12. Fig. 11.12C shows the positions at which votes are accumulated in parameter space. There are four peaks with heights of 3, 1, 1, 1, it being clear that, in the absence of complicating occlusions and defects, the object is locatable at the peak of maximum size. Next, the method is applied to the general quadrilateral example of Fig. 11.13: this leads to seven peaks in parameter space, whose sizes are 6, 1, 1, 1, 1, 1, 1 (Fig. 11.13C).

Close examination of Figs. 11.12–11.14 indicates that every peak in parameter space corresponds to a maximal clique in the match graph. Indeed, there is a one-to-one relation between the two. In the uncomplicated situation being examined here, this is bound to be so for any general arrangement of features within an object, as every pairwise compatibility between features corresponds to two potential object locations, one correct and one that can be correct only from the point of view of that pair of features. Hence, the correct locations all add to give a large maximal clique and a large peak in parameter space, whereas the incorrect ones give maximal cliques each containing two wrong assignments and each corresponding to a false peak of size 1 in parameter space. This situation still applies even when occlusions occur or additional features are present (see Fig. 11.14). The situation is slightly more complicated when symmetries are present, the two methods each deviating in a different way: space does not permit the matter to be explored in depth here but the solution for the case of Fig. 11.15A is presented in

Fig. 11.15C. Overall, it seems simplest to imagine that there is still a one-to-one relationship between the solutions from the two approaches.

Finally, consider again the example of Section 11.7.1 (Fig. 11.17A), this time obtaining a solution by the GHT. Fig. 11.17C shows the positions of candidate object centers as found by the GHT. The small isolated crosses indicate the positions of single votes, and those very close to the two large crosses lead to voting peaks of weights 10 and 6 at these respective positions. Hence, object location is both accurate and robust, as required.

11.9.1 COMPUTATIONAL LOAD

This subsection compares the computational requirements of the maximal clique and GHT approaches to object location. For simplicity, imagine an image that contains just one wholly visible example of the object being sought. Moreover, suppose that the object possesses n features and that we are trying to recognize it by seeking all possible pairwise compatibilities, whatever their distance apart (as for all examples in Section 11.7).

For an object possessing n features, the match graph contains n^2 nodes (i.e., possible assignments), and there are ${}^{n^2}C_2 = n^2(n^2 - 1)/2$ possible pairwise compatibilities to be checked in building the graph. The amount of computation at this stage of the analysis is $O(n^4)$. To this must be added the cost of finding the maximal cliques. As the problem is NP-complete, the load rises at a rate which is faster than polynomial, and probably exponential in n^2 (Gibbons, 1985).

Now consider the cost of getting the GHT to find objects via pairwise compatibilities. As has been seen, the total height of all the peaks in parameter space is in general equal to the number of pairwise compatibilities in the match graph. Hence, the computational load is of the same order, $O(n^4)$. Next comes the problem of locating all the peaks in parameter space. In this case, parameter space is congruent to image space. Hence, for an $N \times N$ image, only N^2 points have to be visited in parameter space, and the computational load is $O(N^2)$. Note, however, that an alternative strategy is available in which a running record is kept of the relatively small numbers of voting positions in parameter space. The computational load for this strategy will be $O(n^4)$: although of a higher *order*, this often represents less computation in practice.

The reader may have noticed that the basic GHT scheme as outlined so far is able to locate objects from their features but does not determine their orientations. However, orientations can be computed by running the algorithm a second time and finding all the assignments that contribute to each peak. Alternatively, the second pass can aim to find a different localization point within each object. In either case, the overall task should be completed in little over twice the time, i.e., still in $O(n^4 + N^2)$ time.

Although the GHT at first appears to solve the maximal clique problem in polynomial time, what it actually achieves is to solve a real-space template-matching problem in polynomial time: it does not solve an *abstract* graph-theoretic problem

in polynomial time. The overall lesson is that the graph theory representation is not well matched to real space, not that real space can be used to solve abstract NP-complete problems in polynomial time.

11.10 GENERALIZING THE MAXIMAL CLIQUE AND OTHER APPROACHES

This section considers how the graph-matching concept can be generalized to cover alternative types of feature and also various attributes of features. The earlier discussion was restricted to point features and in particular to small holes. Corners were also taken as point features by ignoring attributes other than position coordinates. Thus, holes and corners seem to be ideal, in that they give maximum localization and hence maximum accuracy for object location.

