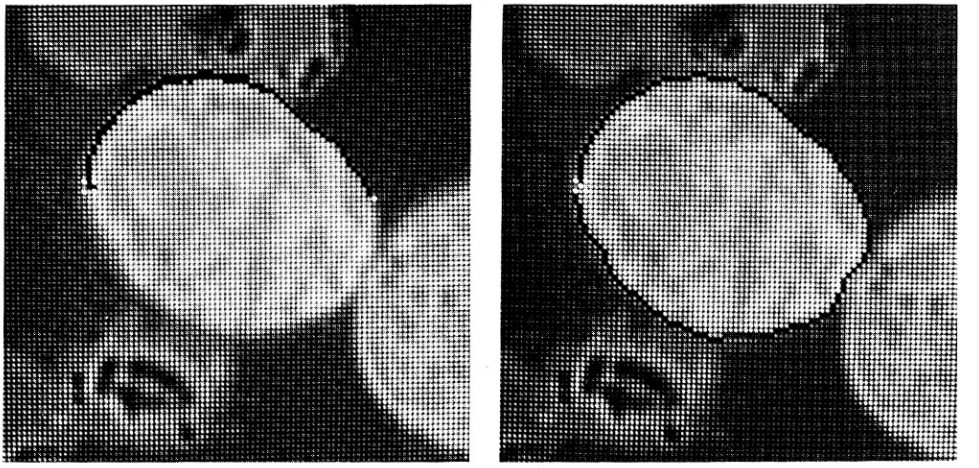


(a) (b)

Fig. 4.14 Using least maximum cost in heuristic search to find cell boundaries in microscope images. (a) A stage in the search process. (b) The completed boundary.

However, the price paid is the sacrifice of the mathematical guarantee of finding the least-cost path. This could be reflected in unsatisfactory boundaries. This method has been used in cineangiograms with satisfactory results [Ashkar and Modestino 1978].

4.5 EDGE FOLLOWING AS DYNAMIC PROGRAMMING

4.5.1 Dynamic Programming

Dynamic programming [Bellman and Dreyfus 1962] is a technique for solving optimization problems when not all variables in the evaluation function are interrelated simultaneously. Consider the problem

$$\max_{x_i} h(x_1, x_2, x_3, x_4) \tag{4.8}$$

If nothing is known about h, the only technique that guarantees a global maximum is exhaustive enumeration of all combinations of discrete values of $x_1, \ldots, x_4$. Suppose that

$$h(\cdot) = h_1(x_1, x_2) + h_2(x_2, x_3) + h_3(x_3, x_4) \tag{4.9}$$

x_1 only depends on x_2 in h_1. Maximize over x_1 in h_1 and tabulate the best value of $h_1(x_1, x_2)$ for each x_2:

$$f_1(x_2) = \max_{x_1} h_1(x_1, x_2) \tag{4.10}$$

Since the values of h_2 and h_3 do not depend on x_1, they need not be considered at

this point. Continue in this manner and eliminate x_2 by computing $f_2(x_3)$ as

$$f_2(x_3) = \max_{x_2}[f_1(x_2) + h_2(x_2, x_3)] \tag{4.11}$$

and

$$f_3(x_4) = \max_{x_3}[f_2(x_3) + h_3(x_3, x_4)] \tag{4.12}$$

so that finally

$$\max_{x_i} h = \max_{x_4} f_3(x_4) \tag{4.13}$$

Generalizing the example to N variables, where $f_0(x_1) = 0$,

$$f_{n-1}(x_n) = \max_{x_{n-1}}[f_{n-2}(x_{n-1}) + h_{n-1}(x_{n-1}, x_n)] \tag{4.14}$$

$$\max_{x_i} h(x_i, \ldots, x_N) = \max_{x_N} f_{N-1}(x_N)$$

If each x_i took on 20 discrete values, then to compute $f_N(x_{N+1})$ one must evaluate the maximand for 20 different combinations of x_N and x_{N+1}, so that the resultant computational effort involves $(N-1)20^2 + 20$ such evaluations. This is a striking improvement over exhaustive evaluation, which would involve 20^N evaluations of h!

