CHAPTER

11 Line Detection

Detection of macroscopic features is an important part of image analysis and visual pattern recognition. Of particular interest is the identification of straight lines in images, as these are ubiquitous—appearing both on manufactured parts and in the built environment. The Hough transform (HT) provides the means for locating these features highly robustly in digital images. This chapter describes the processes and principles needed to achieve this.

Look out for:

- the basic HT technique for locating straight lines in images.
- alternative parametrizations of the HT.
- how lines can be localized along their length.
- how final line fitting can be made more accurate.
- why the HT is robust against noise and background clutter.
- how speed of processing can be improved.
- the RANSAC approach to line fitting.
- the relative efficiencies of RANSAC and the HT.
- how laparoscopic tools may be located.

While this chapter provides interesting methods for detecting line features in images, they may appear somewhat specialized. However, later chapters will show that both the HT and RANSAC (RANdom SAmpling Consensus) have much wider applicability than such arguments might suggest.

11.1 INTRODUCTION

Straight edges are among the most common features of the modern world, arising in perhaps the majority of manufactured objects and components, not least in the very buildings in which we live. Yet it is arguable whether true straight lines ever arise in the natural state: possibly the only example of their appearance in virgin

outdoor scenes is the horizon—although even this is clearly seen from space as a circular boundary. The surface of water is essentially planar, although it is important to realize that this is a deduction. The fact remains that straight lines seldom appear in completely natural scenes. Be all this as it may, it is clearly vital both in city pictures and in the factory to have effective means of detecting straight edges. This chapter studies available methods for locating these important features.

Historically, the Hough transform (HT) has been the main means of detecting straight edges, and since the method was originally invented (Hough, 1962) it has been developed and refined for this purpose. Hence this chapter concentrates on this particular technique; it also prepares the ground for applying the HT to the detection of circles, ellipses, corners, etc. in the next few chapters. We start by examining the original Hough scheme, even though it is now seen to be wasteful in computation, since important principles are involved. By the end of the chapter we shall see that the HT is not alone in its capabilities for line detection: RANSAC is also highly capable in this direction. In fact, both approaches have their advantages and limitations, as the discussion in Section 11.6 will show.

11.2 APPLICATION OF THE HOUGH TRANSFORM TO LINE DETECTION

The basic concept involved in locating lines by the HT is point−line duality. A point P can be defined either as a pair of coordinates or in terms of the set of lines passing through it. The concept starts to make sense if we consider a set of collinear points P_i, then list the sets of lines passing through each of them, and finally note that there is just one line that is common to all these sets. Thus, it is possible to find the line containing all the points P_i merely by eliminating those that are not multiple hits. Indeed, it is easy to see that if a number of noise points Q_j are intermingled with the signal points P_i, the method will be able to discriminate the collinear points from among the noise points at the same time as finding the line containing them, merely by searching for multiple hits. Thus, the method is inherently robust against noise, as indeed it is in discriminating against currently unwanted signals such as circles.

In fact, the duality goes further. For just as a point can define (or be defined by) a set of lines, so a line can define (or be defined by) a set of points, as is obvious from the above argument. This makes the above approach to line detection a mathematically elegant one and it is perhaps surprising that the Hough detection scheme was first published as a patent (Hough, 1962) of an electronic apparatus for detecting the tracks of high-energy particles, rather than as a paper in a learned journal.

The form in which the method was originally applied involves parametrizing lines using the slope−intercept equation:

$$y = mx + c \tag{11.1}$$

Every point on a straight edge is then plotted as a line in (m, c) space corresponding to all the (m, c) values consistent with its coordinates, and lines are detected in this space. The embarrassment of unlimited ranges of the (m, c) values (near-vertical lines require near-infinite values of these parameters) is overcome by using two sets of plots, the first corresponding to slopes of less than 1.0 and the second to slopes of 1.0 or more; in the latter case, Eq. (11.1) is replaced by the form:

$$x = \tilde{m}x + \tilde{c} \tag{11.2}$$

where

$$\tilde{m} = \frac{1}{m} \tag{11.3}$$

The need for this rather wasteful device was removed by the Duda and Hart (1972) approach, which replaces the slope−intercept formulation with the so-called "normal" (θ, ρ) form for the straight line (see Fig. 11.1):

$$\rho = x \cos\theta + y \sin\theta \tag{11.4}$$

To apply the method using this form, the set of lines passing through each point P_i is represented as a set of sine curves in (θ, ρ) space: for example, for point $P_1(x_1, y_1)$, the sine curve has equation:

$$\rho = x_1 \cos\theta + y_1 \sin\theta \tag{11.5}$$

Then multiple hits in (θ, ρ) space indicate, *via* their (θ, ρ) values, the presence of lines in the original image.

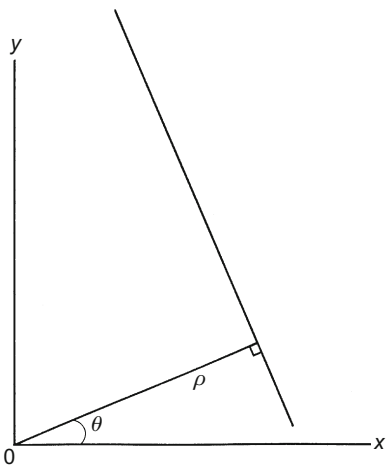


FIGURE 11.1

Normal (θ, ρ) parametrization of a straight line.