Other types of feature generally have more than two specifying parameters, one of which may be contrast and the other size. This applies for most holes and circular objects, although for the smallest holes, it is sometimes most practicable to take the central dip in intensity as the measured parameter. Corners may have a number of attributes, including contrast, color, sharpness, and orientation, though these may not be known to high accuracy. Finally, more complex shapes such as ellipses have orientation, size, and eccentricity, and again contrast or color may be a usable attribute.

In fact, so much information is available that we need to consider how best to use it for locating objects. For convenience, this is discussed in relation to the maximal clique method. In fact, the answer is very simple. When compatibilities are being considered and the arcs are being drawn in the match graph, *any* available information may be taken into account in deciding whether a pair of features in the image matches a pair of features in the object model. In Section 11.7, the discussion was simplified by taking interfeature distances as the only relevant measurements. However, it is quite acceptable to describe the features in the object model more fully and to insist that they all match within prespecified tolerances. For example, holes and corners may be permitted to lead to a match only if the former are of the correct size, the latter are of the correct sharpness and orientation, and the interfeature distances are also appropriate. All relevant information has to be held in suitable lookup tables. In general, the gains easily outweigh the losses, as a considerable number of potential interpretations will be eliminated—hence, making the match graph significantly simpler and reducing, in many cases by a substantial factor, the amount of computation that is required to find the maximal cliques.

Overall, extrafeature attributes can be of great value in cutting down computation: they are also useful in reducing the possibility of erroneous interpretations.

11.11 SEARCH

The above sections have shown how the maximal clique approach may be used to locate objects in an image, or alternatively to label scenes according to predefined rules about what arrangements of regions are expected in scenes. In either case, the basic process being performed is that of search for solutions that are compatible with the observed data. This search takes place in assignment space, i.e., a space in which all combinations of assignments of observed features with possible interpretations exist. The problem is that of finding one or more valid sets of observed assignments.

It generally happens that the search space is very large, so that an exhaustive search for all solutions would involve enormous computational effort and would take considerable time. Unfortunately, one of the most obvious and appealing methods of obtaining solutions, the maximal clique approach, is NP-complete and can require impracticably large amounts of time to find solutions. It is therefore useful to clarify the nature of the maximal clique approach: to achieve this, we first describe the two main categories of search—breadth−first and depth−first search.

Breadth−first search is a form of search that systematically works down a tree of possibilities, never taking shortcuts to nearby solutions. Depth−first search, in contrast, involves taking as direct a path as possible to individual solutions, stopping the process when a solution is found and backtracking up the tree whenever a wrong decision is found to have been made. It is normal to curtail the depth−first search when sufficient solutions have been found, and this means that much of the tree of possibilities will not have been explored. Although breadth−first search can be curtailed similarly when enough solutions have been found, the maximal clique approach as described earlier is in fact a form of breadth−first search that is exhaustive and runs to completion.

In addition to being an exhaustive breadth−first search, the maximal clique approach may be described as being "blind" and "flat"—i.e., it involves neither heuristic nor hierarchical means of guiding the search. In fact, faster search methods involve guiding the search in various ways. First, heuristics are used to specify at various stages in which direction to proceed (which node of the tree to expand), or which paths to ignore (which nodes to prune). Second, the search can be made more "hierarchical," so that it searches first for outline features of a solution, returning later (perhaps in several stages) to fill in the details. Details of these techniques are omitted here. However, an interesting approach was used by Rummel and Beutel (1984): they searched images for industrial components using features such as corners and holes, alternating at various stages between breadth−first and depth−first search by using a heuristic based on a dynamically adjusted parameter: this being computed on the basis of how far the search is still away from its goal, and the quality of the fit so far. Rummel and Beutel noted the existence of a trade-off between speed and accuracy as a "guide factor," based on the number of features required for recognition, is adjusted—the problem being that trying to increase speed introduces some risk of not finding the optimum solution.