Consider the artificial example summarized in Table 4.2. In this example, each **x** can take on one of three discrete values. The h_i are completely described by their respective tables. For example, the value of $h_i(0, 1) = 5$. The solution steps are summarized in Table 4.3. In step 1, for each x_2 the value of x_1 that maximizes $h_1(x_1, x_2)$ is computed. This is the largest entry in each of the columns of h. Store the function value as $f_1(x_2)$ and the optimizing value of x_1 also as a function of x_2. In step 2, add $f_1(x_2)$ to $h_2(x_2, x_3)$. This is done by adding f_1 to each row of h_2, thus computing the quantity inside the braces of (4.11). Now to complete step 2, for each x_3, compute the x_2 that maximizes $h_2 + f_1$ by selecting the largest entry in each row of the appropriate table. The rest of the steps are straightforward once these are understood. The solution is found by tracing back through the tables. For example, for $x_4 = 2$ we see that the best x_3 is -1, and therefore the best x_2 is 3 and x_1 is 1. This step is denoted by arrows.

Table 4.2

DEFINITION OF *h*

x_1 \ x_2	1	2	3
0	5	7	3
1	2	1	8
2	6	3	3

h_1

x_2 \ x_3	−1	0	1
1	1	7	1
2	1	1	3
3	5	6	2

h_2

x_3 \ x_4	1	2	3
−1	7	9	8
0	2	3	6
1	5	4	1

h_3

Table 4.3

METHOD OF SOLUTION USING DYNAMIC PROGRAMMING

Step 1

x_2	f_1	x_1
1	6	2
2	7	0
(3)	8	(1)

Step 2

x_2 \ x_3	−1	0	1
1	7	13	7
2	8	8	(10)
3	(13)	(14)	

x_3	f_2	x_2
(−1)	13	(3)
0	14	3
1	10	2

Step 3

x_3 \ x_4	1	2	3
−1	(20)	(22)	(21)
0	16	17	20
1	15	14	11

x_4	f_3	x_3
1	20	−1
(2)	(22)	(−1)
3	21	−1

Step 4: Optimal x_i's are found by examing tables (dashed line shows the order in which they are recovered).

Solution: $h^* = 22$
$x_1^* = 1, x_2^* = 3, x_3^* = -1, x_4^* = 2$

4.5.2 Dynamic Programming for Images

To formulate the boundary-following procedure as dynamic programming, one must define an evaluation function that embodies a notion of the "best boundary" [Montanari 1971; Ballard 1976]. Suppose that a local edge detection operator is ap-

plied to a gray-level picture to produce edge magnitude and direction information. Then one possible criterion for a "good boundary" is a weighted sum of high cumulative edge strength and low cumulative curvature; that is, for an n-segment curve,

$$h(\mathbf{x}_1, \ldots, \mathbf{x}_n) = \sum_{k=1}^{n} s(\mathbf{x}_k) + \alpha \sum_{k=1}^{n-1} q(\mathbf{x}_k, \mathbf{x}_{k+1}) \tag{4.16}$$

where the implicit constraint is that consecutive $\mathbf{x}_k$'s must be grid neighbors:

$$\|\mathbf{x}_k - \mathbf{x}_{k+1}\| \leqslant \sqrt{2} \tag{4.17}$$

$$q(\mathbf{x}_k, \mathbf{x}_{k+1}) = diff\,[\phi(\mathbf{x}_k), \phi(\mathbf{x}_{k+1})] \tag{4.18}$$

where α is negative. The function g we take to be edge strength, i.e., $g(x) = s(x)$. Notice that this evaluation function is in the form of (4.9) and can be optimized in stages:

$$f_0(\mathbf{x}_1) \equiv 0 \tag{4.19}$$

$$f_1(\mathbf{x}_2) = \max_{x_1} [s(\mathbf{x}_1) + \alpha q(\mathbf{x}_1, \mathbf{x}_2) + f_0(\mathbf{x}_1)] \tag{4.20}$$

$$f_k(\mathbf{x}_{k+1}) = \max_{x_k} [s(\mathbf{x}_k) + \alpha q(\mathbf{x}_k, \mathbf{x}_{k+1}) + f_{k-1}(\mathbf{x}_k)] \tag{4.21}$$