Each of the methods described above has the feature that it employs an "abstract" parameter space in which multiple hits are sought. Above we talked about "plotting" points in parameter space, but in fact the means of looking for hits is to seek peaks that have been built by *accumulation* of data from various sources. Although it might be possible to search for hits by logical operations such as use of the AND function, the Hough method gains considerably by *accumulating evidence* for events by a *voting scheme*. It will be seen below that this is the source of the method's high degree of robustness.

Although the methods described above are mathematically elegant and are capable of detecting lines (or sets of collinear points—which may be completely isolated from each other) amid considerable interfering signals and noise, they are subject to considerable computational problems. The reason for this is that every prominent point[1] in the original image gives rise to a great many votes in parameter space, so for a 256×256 image the (m, c) parametrization requires 256 votes to be accumulated, while the (θ, ρ) parametrization requires a similar number—360 if the θ quantization is to be fine enough to resolve 1° changes in line orientation.

A vital key to overcoming this problem was discovered by O'Gorman and Clowes (1976), who noted that points on lines are usually not isolated but instead are joined in fragments that are sufficiently long that their directions can be measured. Supposing that direction is known with high accuracy, it is then sufficient to accumulate just one vote in parameter space for every potential line point (in fact, if the local gradient is known with lesser accuracy then parameter space can be quantized more coarsely—say in 10° steps (O'Gorman and Clowes, 1976)—and again a single vote per point can be accumulated). Clearly, this method is much more modest in its computational requirements and it was soon adopted by other workers.

However, the computational load is still quite substantial: not only is a large two-dimensional (2-D) storage area needed but this must be searched carefully for significant peaks—a tedious task if short line segments are being sought. Various means have been tried for cutting down computation further. Dudani and Luk (1978) tackled the problem by trying to separate out the θ and ρ estimations. They accumulated votes first in a 1-D parameter space for θ—i.e., a histogram of θ values (it must not be forgotten that such a histogram is itself a simple form of HT).[2] Having found suitable peaks in the θ histogram, they then built a ρ histogram for all the points that contributed votes to a given θ peak, and repeated this for all θ peaks. Thus, two 1-D spaces replace the original 2-D parameter space, with very significant savings in storage and load. However, two-stage methods of

[1]For the present purpose it does not matter in what way these points are prominent. They may in fact be edge points, dark specks, centers of holes, and so on. Later we shall consistently take them to be edge points.

[2]It is now common for any process to be called an HT if it involves accumulating votes in a parameter space, with the intention of searching for significant peaks to find properties of the original data.

this type tend to be less accurate since the first stage is less selective: biased θ values may result from pairs of lines that would be well separated in a 2-D space. In addition, any error in estimating θ values is propagated to the ρ determination stage, making values of ρ even less accurate. For this reason Dudani and Luk added a final least-squares fitting stage to complete the accurate analysis of straight edges present in the image.

From a practical point of view, to proceed with either of the above methods of line detection, it is first necessary to obtain the local components of intensity gradient, and then to deduce the gradient magnitude g and threshold it to locate each edge pixel in the image. θ may be estimated using the arctan function in conjunction with the local edge gradient components g_x, g_y:

$$\theta = \arctan\left(\frac{g_y}{g_x}\right) \tag{11.6}$$

As the arctan function has period π, $\pm\pi$ may have to be added to obtain a principal value in the range $-\pi$ to $+\pi$: this can be decided from the signs of g_x and g_y.[3] Once θ is known, ρ can be found from Eq. (11.4).

Finally, note that straight lines and straight edges are different and need to be detected differently. (Straight *edges* are probably more common and appear as object boundaries, whereas straight *lines* are typified by telephone wires in outdoor scenes.) In fact, we have concentrated above on using the HT to locate straight edges, starting with edge detectors. Straight line segments may be located using Laplacian-type operators and their orientations are defined over a range $0-180^\circ$ rather than $0-360^\circ$: this makes HT design subtly different. For concreteness, in the remainder of this chapter, we concentrate on straight *edge* detection.

11.3 THE FOOT-OF-NORMAL METHOD

An alternative means of saving computation (Davies, 1986b) eliminates the use of trigonometric functions such as arctan by employing a different parametrization scheme. As noted earlier, the methods so far described all employ abstract parameter spaces in which points bear no immediately obvious visual relation to image space. In the alternative scheme, the parameter space is a second image space, which is congruent[4] to image space.

This type of parameter space is obtained in the following way. First, each edge fragment in the image is produced much as required previously so that ρ can be measured, but this time the foot of the normal from the origin is itself taken as a voting

[3]Note that in C++, the basic arctan function is *atan*, with a single argument, which should be g_y/g_x used as indicated above. However, the C++ *atan2* function has two arguments, and if g_y and g_x are used respectively for these, the function automatically returns an angle in the range $-\pi$ to π.

[4]That is, parameter space is like image space, *and* each point in parameter space holds information that is immediately relevant to the corresponding point in image space.

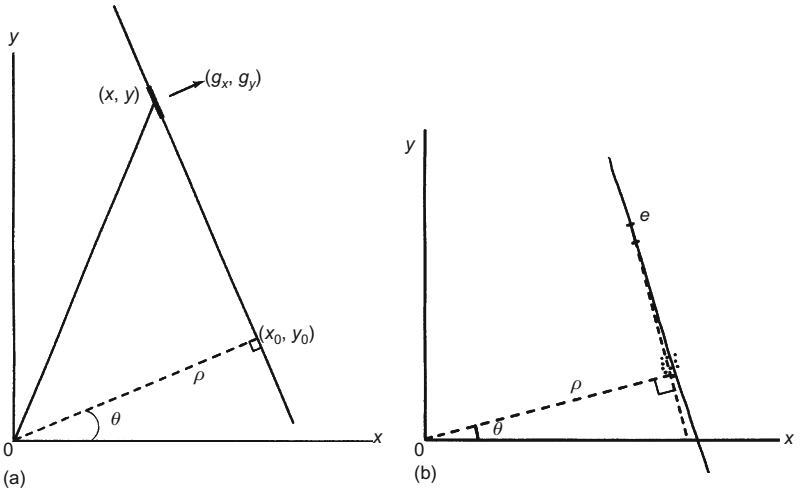


FIGURE 11.2

Image space parametrization of a straight line: (a) parameters involved in the calculation (see text); (b) buildup of foot-of-normal positions in parameter space for a more practical situation, where the line is not exactly straight: e is a typical edge fragment leading to a single vote in the parameter space.