11.12 CONCLUDING REMARKS

The HT was introduced in Chapter 10, Line, Circle, and Ellipse Detection as a line detection scheme and then applied for detecting circles and ellipses. In that chapter, it appeared as a rather cunning method for aiding object detection; although it was seen to offer various advantages, particularly in its robustness in the face of noise and occlusion, there appeared to be no real significance in its rather novel voting scheme. The present chapter has shown that, far from being a trick method, the HT is much more general an approach than originally supposed: indeed, it embodies the properties of the spatial matched filter and is therefore capable of close-to-optimal sensitivity for object detection. However, this does not prevent its implementation from entailing considerable computational load, and significant effort and ingenuity has been devoted to overcoming this problem, both in general and in specific cases. The general case is tackled by the schemes discussed in the earlier sections of this chapter. It is important not to underestimate the value of specific solutions, both because such shapes as lines, circles, ellipses, and polygons cover a large proportion of (or approximations to) manufactured objects, and because methods for coping with specific cases have a habit (as for the original HT) of becoming more general as workers see possibilities for developing the underlying techniques.

To further underline the generality of the GHT, it has also been used for optimal location of lines of known length, by emulating a spatial matched filter detector; this result has been applied to the optimal detection of polygons and corners: for an example of the latter, see Fig. 11.20 (Davies, 1988a, 1989a).

This chapter has also discussed the problem of recognizing objects from point features. The maximal clique approach was seen to be capable of finding solutions to this task, but is limited in being NP-complete. Interestingly, the GHT has been found capable of performing the same task in polynomial time. That this is possible is because the graph theory representation is not well matched to the relevant real-space template-matching task in the way that the GHT is. Here, recall that the GHT is particularly well suited to object detection in real space as it is one type of spatial matched filter, whereas this cannot be said of the maximal clique approach.

Finally, note that, on an absolute scale, the graph-matching approach takes very little note of detailed image structure, using at most only pairwise feature attributes. This is adequate for 2-D image interpretation but inadequate for situations such as 3-D image analysis where there are more degrees of freedom to contend with (normally three degrees of freedom for position and three for orientation, for each object in the scene). Hence, more specialized and complex approaches need to be taken in such cases: these are examined in Part 3.

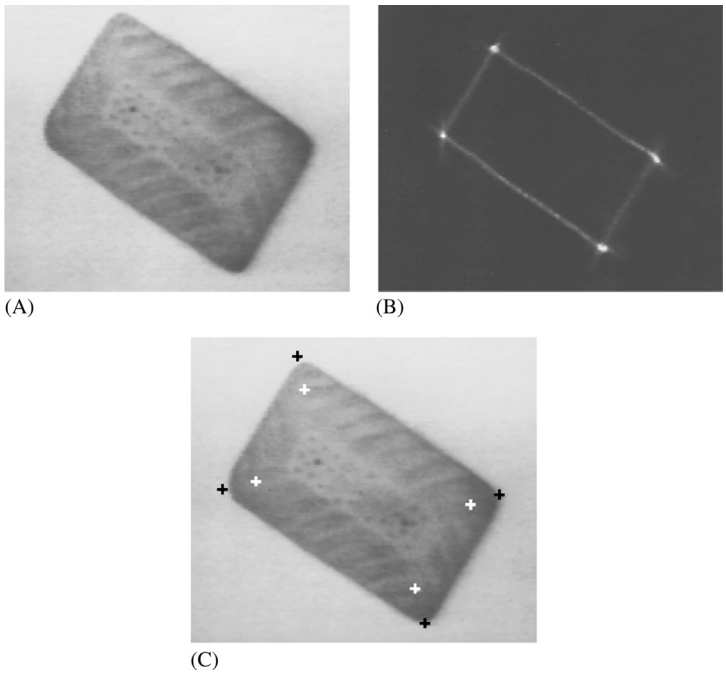


FIGURE 11.20

Example of the generalized Hough transform approach to corner detection. (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 crosses) and idealized corner positions deduced (black crosses). The lateral displacement employed here is close to the optimum for this type of object.

*Although the HT may appear to have a somewhat arbitrary design, this chapter has shown that it has solid roots in matched filtering, which in turn implies that votes should be gradient weighted for optimal sensitivity. The chapter also contrasts three methods for ellipse detection, showing how computational load may be estimated and minimized. In addition, searching for objects via their features is far more efficient than template matching. This chapter has shown that this raises the need to **infer** the presence of objects—a process that can still be computation intensive. In this respect, tests show that the GHT can be much more efficient than graph matching.*

11.13 BIBLIOGRAPHICAL AND HISTORICAL NOTES

Although the HT was introduced as early as 1962, a number of earlier ideas—including especially those of Merlin and Farber (1975) and Kimme

et al. (1975)—were required before the GHT could be developed (Ballard, 1981). By that time, the HT was already known to be formally equivalent to template matching (Stockman and Agrawala, 1977) and to spatial matched filtering (Sklansky, 1978).