These equations can be put into the following steps:

Algorithm 4.5: Dynamic Programming for Edge Finding

1. Set $k = 1$.
2. Consider only $\mathbf{x}$ such that $s(\mathbf{x}) \geqslant T$. For each of these $\mathbf{x}$, define low-curvature pixels "in front of" the contour direction.
3. Each of these pixels may have a curve emanating from it. For $k = 1$, the curve is one pixel in length. Join the curve to $\mathbf{x}$ that optimizes the left-hand side of the recursion equation.
4. If $k = N$, pick the best f_{N-1} and stop. Otherwise, set $k = k + 1$ and go to step 2.

This algorithm can be generalized to the case of picking a *curve* emanating from $\mathbf{x}$ (that we have already generated): Find the end of that curve, and join the best of three curves emanating from the end of that curve. Figure 4.15 shows this process. The equations for the general case are

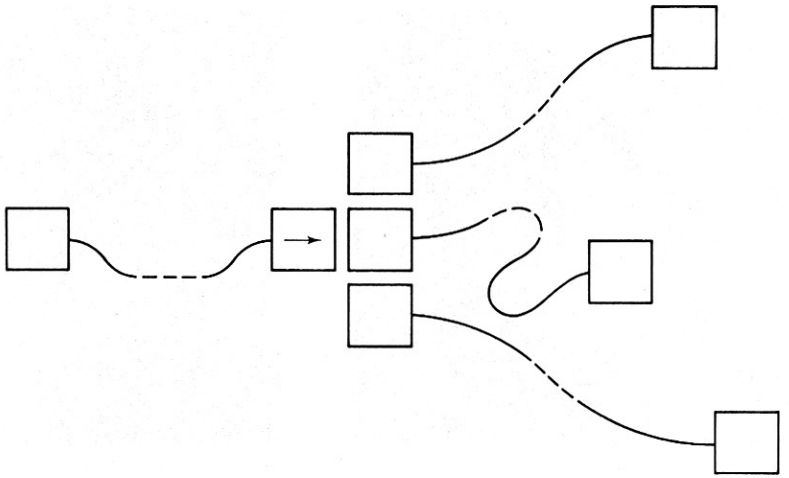


Fig. 4.15 DP optimization for boundary tracing.

$$f_0(\mathbf{x}_1) \equiv 0$$

$$f_l(\mathbf{x}_{k+1}) = \max_{x_k}[s(\mathbf{x}_k) + \alpha q(\mathbf{x}_k, t(\mathbf{x}_{k+1})) + f_{l-1}(\mathbf{x}_k)] \tag{4.22}$$

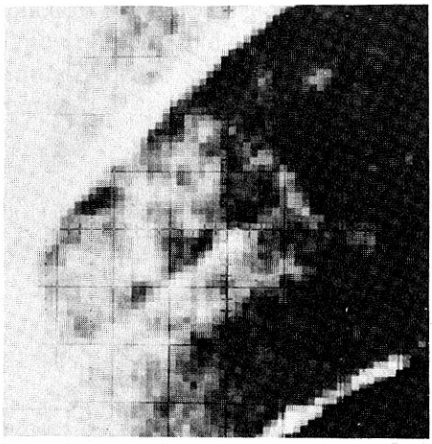
where the curve length n is related to α by a building sequence $n(l)$ such that $n(1) = 1$, $n(L) = N$, and $n(l) - n(l-1)$ is a member of $\{n(k) | k = 1, \ldots, l-1\}$. Also, $t(\mathbf{x}_k)$ is a function that extracts the tail pixel of the curve headed by $\mathbf{x}_k$. Further details may be found in [Ballard 1976].