Source: (b) © Unicom 1988

position in parameter space (Fig. 11.1). Clearly, the foot-of-normal position embodies all the information previously carried by the ρ and θ values, and mathematically the methods are essentially equivalent. However, the details differ, as will be seen.

The detection of straight edges starts with the analysis of (a) local pixel coordinates (x, y) and (b) the corresponding local components of intensity gradient (g_x, g_y) for each edge pixel. Taking (x_0, y_0) as the foot of the normal from the origin to the relevant line (produced if necessary—see Fig. 11.2), it is found that

$$\frac{g_y}{g_x} = \frac{y_0}{x_0} \tag{11.7}$$

$$(x - x_0)x_0 + (y - y_0)y_0 = 0 \tag{11.8}$$

These two equations are sufficient to compute the two coordinates (x_0, y_0). Solving for x_0 and y_0 gives

$$x_0 = vg_x \tag{11.9}$$

$$y_0 = vg_y \tag{11.10}$$

where

$$v = \frac{xg_x + yg_y}{g_x^2 + g_y^2} \tag{11.11}$$

Note that these expressions involve only additions, multiplications, and just one division, so voting can be carried out efficiently using this formulation.

11.3.1 Application of the Foot-of-Normal Method

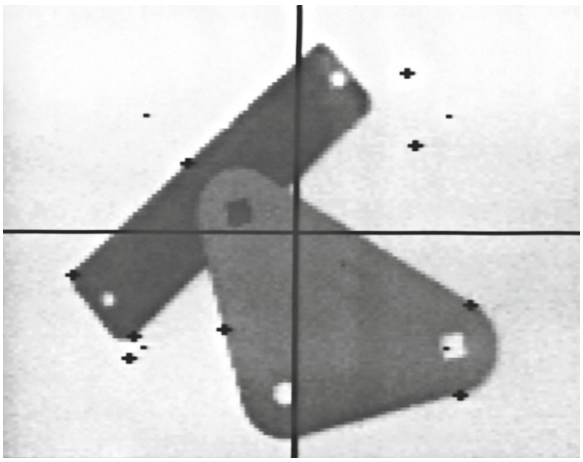
Although the foot-of-normal method is mathematically similar to the (θ, ρ) method, it is unable to determine line orientation directly with quite the same degree of accuracy. This is because the orientation accuracy depends on the fractional accuracy in determining ρ—which in turn depends on the absolute magnitude of ρ. Hence, for small ρ the orientation of a line that is predicted from the position of the peak in parameter space will be relatively inaccurate, even though the *position* of the foot-of-normal is known accurately. However, accurate values of line orientation can always be found by identifying the points that contributed to a given peak in the foot-of-normal parameter space and making them contribute to a θ histogram, from which line orientation may be determined accurately.

Typical results with the above method are shown in Fig. 11.3. Here, it was applied in subimages of size 64×64 within 128×128 images. Clearly, some of the objects in these pictures are grossly overdetermined by their straight edges, so low ρ values are not a major problem. For those peaks where $\rho > 10$, line orientation is estimated within approximately 2°; as a result, these objects are located within 1 pixel and orientated within 1° by this technique, without the necessity for θ histograms. Figure 11.4(a) and (b) contain some line segments that are not detected. This is due partly to their relatively low contrast, higher noise levels, above average fuzziness, or short length. However, it is also due to the thresholds set on the initial edge detector and on the final peak detector: when these were set at lower values, additional lines were detected but other noise peaks also became prominent in parameter space, and each of these needed to be checked in detail to confirm the presence of the corresponding line in the image. This is one aspect of a general problem that arises right through the field of image analysis.

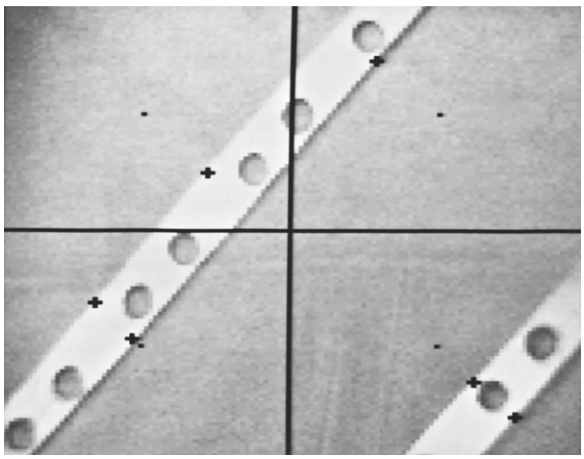
11.4 LONGITUDINAL LINE LOCALIZATION

The preceding sections have provided a variety of means for locating lines in digital images and finding their orientations. However, these methods are insensitive to where along an infinite idealized line an observed segment appears. The reason for this is that the fit includes only two parameters. There is some advantage to be gained in this, in that partial occlusion of a line does not prevent its detection. Indeed, if several segments of a line are visible, they can all contribute to the peak in parameter space, hence improving sensitivity. On the other hand, for full image interpretation, it is useful to have information about the longitudinal placement of line segments.