By 1985, the computational load of the HT became the critical factor preventing its more general use—particularly as it could be used for most types of arbitrary shape detection, with well-attested sensitivity and considerable robustness. Li et al. (1985, 1986) showed the possibility of much faster peak location by using nonuniformly quantized parameter spaces. This work was developed further by Princen et al. (1989a,b) and Davies (1992g). An important development has been the randomized Hough transform, pioneered by Xu and Oja (1993) amongst others: it involves casting votes until specific peaks in parameter space become evident, thereby saving unnecessary computation.

Accurate peak location remains an important aspect of the HT approach. Properly, this is the domain of robust statistics which handles the elimination of outliers (see Appendix A). Davies (1992f) has shown a computationally efficient means of accurately locating HT peaks and has found why peaks sometimes appear narrower than a priori considerations would indicate (Davies, 1992b). Kiryati and Bruckstein (1991) have tackled aliasing effects which can arise with the HT, and which have the effect of cutting down accuracy.

Over time, the GHT approach has been broadened by geometric hashing, structural indexing, and other approaches (e.g., Lamdan and Wolfson, 1988; Gavrila and Groen, 1992; Califano and Mohan, 1994). At the same time, a probabilistic approach to the subject has been developed (Stephens, 1991) which puts it on a firmer footing. Grimson and Huttenlocher (1990) warn (possibly overpessimistically) against the blithe use of the GHT for complex object recognition tasks, because of the false peaks that can appear in such cases. For further review of the state of the subject up to 1993, see Leavers (1993).

In various chapters of Part 2, the statement has been made that the HT carries out a search leading to hypotheses that should be checked before a final decision about the presence of an object can be made. (A similar statement can be made in the case of graph-matching methods such as the maximal clique approach to object location.) However, Princen et al. (1994) show that the performance of the HT can be improved if it is itself regarded as a hypothesis testing framework: this is in line with the concept that the HT is a model-based approach to object location. Kadyrov and Petrou (2001) have developed the trace transform, which can be regarded as a generalized form of the Radon transform—itself closely related to the HT.

Other workers have used the HT for affine-invariant search: Montiel et al. (2001) made an improvement to reduce the incidence of erroneous evidence in the gathered data, whereas Kimura and Watanabe (2002) made an extension for 2-D shape detection that is less sensitive to the problems of occlusion and broken boundaries. Kadyrov and Petrou (2002) have adapted the trace transform to cope with affine parameter estimation.

In a generalization of the work of Atherton and Kerbyson (1999), and of Davies (1987a) on gradient weighting (see Section 11.4), Anil Bharath and his colleagues have examined how to optimize the sensitivity of the HT (private communication, 2004). Their method is particularly valuable in solving the problems of early threshold setting that limit many HT techniques. Similar sentiments come out in a different way in the work of Kesidis and Papamarkos (2000), which maintains the gray-scale information throughout the transform process, thereby leading to more exact representations of the original images.

Olson (1999) has shown that localization accuracy can be improved efficiently by transferring local error information into the HT and handling it rigorously. An important finding is that the HT can be divided into several subproblems without decrease in performance. This finding is elaborated in a 3-D model-based vision application where it is shown to lead to reduced false positive rates (Olson, 1998). Wu et al. (2002) extend the 3-D possibilities further by using a 3-D HT to find glasses: first a set of features are located that lie on the same plane, and this is then interpreted as the glasses rim plane. This approach allows the glasses to be separated from the face, and then they can be located in their entirety.

van Dijck and van der Heijden (2003) develop the geometric hashing method of Lamdan and Wolfson (1988) to perform 3-D correspondence matching using full 3-D hashing. This is found to have advantages in that knowledge of 3-D structure can be used to reduce the number of votes and spurious matches. Tuytelaars et al. (2003) describe how invariant-based matching and HTs can be used to identify regular repetitions in planes appearing within visual (3-D) scenes in spite of perspective skew: the overall system has the ability to reason about consistency and is able to cope with periodicities, mirror symmetries, and reflections about a point.