Results from the area of tumor detection in radiographs give a sense of this method's performance. Here it is known that the boundary inscribes an approximately circular tumor, so that circular cues can be used to assist the search. In Fig. 4.16, (a) shows the image containing the tumor, (b) shows the cues, and (c) shows the boundary found by dynamic programming overlaid on the image.

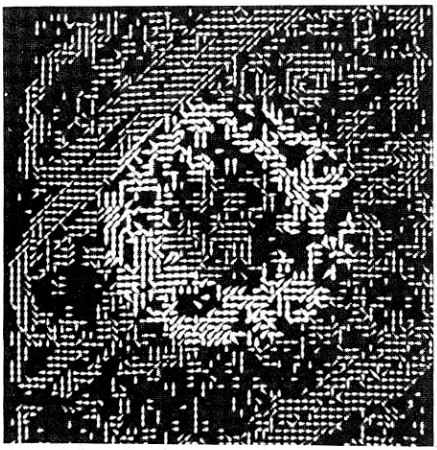
Another application of dynamic programming may be found in the pseudo-parallel road finder of Barrow [Barrow 1976].

4.5.3 Lower Resolution Evaluation Functions

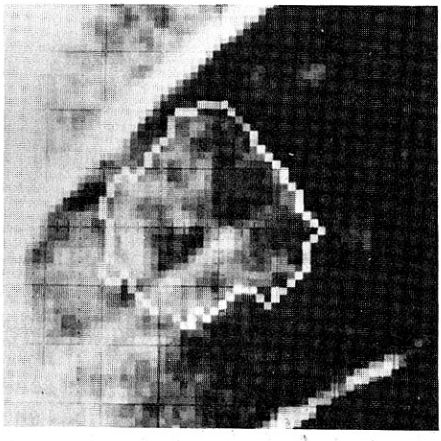
In the dynamic programming formulation just developed, the components $g(\mathbf{x}_k)$ and $q(\mathbf{x}_k, \mathbf{x}_{k+1})$ in the evaluation function are very localized; the variables $\mathbf{x}$ for successive s and q are in fact constrained to be grid neighbors. This need not be the case: The $\mathbf{x}$ can be very distant from each other without altering the basic technique. Furthermore, the functions g and q need not be local gradient and absolute curvature, respectively, but can be any functions defined on permissible $\mathbf{x}$. This general formulation of the problem for images was first described by [Fischler and



(a)



(b)



(c)

Fig. 4.16 Results of DP in boundary tracing. (a) Image containing tumor. (b) Contour cues. (c) Resultant boundary.

Elschlager 1973]. The Fischler and Elschlager formulation models an object as a set of parts and relations between parts, represented as a graph. Template functions, denoted by $g(\mathbf{x})$, measure how well a part of the model matches a part of the image at the point $\mathbf{x}$. (These local functions may be defined in any manner whatsoever.) "Relational functions," denoted by $q_{kj}(\mathbf{x}, \mathbf{y})$, measure how well the position of the match of the kth part at $(\mathbf{x})$ agrees with the position of the match of the jth part at $(\mathbf{y})$.

The basic notions are shown by a technique simplified from [Chien and Fu 1974] to find the boundaries of lungs in chest films. The lung boundaries are modeled with a polygonal approximation defined by the five key points. These points are the top of the lung, the two clavicle-lung junctions, and the two lower corners. To locate these points, local functions $g(\mathbf{x}_k)$ are defined which should be maximized when the corresponding point $\mathbf{x}_k$ is correctly determined. Similarly, $q(\mathbf{x}_k, \mathbf{x}_j)$ is a function relating points $\mathbf{x}_k$ and $\mathbf{x}_j$. In their case, Chien and Fu used the following functions:

$$T(\mathbf{x}) \equiv \text{template centered at } \mathbf{x} \text{ computed as an aggregate of a set of chest radiographs}$$

$$g(\mathbf{x}_k) = \sum_{\mathbf{x}} \frac{T(\mathbf{x} - \mathbf{x}_k) f(\mathbf{x})}{|T||f|}$$

and

$$\theta(\mathbf{x}_k, \mathbf{x}_j) = \text{expected angular orientation of } \mathbf{x}_k \text{ from } \mathbf{x}_j$$

$$q(\mathbf{x}_k \mathbf{x}_j) = \left[\theta(\mathbf{x}_k, \mathbf{x}_j) - \arctan \frac{y_k - y_j}{x_k - x_j} \right]$$

With this formulation no further modifications are necessary and the solution may be obtained by solving Eqs. (4.19) through (4.21), as before. For purposes of comparison, this method was formalized using a lower-resolution objective function. Figure 4.17 shows Chien and Fu's results using this method with five template functions.

4.5.4 Theoretical Questions about Dynamic Programming

The Interaction Graph

This graph describes the interdependence of variables in the objective function. In the examples the interaction graph was simple: Each variable depended on only two others, resulting in the graph of Fig. 4.18a. A more complicated case is the one in 4.18b, which describes an objective function of the following form:

$$h(\,) = h_1(x_1, x_2) + h_2(x_2, x_3, x_4) + h_3(x_3, x_4, x_5, x_6)$$

For these cases the dynamic programming technique still applies, but the computational effort increases exponentially with the number of interdependencies. For example, to eliminate x_2 in h_2, all possible combinations of x_3 and x_4 must be considered. To eliminate x_3 in h_3, all possible combinations of x_4, x_5, and x_6, and so forth.

Dynamic Programming versus Heuristic Search

It has been shown [Martelli 1976] that for finding a path in a graph between two points, which is an abstraction of the work we are doing here, heuristic search methods can be more efficient than dynamic programming methods. However, the point to remember about dynamic programming is that it efficiently builds paths from multiple starting points. If this is required by a particular task, then dynamic programming would be the method of choice, unless a very powerful heuristic were available.

4.6 CONTOUR FOLLOWING

If nothing is known about the boundary shape, but regions have been found in the image, the boundary is recovered by one of the simplest edge-following operations: "blob finding" in images. The ideas are easiest to present for binary images:

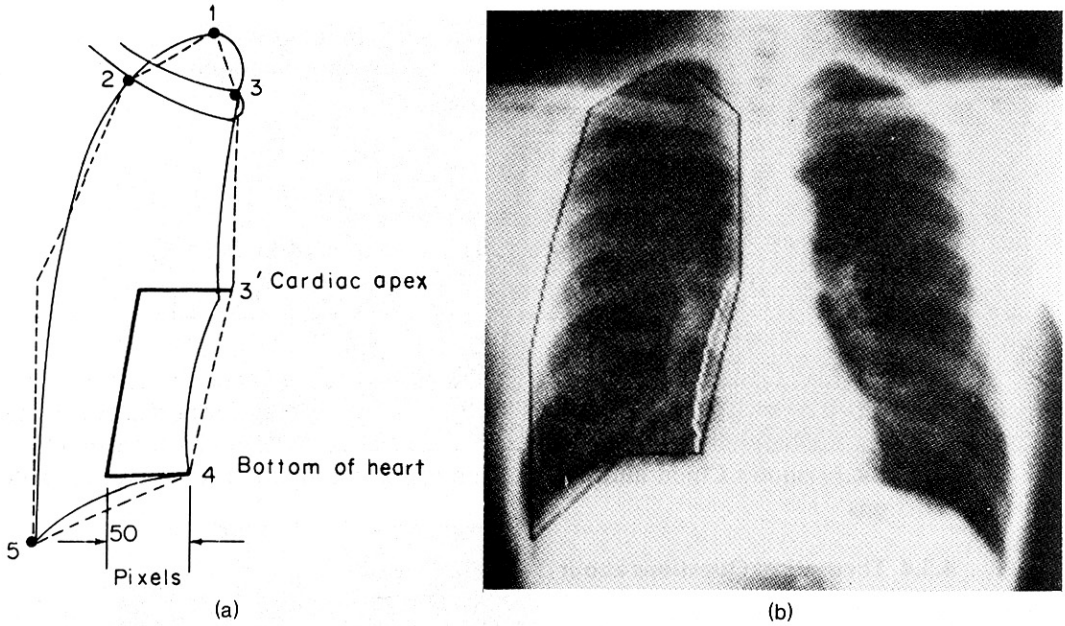


Fig. 4.17 Results of using local templates and global relations. (a) Model. (b) Results.

Given a binary image, the goal is find the boundaries of all distinct regions in the image.

This can be done simply by a procedure that functions like Papert's turtle [Papert 1973; Duda and Hart 1973]:

1. Scan the image until a region pixel is encountered.
2. If it is a region pixel, turn left and step; else, turn right and step.
3. Terminate upon return to the starting pixel.

Figure 4.19 shows the path traced out by the procedure. This procedure requires the region to be four-connected for a consistent boundary. Parts of an eight-connected region can be missed. Also, some bookkeeping is necessary to generate an exact sequence of boundary pixels without duplications.

A slightly more elaborate algorithm due to [Rosenfeld 1968] generates the boundary pixels exactly. It works by first finding a four-connected background pixel from a known boundary pixel. The next boundary pixel is the first pixel encountered when the eight neighbors are examined in a counter clockwise order from the background pixel. Many details have to be introduced into algorithms that follow contours of irregular eight-connected figures. A good exposition of these is given in [Rosenfeld and Kak 1976].

4.6.1 Extension to Gray-Level Images

The main idea behind contour following is to start with a point that is believed to be on the boundary and to keep extending the boundary by adding points in the contour directions. The details of these operations vary from task to task. The gen-

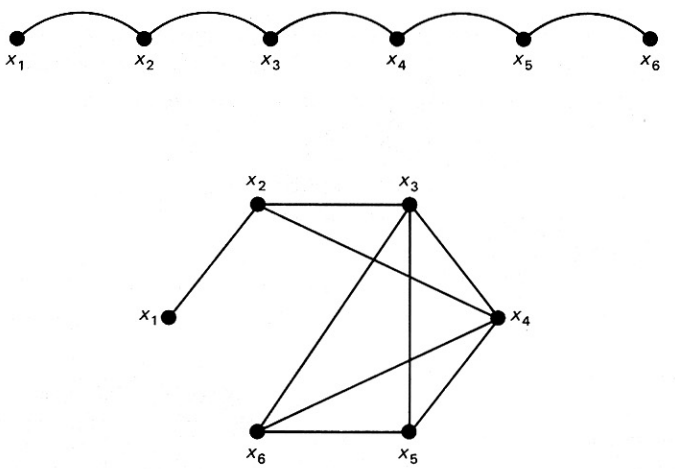


Fig. 4.18 Interaction graphs for DP (see text).

eralization of the contour follower to gray-level images uses local gradients with a magnitude $s(\mathbf{x})$ and direction $\phi(\mathbf{x})$ associated with each point $\mathbf{x}$. ϕ points in the direction of maximum change. If $\mathbf{x}$ is on the boundary of an image object, neighboring points on the boundary should be in the general direction of the contour directions, $\phi(\mathbf{x}) \pm \pi/2$, as shown by Fig. 4.20. A representative procedure is adapted from [Martelli 1976]:

1. Assume that an edge has been detected up to a point $\mathbf{x}_i$. Move to the point $\mathbf{x}_j$ adjacent to $\mathbf{x}_i$ in the direction perpendicular to the gradient of $\mathbf{x}_i$. Apply the gradient operator to $\mathbf{x}_j$; if its magnitude is greater than (some) threshold, this point is added to the edge.
2. Otherwise, compute the average gray level of the 3×3 array centered on $\mathbf{x}_j$, compare it with a suitably chosen threshold, and determine whether $\mathbf{x}_j$ is inside or outside the object.
3. Make another attempt with a point $\mathbf{x}_k$ adjacent to $\mathbf{x}_i$ in the direction perpendicular to the gradient at $\mathbf{x}_i$ plus or minus $(\pi/4)$, according to the outcome of the previous test.