This is achieved by a further stage of processing. The additional stage involves finding which points contributed to each peak in the main parameter space and carrying out connectivity analysis in each case. Dudani and Luk (1978)



(a)



(b)

FIGURE 11.3

Results of image space parametrization of mechanical parts. The dot at the center of each quadrant is the origin used for computing the image−space transform. The crosses are the positions of peaks in parameter space that mark the individual straight edges (produced if necessary). For further explanation, see text.

Source: (a) © Unicom 1988

called this process "$xy-$grouping." It is not vital that the line segments should be 4- or 8-connected—just that there should be sufficient points on them so that adjacent points are within a threshold distance apart, i.e., groups of points are merged if they are within the prespecified distance (typically, 5 pixels). Finally, segments shorter than a certain minimum length (also typically ~ 5 pixels) can be ignored as too insignificant to help with image interpretation.

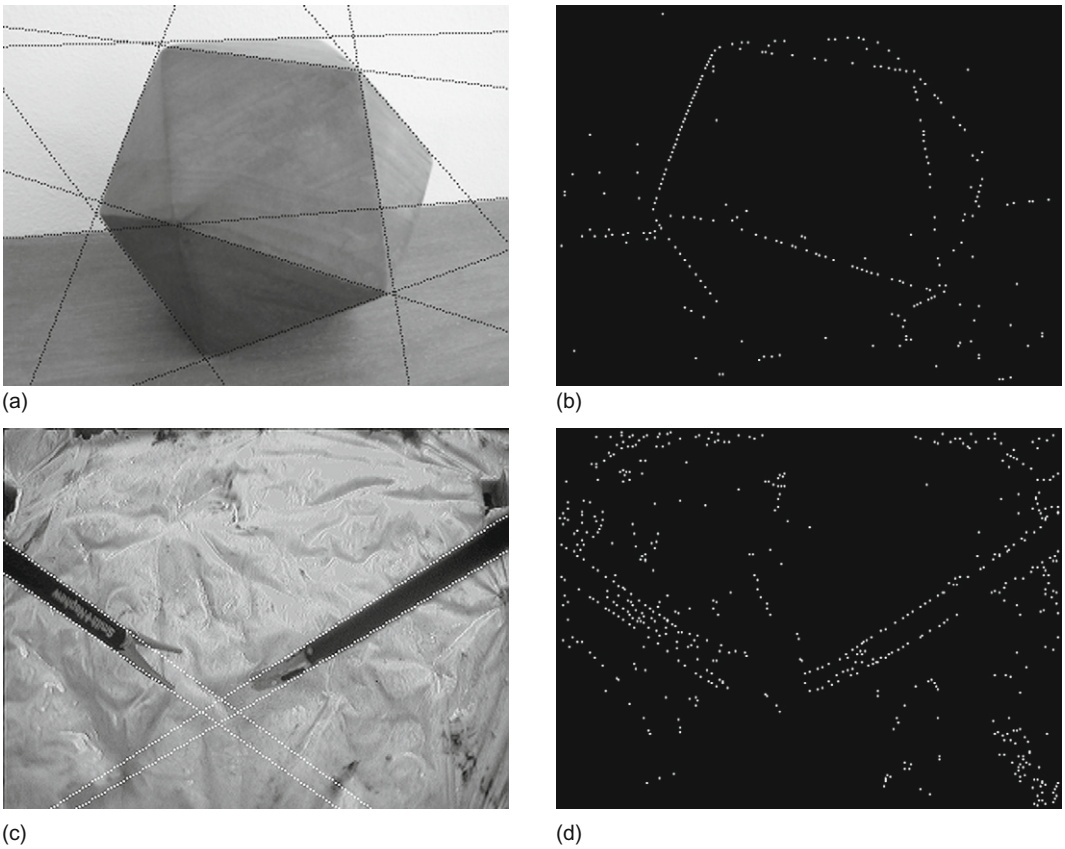


FIGURE 11.4

Straight line location using the RANSAC technique. (a) An original grayscale image with various straight edges located using the RANSAC technique; (b) the edge points fed to RANSAC for (a). These were isolated points that were local maxima of the gradient image. (c) The straight edges of a pair of laparoscopic tools—a cutter and a gripper—which have been located by RANSAC. (d) The points fed to RANSAC for (c). In (a) three edges of the icosahedron are missed. This is because they are roof edges with low-contrast and low-intensity gradient. RANSAC missed a fourth edge because of a lower limit placed on the level of support (see text).

11.5 FINAL LINE FITTING

Dudani and Luk (1978) found it useful to include a least-squares fitting stage to complete the analysis of the straight edges present in an image. This is carried out by taking the x, y coordinates of points on the line as input data and minimizing the sum of squares of distances from the data points to the best fit line. Their reason for adding such a stage was to eliminate local edge orientation accuracy as

a source of error. This motivation is generally reasonable if the highest possible accuracy is required (e.g., obtaining line orientation to significantly better than 1°). However, many workers have criticized the use of the least-squares technique, since it tends to weight up the contribution of less accurate points—including those points that have nothing to do with the line in question, but which arise from image "clutter." This criticism is probably justified, although the least-squares technique is convenient in yielding to mathematical analysis and in this case in giving explicit equations for θ and ρ in terms of the input data (Dudani and Luk, 1978). Dudani and Luk obtained the endpoints by reference to the best-fit line thus obtained. To understand the limitations of the least-squares technique, see the Appendix A.