Graph-matching and clique-finding algorithms started to appear in the literature around 1970: for an early solution to the graph isomorphism problem, see Corneil and Gottlieb (1970). The subgraph isomorphism problem was tackled soon after by Barrow et al. (1972): see also Ullmann (1976). The double subgraph isomorphism (or subgraph−subgraph isomorphism) problem was commonly tackled by seeking maximal cliques in the match graph, and algorithms for achieving this have been described by Bron and Kerbosch (1973), Osteen and Tou (1973), and Ambler et al. (1975) (note that in 1989, Kehtarnavaz and Mohan reported preferring the algorithm of Osteen and Tou on the grounds of speed). Improved speed has also been achieved using the minimal match graph concept (Davies, 1991a).

Bolles (1979) applied the maximal clique technique to real-world problems (notably the location of engine covers) and showed how operation could be made more robust by taking additional features into account. By 1982, Bolles and Cain had formulated the LFF method, which (1) searches for restricted sets of features on an object, (2) takes symmetry into account to save computation, and (3) reconsiders the original image data in order to confirm a valid match: the paper gives various criteria for ensuring satisfactory solutions with this type of method.

Not satisfied with the speed of operation of maximal clique methods, other workers have tended to use depth−first search techniques. Rummel and Beutel (1984) developed the idea of alternating between depth−first and breadth−first search as dictated by the data—a powerful approach, although the heuristics that they used for this may well lack generality. Meanwhile, Kasif et al. (1983) showed how a modified GHT (the "relational HT") could be used for graph matching, although their paper gives few practical details. A somewhat different application of the GHT to perform 2-D matching was described in Section 11.9 and has been extended to optimize accuracy (Davies, 1992c). Geometric hashing has been developed to perform similar tasks on objects with complex polygonal shapes (Tsai, 1996).

Over the past decades inexact matching algorithms have acquired increasing predominance over exact matching methods, because of the ubiquitous presence of noise, distortions, and missing or added feature points, together with inaccuracies and thus mismatches of feature attributes. One class of work on inexact (or "error-tolerant") matching considers how structural representations should be compared (Shapiro and Haralick, 1985); this early work on similarity measures shows how the concept of "string edit distance" can be applied to graphical structures (Sanfeliu and Fu, 1983); the formal concept of edit distance was later extended by Bunke and Shearer (1998) and Bunke (1999), who considered and rationalized the cost functions for methods such as graph isomorphisms, subgraph isomorphisms, and maximum common subgraph isomorphisms: choice of cost functions was shown to be of crucial importance to success in each particular data set, though detailed analysis demonstrated important subtleties in the situation (Bunke, 1999).

Yet another class of work is that on optimization. This has included work on simulated annealing (Herault et al., 1990), genetic search (Cross et al., 1997), and neural processing (Pelillo, 1999). The work of Umeyama (1988) develops the least squares approach using a matrix eigendecomposition method to recover the permutation matrix relating the two graphs being matched. One of the most recent developments has been the use of spectral graph theory to recover the permutation structure. Spectral graph theory involves analysis of the structural properties of graphs using the eigenvalues and eigenvectors of the adjacency matrix. In fact, the Umeyama (1988) approach only matches graphs of the same size. Other related methods have emerged (e.g., Horaud and Sossa, 1995), but they have all suffered from an inability to cope with graphs of different sizes. However, Luo and Hancock (2001) have demonstrated how this particular problem can be overcome—by showing how the graph-matching task can be posed as maximum likelihood estimation using the EM algorithm formalism. Hence, singular value decomposition is used efficiently to solve correspondence problems. Ultimately, the method is important because it helps to move graph matching away from a discrete process in which a combinatorial search problem exists toward a continuous optimization problem which moves systematically toward the optimum solution. It ought to be added that the method works under considerable levels of

structural corruption—such as when 50% of the initial entries in the data-graph adjacency matrix are in error (Luo and Hancock, 2001). In a later development, Robles-Kelly and Hancock (2002) managed to achieve the same end, and to achieve even better performance within the spectral graph formalism itself.

Meanwhile, other developments included a fast, phased approach to inexact graph matching (Hlaoui and Wang, 2002); a reproducible kernel Hilbert space (RKHS) interpolator-based graph-matching algorithm capable of efficiently matching huge graphs of more than 500 vertices (e.g., those extracted from aerial scenes) on a PC (van Wyk et al., 2002). For a more detailed appraisal of inexact matching algorithms, see Lladós et al. (2001): note that the latter appears in a special section of IEEE Trans. PAMI on *Graph Algorithms and Computer Vision* (Dickinson et al., 2001).