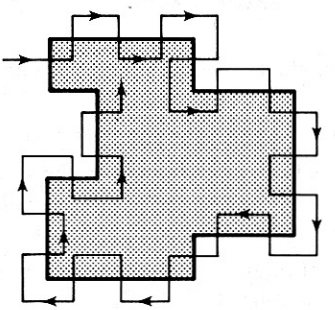


Fig. 4.19 Finding the boundary in a binary image.

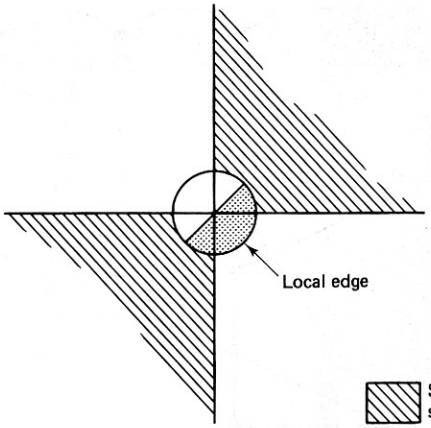


Fig. 4.20 Angular orientations for contour following.

4.6.2 Generalization to Higher-Dimensional Image Data

The generalization of contour following to higher-dimensional spaces is straightforward [Liu 1977; Herman and Liu 1978]. The search involved is, in fact, slightly more complex than contour following and is more like the graph searching methods described in Section 4.4. Higher-dimensional image spaces arise when the image has more than two spatial dimensions, is time-varying, or both. In these images the notion of a gradient is the same (a vector describing the maximum gray-level change and its corresponding direction), but the intuitive interpretation of the corresponding edge element may be difficult. In three dimensions, edge elements are primitive surface elements, separating volumes of differing gray level. The objective of contour following is to link together neighboring surface elements with high gradient modulus values and similar orientations into larger boundaries. In four dimensions, "edge elements" are primitive volumes; contour following links neighboring volumes with similar gradients.

The contour following approach works well when there is little noise present and no "spurious" boundaries. Unfortunately, if either of these conditions is present, the contour-following algorithms are generally unsatisfactory; they are easily thwarted by gaps in the data produced by noise, and readily follow spurious boundaries. The methods described earlier in this chapter attempt to overcome these difficulties through more elaborate models of the boundary structure.

EXERCISES

4.1 Specify a heuristic search algorithm that will work with "crack" edges such as those in Fig. 3.12.

4.2 Describe a modification of Algorithm 4.2 to detect parabolae in gray-level images.

4.3 Suppose that a relation $h(x_1, x_6)$ is added to the model described by Fig. 4.18a so that now the interaction graph is cyclical. Show formally how this changes the optimization steps described by Eqs. (4.11) through (4.13).

4.4 Show formally that the Hough technique without gradient direction information is equivalent to template matching (Chapter 3).

4.5 Extend the Hough technique for ellipses described by Eqs. (4.7a) through (4.7d) to ellipses oriented at an arbitrary angle θ to the x axis.

4.6 Show how to use the generalized Hough technique to detect hexagons.

REFERENCES

ASHKAR, G. P. and J. W. MODESTINO. "The contour extraction problem with biomedical applications." *CGIP 7*, 1978, 331–355.

BALLARD, D. H. *Hierarchic detection of tumors in chest radiographs.* Basel: Birkhäuser-Verlag (ISR-16), January 1976.