11.6 USING RANSAC FOR STRAIGHT LINE DETECTION

RANSAC is an alternative model-based search schema that can often be used instead of the HT. In fact, it is highly effective when used for line detection, which is why the method is introduced here. The strategy can be construed as a voting scheme, but it is used in a different way from that in the HT. The latter operates by building up the evidence for instances of target objects in the form of votes in parameter space, and then making decisions about their existence (or by making hypotheses about their existence that are then finally checked out). RANSAC operates by making a sequence of hypotheses about the target objects, and determines the support for each of them by counting how many data points agree with them. As might be expected, for any potential target object, only the hypotheses with the maximum support are retained at each stage. This results in more compact information storage than for the HT: i.e., for RANSAC a list of hypotheses is held in current memory, whereas for the HT a whole parameter space, which is usually only sparsely populated, is held in memory. Thus, the RANSAC data are abstract lists, whereas the HT data can often be viewed as pictures in parameter space—as in the case of the foot-of-normal line detector. None of this prevents the RANSAC output being displayed (e.g., as straight lines) in image space, nor does it prevent the HT being accumulated using a list representation.

To explain RANSAC in more detail, we take the case of line detection. As for the HT, we start by applying an edge detector and locating all the edge points in the image. As we shall see, RANSAC operates best with a limited number of points, so it is useful to find the edge points that are local maxima of the intensity gradient image. (This does not correspond to the type of nonmaximum suppression used in the Canny operator, which produces thin connected *strings* of edge points but to individual *isolated* points: we shall return to this point below.) Next, to form a straight line hypothesis, all that is necessary is to take any pair of edge points from the list of N that remain after applying the local maximum operation. For each hypothesis we run through the list of N points finding how many points M support the hypothesis. Then we take more hypotheses (more pairs of edge

Table 11.1 Basic RANSAC Algorithm for Finding the Line with Greatest Support

```
Mmax = 0;
for all pairs of edge points do {
      find equation of line defined by the two points i, j;
      M = 0;
      for all N points in list do
            if (point k is within threshold distance d of line) M++;
      if (M > Mmax) {
            Mmax = M;
            imax = i;
            jmax = j;
            //this records the hypothesis giving the maximum support so far
      }
}
/* if Mmax > 0, (x[imax], y[imax]) and (x[jmax], y[jmax]) will be the coordinates of
the points defining the line having greatest support */
```

This algorithm only returns one line: in fact it returns the specific line model that has greatest support, for the line that has greatest support. Lines with less support are in the end ignored.

points) and at each stage retain only the one giving maximum support $M_{\max}$. This process is shown in Table 11.1.

The algorithm in Table 11.1 corresponds to finding the center of the highest peak in parameter space in the case of the HT. To find all the lines in the image, the most obvious strategy is the following: find the first line, then eliminate all the points that gave it support; then find the next line and eliminate all the points that gave it support; and so on until all the points have been eliminated from the list. The process may be written more compactly in the form:

```
repeat {
      find line;
      eliminate support;
}
until no data points remain;
```

Such a strategy carries the problem that if lines cross each other, support for the second line (which will necessarily be the weaker one) could be lesser support than it deserves. However, this should only be a serious disadvantage if the image is severely cluttered with lines. Nevertheless, the process is sequential and as such the results (i.e., the exact line locations) will depend on the order in which lines are eliminated, as the support regions will be minutely altered at each stage. Overall, the interpretation of complex images almost certainly has to proceed sequentially, and there is significant evidence that the human eye−brain system interprets images in this way, following early cues in order to progressively make sense of the data. Interestingly, the HT seems to escape from this by the potential capability for *parallel*

identification of peaks. While for simple images this may well be true, for complicated images containing many overlapping edges, there will again be the need for sequential analysis of the type envisaged above (e.g., see the back-projection method of Gerig and Klein, described in Section 13.11). The point is that with the particular list representation employed by RANSAC, we are *immediately* confronted with the problem of how to identify multiple targets, whereas for the reasons given above this does not immediately happen with the HT. Finally, on the plus side, successive elimination of support points necessarily makes it progressively easier and less computation intensive to find subsequent target objects. But the process itself is not cost-free, as the whole RANSAC procedure in Table 11.1 has to be run twice per line in order to identify the support points that have to be eliminated.

Next, we consider the computational load of the RANSAC process. If there are N edge points, the number of potential lines will be ${}^{N}C_2$, corresponding to a computational load of $O(N^2)$. However, finding the support for each line will involve $O(N)$ operations, so the overall computational load will be $O(N^3)$. In addition, the need to eliminate the support points for each line found will require computation proportional to the number of lines n, amounting to only $O(nN)$, and this will have little effect on the overall computational load.

One point that has not yet been made is that all N edge points will not arise from straight lines: some will arise from lines, some from curves, some from general background clutter, and some from noise. To limit the number of false positives, it will be useful to set a support threshold M_{thr} such that potential lines for which $M > M_{\text{thr}}$ are most likely to be true straight lines, while others are most likely to be artifacts, such as parts of curves or noise points. Thus, the RANSAC procedure can be terminated when $M_{\max}$ drops below M_{thr}. Of course, it may be required to retain only "significant" lines, e.g., those having length greater than L pixels. In that case, analysis of each line could allow many more points to be eliminated as the RANSAC algorithm proceeds. Another factor is whether hypotheses corresponding to pairs whose points are too close together should be taken into account. In particular, it might be considered that points closer together than 5 pixels would be superfluous as they would be likely to have much reduced chance of pointing along the direction of a line. However, it turns out that RANSAC is fail-safe in this respect, and there is some gain from keeping pairs with quite small separations, as some of the resulting hypotheses can actually be more accurate than any others. Overall, restricting pairs by their separations can be a useful way of reducing computational load, bearing in mind that $O(N^3)$ is rather high. Here, we should recall that the load for the HT is $O(N^2)$ during voting, if pairs of points are used, or $O(N)$, if single edge points and their gradients are used instead.