11.13.1 MORE RECENT DEVELOPMENTS

Amongst the most recent developments are the following. Aragon-Camarasa and Siebert (2010) considered using the GHT for clustering SIFT feature matches. However, it turned out that a *continuous* rather than discretized HT space was needed for this application. This meant that each matched point had to be stored at the full machine precision in a Hough space consisting of a list data structure. Therefore, peak location had to take the form of standard unsupervised clustering algorithms. This was an interesting case where the intended GHT could not follow the standard voting and accumulating procedure. Assheton and Hunter (2011) also deviated sharply from the standard GHT approach when performing pedestrian detection and tracking: they used a shape-based voting algorithm based on Gaussian mixture models. The algorithm was stated to be highly effective for detecting pedestrians based on the silhouette shape. Chung et al. (2010) studied the problem of information retrieval from databases. They produced a region-based solution for object retrieval using the GHT and adaptive image segmentation. A key aspect of the overall scheme was the location of affine invariant MSERs (see Chapter 6: Corner, Interest Point, and Invariant Feature Detection) in the database and query images. Roy et al. (2011) applied the GHT to the detection and verification of seals (stamps) containing lettering and geometric patterns. This is a difficult problem because of the likely presence of noise, interfering text and signatures as well as incompleteness due to the application of uneven pressure to the stamp. In practice, a seal has to be located using scale and rotation invariant features (particularly text characters); it is then detected as a GHT peak resulting from application of a spatial feature descriptor of neighboring connected component pairs: i.e., in this application, the text characters in the seal are used as basic features for seal detection instead of individual edge or feature points. Memory demands are limited by splitting the R-table into two different look-up-tables—the character pair table and the distance table.

Silletti et al. (2011) have devised a variant approach to spectral graph matching in which new similarity measures are applied. The approach permits

application to a variety of types of image and yields results that are said to show significant improvements over certain preexisting methods. Gope and Kehtarnavaz (2007) have demonstrated a new method for affine matching between planar point sets. The method makes use of the convex hulls of the point sets and performs matching between them: this is a useful approach because (1) convexity is affine invariant, and (2) use of the convex hull is intrinsically robust. Property (2) follows from the fact that convex hulls are only locally altered by point perturbations including insertions and deletions. The method makes use of an enhanced modified Haussdorff distance and achieves better results in the presence of noise and occlusion than a number of standard methods. Aguilar et al. (2009) have developed a new "graph transformation matching" algorithm, to match points between pairs of images. It validates each match through the spatial configuration of the points by constructing a *k*-nearest-neighbor graph for each image; vertices that introduce structural dissimilarity between the graphs are iteratively eliminated, thereby yielding a consensus graph representing a correct set of point matches between the images.

11.14 PROBLEMS

1. **a.** Describe the main stages in the application of the HT to locate objects in digital images. What are the particular advantages offered by the HT technique? Give reasons why they arise.
   b. It is said that the HT only leads to *hypotheses* about the presence of objects in images, and that they should all be checked independently before making a final decision about the contents of any image. Comment on the accuracy of this statement.
2. Devise a GHT version of the spatial matched filter for detecting lines of known length L. Show that when used to detect an ideal line of length L, it gives a distributed response of length $2L$ that peaks at the center of the line, but when used to detect a partially occluded version of the line, it gives a response that is flat-topped over a range that includes the center of the line.
3. Show how a GHT version of the spatial matched filter can be devised to detect an equilateral triangle, leading to a star-shaped transform that peaks at the center the triangle. How may this approach be adapted for (1) a general triangle, (2) a regular polygon having N sides?
4. Find the match graph for a set of features arranged in the form of an isosceles triangle. Find how much simplification occurs by taking account of symmetry and using the symmetry-reduced match graph. Extend your results to the case of a kite (two isosceles triangles arranged symmetrically base-to-base).
5. Two lino-cutter blades (trapeziums) are to be located from their corners. Consider images in which two corners of one blade are occluded by the other blade. Sketch the possible configurations, counting the number of

corners in each case. If corners are treated like point features with no other attributes, show that the match graph will lead to an ambiguous solution. Show further that the ambiguity can in general be eliminated if proper account is taken of corner orientation. Specify how accurately corner orientation would need to be determined for this to be possible.