BALLARD, D. H. "Generalizing the Hough transform to detect arbitrary shapes." *Pattern Recognition 13*, 2, 1981, 111–122.

BALLARD, D. H. and J. SKLANSKY. "A ladder-structured decision tree for recognizing tumors in chest radiographs." *IEEE Trans. Computers 25*, 1976, 503-513.

BALLARD, D. H., M. MARINUCCI, F. PROIETTI-ORLANDI, A. ROSSI-MARI, and L. TENTARI. "Automatic analysis of human haemoglobin fingerprints." *Proc.*, 3rd Meeting, International Society of Haemotology, London, August 1975.

BARROW, H. G. "Interactive aids for cartography and photo interpretation." Semi-Annual Technical Report, AI Center, SRI International, December 1976.

BELLMAN, R. and S. DREYFUS. *Applied Dynamic Programming.* Princeton, NJ: Princeton University Press, 1962.

BOLLES, R. "Verification vision for programmable assembly." *Proc.*, 5th IJCAI, August 1977, 569-575.

CHIEN, Y. P. and K. S. FU. "A decision function method for boundary detection." *CGIP 3*, 2, June 1974, 125-140.

DUDA, R. O. and P. E. HART. "Use of the Hough transformation to detect lines and curves in pictures." *Commun. ACM 15*, 1, January 1972, 11–15.

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

FISCHLER, M. A. and R. A. ELSCHLAGER. "The representation and matching of pictoral patterns." *IEEE Trans. Computers 22*, January 1973.

HERMAN, G. T. and H. K. LIU. "Dynamic boundary surface detection." *CGIP 7*, 1978, 130-138.

HOUGH, P. V. C. "Method and means for recognizing complex patterns." U.S. Patent 3,069,654; 1962.

KELLY, M.D. "Edge detection by computer using planning." In *MI6*, 1971.

KIMME, C., D. BALLARD, and J. SKLANSKY. "Finding circles by an array of accumulators." *Commun. ACM 18*, 2, 1975, 120-122.

LANTZ, K. A., C. M. BROWN and D. H. BALLARD. "Model-driven vision using procedure decription: motivation and application to photointerpretation and medical diagnosis." *Proc.*, 22nd International Symp., Society of Photo-optical Instrumentation Engineers, San Diego, CA, August 1978.

LESTER, J. M., H. A. WILLIAMS, B. A. WEINTRAUB, and J. F. BRENNER, "Two graph searching techniques for boundary finding in white blood cell images." *Computers in Biology and Medicine 8*, 1978, 293-308.

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

MARR, D. "Analyzing natural images; a computational theory of texture vision." Technical Report 334, AI Lab, MIT, June 1975.

MARTELLI, A. "Edge detection using heuristic search methods." *CGIP 1*, 2, August 1972, 169-182.

MARTELLI, A. "An application of heuristic search methods to edge and contour detection." *Commun. ACM 19*, 2, February 1976, 73–83.

MONTANARI, U. "On the optimal detection of curves in noisy pictures." *Commun. ACM 14*, 5, May 1971, 335-345.

NILSSON, N. J. *Problem-Solving Methods in Artificial Intelligence.* New York: McGraw-Hill, 1971.

NILSSON, N. J. *Principles of Artificial Intelligence.* Palo Alto, CA: Tioga, 1980.

PAPERT, S. "Uses of technology to enhance education." Technical Report 298, AI Lab, MIT, 1973.

PERSOON, E. "A new edge detection algorithm and its applications in picture processing." *CGIP 5*, 4, December 1976, 425-446.

RAMER, U. "Extraction of line structures from photographs of curved objects." *CGIP 4*, 2, June 1975, 81-103.

ROSENFELD, A. *Picture Processing by Computer.* New York: Academic Press, 1968.

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

SELFRIDGE, P. G., J. M. S. PREWITT, C. R. DYER, and S. RANADE. "Segmentation algorithms for abdominal computerized tomography scans." *Proc.*, 3rd COMPSAC, November 1979, 571-577.

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