As we have just seen, RANSAC does not compare well with the HT regarding computational load, so it is better to employ RANSAC when N can be reduced in some way. This is why it is useful to use N local maxima rather than a full list of edge points in the form of strings of edge points generated by nonmaximum suppression or *a fortiori*, those existing before nonmaximum suppression. Indeed,

there is much to be gained by repeated random sampling[5] from the full list until sufficient hypotheses have been tested to be confident that all significant lines have been detected. In this way, the computational load is reduced from $O(N^3)$ to $O(N^2)$ or even $O(N)$ (it is difficult to predict the resulting computational complexity: in any case, the achievable computational load will be highly data dependent). Confidence that all significant lines have been detected can be obtained by estimating the risk that a significant line will be missed because no representative pair of points lying on the line was considered. This aspect of the problem will be examined more fully in Section A.6 of the appendix on Robust Statistics.

Before proceeding further, we shall briefly consider another way of reducing computational load. That is, by using the connected strings of edge points resulting from nonmaximum suppression, but eliminating those that are very short and coding the longer ones as isolated points every p pixels, where $p \approx 10$. In this way, there would be far fewer points than for our earlier paradigm, and also those that are employed would have increased coherence and probability of lying on straight edges; hence there would be a concentration on high quality data, while the $O(N^3)$ computation factor would be dramatically reduced. Clearly, in high noise situations this would not work well. It is left to the reader to judge how well it would work for the sets of edges located by the Canny operator in Figs. 5.7 and 5.8.

We are now in a position to consider actual results obtained by applying RANSAC to straight line detection. In the tests described, pairs of points were employed as hypotheses, and all edge points were local maxima of the intensity gradient. The cases shown in Fig. 11.4 correspond to detection of a block of wood in the shape of an icosahedron, and a pair of laparoscopic tools with parallel sides. Note that one line on the right of Fig. 11.4(a) was missed because a lower limit had to be placed on the level of support for each line. This was necessary because below this level of support the number of chance collinearities rose dramatically even for the relatively small number of edge points shown in Fig. 11.4(b), leading to a sharp rise in the number of false-positive lines. Figs. 23.2 and 23.3 show RANSAC being used to locate road lane markings. The same version of RANSAC was used in all the above cases, albeit in the case of Fig. 23.3 a refinement was added to allow improved elimination of points on lines that had already been located (this point will be clarified at the end of this section). Overall, this set of examples shows that RANSAC is a highly important contender for location of straight lines in digital images. Not discussed here is the fact that RANSAC is useful for obtaining robust fits to many other types of shape in 2-D and in 3-D.

It should be mentioned that one characteristic of RANSAC is that it is less influenced by aliasing along straight lines than the HT. This is because HT peaks tend to be fragmented by aliasing, so the best hypotheses can be difficult to obtain

[5]This corresponds to the original reason for the term RANSAC, which stands for RANdom SAmpling Consensus, "consensus" indicating that any hypothesis has to form a consensus with the available support data.

without excessive smoothing of the image. The reason why RANSAC wins in this context is that it does not rely on individual hypotheses being accurate: rather it relies on enough hypotheses easily being generatable, and by the same token, discardable.

Finally, we return to the above comment about obtaining improved deletion of points on lines that have already been located. Suppose the cross-section of a line is characterized by a (lateral) Gaussian distribution of edge points. As a true Gaussian extends to infinity in either direction, the support region is not well defined, but for high accuracy it is reasonable to take it as the region within $\pm\sigma$ of the centerline of the line. However, if just these points are eliminated, the remaining points near the line could later on give rise to alternative lines or combine with other points to lead to false positives. It is therefore better (Mastorakis and Davies, 2011) to make the "delete distance" d_d larger than the "fit distance" d_f that is used for support during detection, e.g., $d_d = 2\sigma$ or 3σ, where $d_f = \sigma$. ($d_d \approx 3\sigma$ can be considered to be close to optimal because 99.9% of the samples in a Gaussian distribution lie within $\pm 3\sigma$.) Figure 23.3 shows instances in which the fit distance is 3 pixels and the delete distance has values of 3, 6, 10, and 11 pixels, showing the advantages that can be gained by making d_d significantly greater than d_f. Figure 23.4 shows a flowchart of the algorithm used in this case.

11.7 LOCATION OF LAPAROSCOPIC TOOLS

Section 11.6 showed how RANSAC can provide a highly efficient means for locating straight edges in digital pictures, and gave an example of its use to locate the handles of laparoscopic tools. These are used for various forms of "keyhole" surgery: specifically, one tool (e.g., a cutter) might be inserted through one incision and another (e.g., a gripper) through a second incision. Additional incisions are needed for viewing *via* a laparoscope that employs optical fiber technology, and for inflating the cavity—e.g., the abdominal or chest cavity. In this section, we consider what information can be obtained *via* the laparoscope.