6. In problem 11.5, would the situation be any better if the GHT were used?

7. **a.** Metal flanges are to be located from their holes using a graph-matching (maximal clique) technique. Each bar has four identical holes at distances from the narrow end of the bar of 1, 2, 3, 5 cm, as shown in Fig. 11.P1. Draw match graphs for the four different cases in which one of the four holes of a given flange is obscured: determine in each case whether the method is able to locate the metal flange without any error, and whether any ambiguity arises.

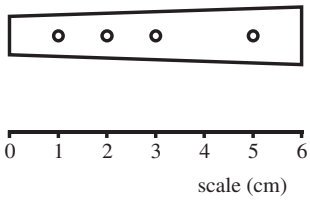


FIGURE 11.P1

Metal flanges for location using the GHT.

b. Do your results tally with the results for human perception? How would any error or ambiguity be resolved in practical situations?

8. **a.** Describe the maximal clique approach to object location. Explain why the largest maximal clique will normally represent the most likely solution to any object location task.

b. If symmetrical objects with four feature points are to be located, show that suitable labeling of the object template will permit the task to be simplified. Does the type of symmetry matter? What happens in the case of a rectangle? What happens in the case of a parallelogram? (In the latter case, see points A, B, C, D in Fig. 11.P2.)

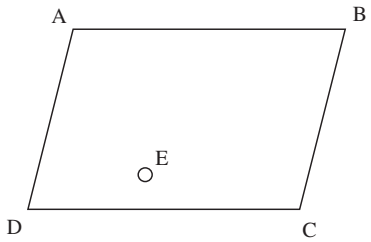


FIGURE 11.P2

Object with five feature points.

c. A nearly symmetrical object with *five* feature points (see Fig. 11.P2) is to be located. This is to be achieved by looking initially for the feature points A, B, C, D and ignoring the fifth point E. Discuss how the fifth point may be brought into play to finally determine the orientation of the object, using the maximal clique approach. What disadvantage might there be in adopting this two-stage approach?

9. a. What is template matching? Explain why objects are normally located from their features rather than using whole object templates. What are the features that are commonly used for this purpose?

b. Describe templates that can be used for corner and hole detection.

c. An improved type of lino-cutter blade (Fig. 11.P3) is to be placed into packs of six by a robot. Show how the robot vision system could locate the blades *either* from their corners *or* from their holes by applying the maximal clique method (i.e., show that *both* schemes would work).

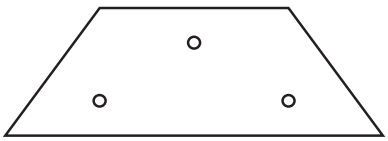


FIGURE 11.P3

Symmetrical lino-cutter blade.

d. After a time, it appears that the robot is occasionally confused when the blades overlap. It is then decided to locate the blades from their holes *and* their corners. Show why this helps to eliminate any confusion. Show also how finally distinguishing the corners from the holes can help in extreme cases of overlap.

10. a. A certain type of lino-cutter blade has four corners and two fixing holes (Fig. 11.P4). Blades of this type are to be located using the maximal clique technique. Assume the objects lie on a worktable and that they are viewed orthogonally at a known distance.

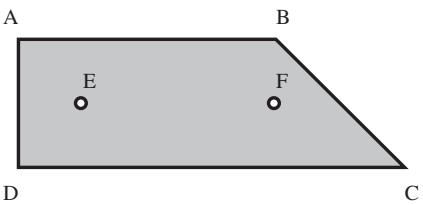


FIGURE 11.P4

Non-symmetrical lino-cutter blade.

b. Draw match graphs for the following situations:

i. the objects are to be located by their holes and their corners, regarding these as *indistinguishable* point features;

ii. the objects are to be located solely by their corners (i.e., matching corners in the image with corners on an idealized object);
iii. the objects are to be located solely by their holes;
iv. the objects are to be located by their holes *and* their corners, but these are to be regarded as *distinguishable* features.

c. Discuss your results with particular reference to:
i. the robustness that can be achieved;
ii. the speed of computation.

d. In the latter case, distinguish the time taken to build the basic match graph from the time taken to find all the maximal cliques in it. State any assumptions you make about the time taken to find a maximal clique of m nodes in a match graph of n nodes.