Figure 11.4(c) shows laparoscopic gripper and cutter tools being located in a simulated flesh background. The latter will normally be a wet surface that is largely red and will exhibit many regions that are close to being specularly reflective. The large variations in intensity that occur under these conditions make the scene quite difficult to interpret. While the surgeon who is in control of the instruments can learn a lot about the scene through tactile feedback and thus bolster his understanding of it, other people viewing the scene, e.g., on a TV monitor or computer, are liable to find it highly confusing. The same will apply for any computer attempting to interpret, analyze, or record the progress of an operation. These latter tasks are potentially important for logging operations, for training other doctors, for communicating with specialists elsewhere, or for analyzing the progress of the operation during any subsequent debriefing. It would therefore be useful if the exact locations,

FIGURE 11.5

Tips of laparoscopic tools located from the parts highlighted in gray.

orientations, and other parameters of the tools could be determined at least relative to the frame of reference of the laparoscope. To this end, RANSAC has provided important 2-D data on the location of the tool handles. Assessing the vanishing points of the pairs of lines from the handles also provides 3-D information. Clearly, further information can be obtained from the coordinates of the ends of the tools.

To identify the ends of the tools, the ends of the handles are first located. This is a straightforward task requiring knowledge of the exact ends of the RANSAC support regions for the tool handle edges. The remainder of the tool ends can now be located by initial approximate prediction, adaptive thresholding, and connected components analysis (Fig. 11.5), special attention being paid to accurate location of the tips of the tool ends. In Fig. 11.5, this is achieved within $\sim$1 pixel for the gripper on the left, and slightly less accurately for the closed cutter on the right. If the gripper had been open, accuracy would have been similar to that for the gripper. Note that, because of the complex intensity patterns in the background, it would have been difficult to locate the tool ends without first identifying the tool handles.

Each of the laparoscopic tools referred to above has (X, Y, Z) position coordinates, together with rotations ψ within the image plane (x, y), θ away from the image plane (toward the Z-axis), and φ about the axis of the handle; in addition, each tool end has an opening angle α (Fig. 11.6). It is bound to be difficult to obtain all seven parameters with any great accuracy from a single monocular view. However, in principle, using an exact CAD model of the tool end, this should be possible with $\sim 15-20^\circ$ accuracy for the angles. The 2-D information about the centerlines and widths of the handles, the convergence of the handle edges, and the exact positions of the tips of the tools, should together permit such a 3-D analysis to be carried out. Here, we have concentrated on the 2-D analysis: details of the relevant 3-D background theory needed to proceed further can be found in Part 3.

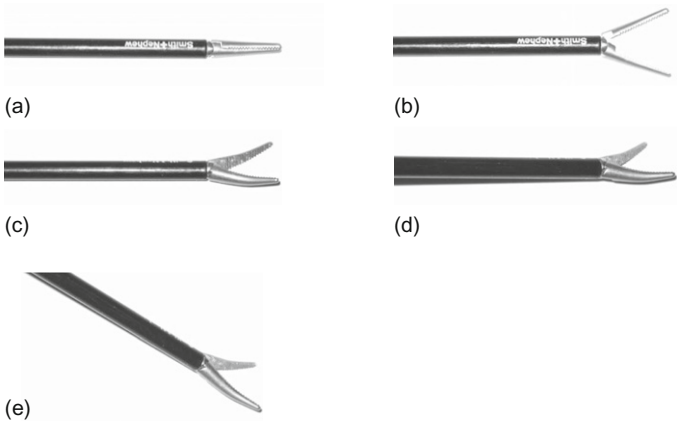


FIGURE 11.6

Orientation parameters for a laparoscopic tool. (a) A gripper tool with closed jaws. (b) Gripper with jaws separated by an angle α. (c) Gripper rotated through an angle φ about a horizontal axis. (d) Gripper tipped through an angle θ away from the image plane. (e) Gripper rotated through an angle ψ about the camera optical axis.

11.8 CONCLUDING REMARKS

This chapter has described a variety of techniques for finding straight lines and straight edges in digital images. Several of these were based on the HT, which is important because it permits systematic extraction of global data from images and has the capability to ignore "local" problems, due for example to occlusions and noise. This is what is required for "intermediate-level" processing, as will be seen repeatedly in later chapters.

The specific techniques covered have involved various parametrizations of a straight line, and means for improving efficiency and accuracy. In particular, speed is improved by using a two-stage line finding procedure—a method that is useful in other applications of the HT, as will be seen in later chapters. Accuracy tends to be cut down by such two-stage processing because of error propagation and because the first stage is liable to be subject to too many interfering signals. However, it is possible to improve the accuracy of approximate solutions by using least-squares refinement procedures.

In fact, by the end of the chapter it became clear that the RANSAC approach also has line fitting capabilities, which are for some purposes superior to those of the HT, although RANSAC tends to be more computation intensive (with N edge points it has a computational load of $O(N^3)$ rather than $O(N^2)$). Suffice it to say that the choice of which approach to use will depend on the exact type of image data, including levels of noise and background clutter.

The Hough transform is one way of inferring the presence of objects from their feature points, and RANSAC is another. Both methods can be said to use voting schemes to select best-fit lines, although only the HT employs a parameter space representation; and both methods are highly robust as they focus only on positive evidence for the existence of objects.

11.9 BIBLIOGRAPHICAL AND HISTORICAL NOTES

The HT was developed in 1962 (Hough, 1962) with the aim of finding (straight) particle tracks in high-energy nuclear physics and was brought into the mainstream image analysis literature much later by Rosenfeld (1969). Duda and Hart (1972) developed the method further and applied it to the detection of lines and curves in digital pictures. O'Gorman and Clowes (1976) soon developed a Hough-based scheme for finding lines efficiently, by making use of edge orientation information, at much the same time that Kimme et al. (1975) applied the same method (apparently independently) to the efficient location of circles. Many of the ideas for fast effective line finding described in this chapter arose in a paper by Dudani and Luk (1978). The author's foot-of-normal method (Davies, 1986b) was developed much later. During the 1990s, work in this area progressed further—see for example Atiquzzaman and Akhtar's (1994) method for the efficient determination of lines together with their end coordinates and lengths; Lutton et al.'s (1994) application of the transform to the determination of vanishing points; and Kamat-Sadekar and Ganesan's (1998) extensions of the technique to cover the reliable detection of multiple line segments, particularly in relation to road scene analysis. Other applications of the HT are covered in the next two chapters.

Some mention should be made of the related Radon transform. This is formed by integrating the picture function $I(x, y)$ along infinitely thin straight strips of the image, with normal coordinate parameters (θ, ρ), and recording the results in a (θ, ρ) parameter space. The Radon transform is a generalization of the Hough transform for line detection (Deans, 1981). In fact, for straight lines the Radon transform reduces to the Duda and Hart (1972) form of the HT that, as remarked earlier, involves considerable computation. For this reason the Radon transform is not covered in depth in this book. The transforms of real lines have a characteristic "butterfly" shape (a pencil of segments passing through the corresponding peak) in parameter space. This phenomenon has been investigated by Leavers and Boyce (1987), who have devised special 3×3 convolution filters for sensitively detecting these peaks.

Contrary to the view of some that the HT is completely worked over and no longer a worthwhile topic for research, there has been strong continuing interest in it. Indeed, the computational difficulties of the method reflect underlying matching problems that are inescapable in computer vision, so development of methods must continue. Thus, Schaffalitsky and Zisserman (2000) carried out an interesting extension of earlier ideas on vanishing lines and points by considering the case of repeated lines, such as those occurring on certain types of fences and

brick buildings; Song et al. (2002) developed HT methods for coping with the problems of fuzzy edges and noise in large-sized images; and Guru et al. (2004) demonstrated viable alternatives to the HT, based for example on heuristic search achieved by small eigenvalue analysis.

11.9.1 More Recent Developments

In 2010, advances were still being made in the application of the HT. In particular, Chung et al. (2010) developed an orientation-based elimination strategy that they have shown to be more efficient than previous line-determination methods based on the HT. It operates by dividing edge pixels into sets with small (typically 10°) ranges of orientation, and for each of these, it carries out the process of line detection. Since this process involves a parameter space of reduced size, both storage and search times are reduced.

The RANSAC procedure was published by Fischler and Bolles in 1981. This must be one of the most cited papers in computer vision, and the method must be one of the most used (more even than the HT, because it only relies on the existence of suitable hypotheses, *however* obtained). The original paper used it for tackling the full perspective *n*-point fitting problem in 3-D (see Chapter 16). Clarke et al. (1996) used it for locating and tracking straight lines. Borkar et al. (2009) used it for locating lane markings on roads, and Mastorakis and Davies (2011) developed it further for the same purpose. Interestingly, Borkar et al. used a low-resolution HT to feed RANSAC and followed it by least-squares fitting of the inliers. The paper does not report on how much was gained by this three-stage approach, either in accuracy or in reliability. (If enough hypotheses are employed—and there is certainly no lack of these in such an application—both the HT and least-squares fitting might be avoided, but here optimization for speed may make the inclusion of least squares essential.) For further discussion of RANSAC, see Chapter 23 and Appendix A.

11.10 PROBLEMS

1. **a.** In the foot-of-normal HT, straight edges are found by locating the foot of the normal F (x_f, y_f) from an origin O (0, 0) at the center of the image to an extended line containing each edge fragment E (x, y), and placing a vote at F in a separate image space.
   b. By examining the possible positions of lines within a square image and the resulting foot-of-normal positions, determine the exact extent of the parameter space that is theoretically required to form the HT.
   c. Would this form of the HT be expected to be (i) more or less robust and (ii) more or less computation intensive than the (ρ, θ) HT for line location?
2. **a.** Why is it sometimes stated that a HT generates *hypotheses* rather than actual solutions for object location? Is this statement justified?

b. A new type of HT is to be devised for detecting straight lines. It will take every edge fragment in the image and extend it in either direction until it meets the boundary of the image, and then accumulate a vote at each position. Thus *two* peaks should result for every line. Explain why finding these peaks will require *less* computation than for the standard HT, but that deducing the presence of lines will then require *extra* computation. How will these amounts of computation be likely to vary with (i) the size of the image and (ii) the number of lines in the image?

c. Discuss whether this approach would be an improvement on the standard approach for straight line location, and whether it would have any disadvantages.

3. a. Describe the *Hough transform* approach to object location. Explain its advantages relative to the *centroidal* (r, θ) *plot* approach. Illustrate your answer with reference to location of circles of known radius R.

b. Describe how the Hough transform may be used to locate straight edges. Explain what is seen in the parameter space if many curved edges also appear in the original image.

c. Explain what happens if the image contains a square object of arbitrary size and *nothing* else. How would you deduce from the information in parameter space that a square object is present in the image? Give the main features of an algorithm to decide that a square object is present and to locate it.

d. Examine in detail whether an algorithm using the strategy described in (c) would become confused if (i) parts of some sides of the square were occluded; (ii) one or more sides of the square were missing; (iii) several squares appeared in the image; (iv) several of these complications occurred together.

e. How important is it to this type of algorithm to have edge detectors that are capable of accurately determining edge orientation? Describe a type of edge detector that is capable of achieving this.