11. a. Decorative biscuits are to be inspected after first locating them from their holes. Show how the maximal clique graph-matching technique can be applied to identify and locate the biscuits shown in Fig. 11.P5A, which are of the same size and shape.

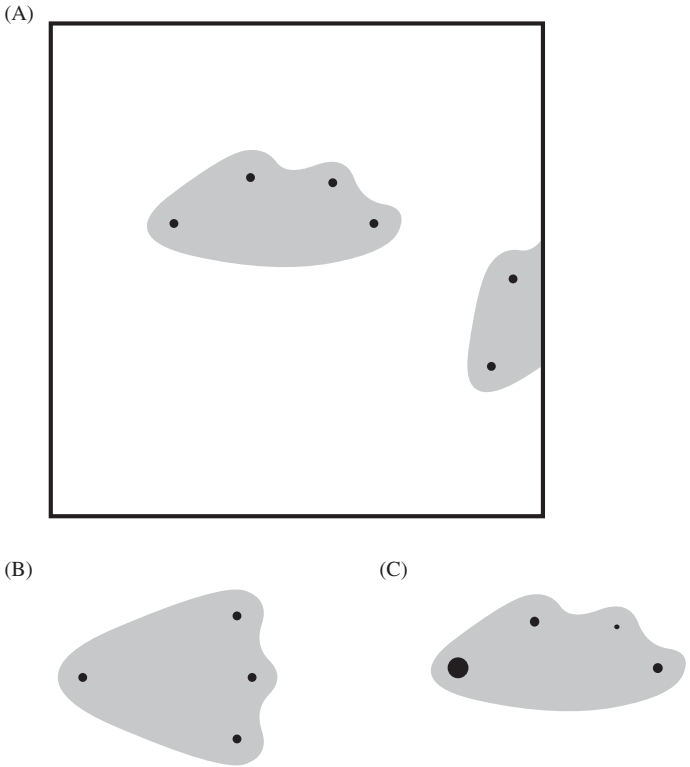


FIGURE 11.P5
Decorative biscuits for inspection.

b. Show how the analysis will be affected for biscuits which have an axis of symmetry, as shown in Fig. 11.P5B. Show also how the technique may be modified to simplify the computation for such a case.

c. A more detailed model of the first type of biscuit shows it has holes of *three* sizes, as shown in Fig. 11.P5C. Analyze the situation and show that a much simplified match graph can be produced from the image data, leading to successful object location.

d. A further matching strategy is devised to make use of the hole size information: matches are *only* shown in the match graph if they arise between pairs of holes of *different* sizes. Determine how successful this strategy is, and discuss whether it is likely to be generally useful, e.g., for objects with increased numbers of features.

e. Work out an optimal object identification strategy, which will be capable of dealing with cases where holes and/or corners are to be used as point features, where the holes might have different sizes, the corners might have different angles and orientations, the object surfaces might have different colors or textures, and objects might have larger numbers of features. Make clear what the term "optimal" should be taken to mean in such cases.

12. a. Fig. 11.P6 shows a 2-D view of a widget with four corners. Explain how the maximal clique technique can be used to locate widgets even if they are partly obscured by various types of object including other widgets.

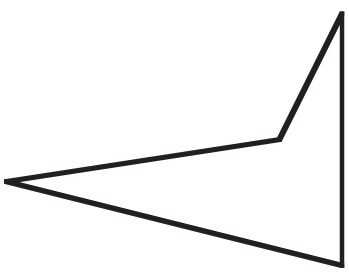


FIGURE 11.P6

Diagram of a widget.

b. Explain why the basic algorithm will not distinguish between widgets that are normally presented from those that are upside down. Consider how the basic method could be extended to ensure that a robot only picks up those that are the right way up.

c. The camera used to view the widgets is accidentally jarred and then reset at a different, unknown height above the worktable. State clearly why the usual maximal clique technique will now be unable to identify the widgets. Discuss how the overall program could be modified to make

sense of the data and make correct interpretations in which all the widgets are identified. Assume *first* that widgets are the only objects appearing in the scene, and *second* that a variety of other objects may appear.

d. The camera is jarred again, and this time is set at a small, unknown angle to the vertical. To be sure of detecting such situations and of correcting for them, a flat calibration object of known shape is to be stuck on the worktable. Decide on a suitable shape and explain how it should be used to make the necessary corrections.

13. Show that flat convex shapes remain convex under affine transformations.